<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 1</td>
<td>Augmenting Models</td>
<td>006</td>
</tr>
<tr>
<td>Section 2</td>
<td>Learning, Knowledge and Reasoning</td>
<td>134</td>
</tr>
<tr>
<td>Section 3</td>
<td>Optimal Action</td>
<td>194</td>
</tr>
<tr>
<td>Section 4</td>
<td>Perception</td>
<td>270</td>
</tr>
<tr>
<td>Section 5</td>
<td>Mastering Language</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>Bibliography</td>
<td>408</td>
</tr>
</tbody>
</table>
Artificial intelligence is everywhere.

On the surface, it seems like we are surrounded by an ocean of AI, spanning much of what we do in our personal and professional lives. But a closer look reveals that AI resides in many islands, rather than being the ocean itself. This book focuses on fundamental research in core techniques of AI that advance it from a narrow form of AI, able to execute specific, well-defined tasks in a single domain, to a broad form of AI that learns more generally and can work across different domains and problem sets. This research spans multiple areas including learning, reasoning, natural language processing, vision and planning.

Today, AI systems have been able to accomplish certain tasks with a level of accuracy surpassing that of humans. These achievements have been within the scope of narrow AI. The next step in the progression to broad AI – enabling AI that can generalize – is far from trivial and poses many challenges that we aim to begin addressing through our research. For example, most AI systems currently require enormous amounts of data to train deep learning models and they have little ability to transfer the knowledge gained through learning from one task to another. This differs greatly from the way humans learn. Imagine a little girl shown a photo of an animal she didn’t know before, and told its name. She easily grasps the nature of it, based on what she already knows about animals. She will now probably be able to identify this animal in other photos, in the wild and even when presented to her as a sketch.

Humans and AI approach tasks very differently today. Humans have a preconceived model of the world, we can learn and reason, and we combine input from all our sensors – sight, hearing, touch, etc. – to truly understand the task at hand. Most of AI today, however, is still falls in the narrow category, focusing on a single type of input and mapping it to the correct output with very little reasoning behind the scenes.

On top of enabling machines to learn at a basic level, we need to apply these abilities across the board – for speech recognition, computer vision, natural language understanding and combinations thereof in a multi-modal environment. Developing AI in different domains requires human research expertise in these domains to complement the progress in basic research. Our research is motivated by both long-term foundational problems that are at the heart of AI and key obstacles in enterprise adoption of AI.

This collection of papers is a subset of our research in advancing AI, from the most basic challenges that today’s learning machines and paradigms pose, to specific research that requires fine domain expertise. We walk the reader through the journey from narrow AI to the more human-like learning of broad AI.
Augmenting Models
Augmenting Models

In traditional machine learning, we use labeled data to train models for a specific task and domain. To be successful, this approach requires relatively large amounts of labeled data for each specific task and domain of interest. Unlike humans, models do not generalize well from one domain to new domains. We are working to address this limitation by using a technique called transfer learning (also referred to as domain adaptation) to enable a model to perform a new task by leveraging existing labeled data from a related task. The goal is to transfer as much knowledge as possible from the source domain to accelerate and improve learning in the target domain.

In addition to the augmentation of data, we are pursuing techniques to augment existing models. In some cases, model augmentation results in enormous numbers of nodes that make it intractable to perform complex inferencing procedures on the entire network. Therefore, seeking network embeddings that are scalable and can adapt to new domains is important.

This chapter highlights our papers on data and model augmentation, including: semi-supervised domain adaptation where labels are given only for the source domain [3], addressing cases in which the domains correspond to different contexts [2], joint formulation for data augmentation and network training [3], data synthesis for an unseen category from few seen examples [4], and sharing knowledge between multiple tasks where it’s helpful, but also allowing private space for different tasks to avoid forgetting [5].

In addition, several papers address network embedding with the first paper focused on learning the network representations with adversarially regularized autoencoders [6] and two additional papers on learning Generative Adversarial Networks that mimic the distribution of real data [7] and [8].

Selected Publications

1. Co-regularized Alignment for Unsupervised Domain Adaptation  
   p. 010

2. Domain Adaptation by Using Causal Inference to Predict Invariant Conditional Distributions  
   p. 024

3. Jointly Optimize Data Augmentation and Network Training: Adversarial Data Augmentation in Human Pose Estimation  
   p. 038

4. Δ-encoder: an effective sample synthesis method for few-shot object recognition  
   p. 048

5. Routing Networks: Adaptive Selection of Non-linear Functions for Multi-task Learning  
   p. 060

6. Learning Deep Network Representations with Adversarially Regularized Autoencoders  
   p. 078

7. Soblev GAN  
   p. 088

   p. 116
Co-regularized Alignment for Unsupervised Domain Adaptation

Abhishek Kumar  
MIT-IBM Watson AI Lab, IBM Research  
abhishk@us.ibm.com

Prasanna Sattigeri  
MIT-IBM Watson AI Lab, IBM Research  
psattig@us.ibm.com

Kahini Wadhawan  
MIT-IBM Watson AI Lab, IBM Research  
kahini.wadhawan@ibm.com

Leonid Karlinsky  
MIT-IBM Watson AI Lab, IBM Research  
leonidka@il.ibm.com

Rogerio Feris  
MIT-IBM Watson AI Lab, IBM Research  /rsferis@us.ibm.com

Bill Freeman  
MIT  
billf@mit.edu

Gregory Wornell  
MIT  
gww@mit.edu

Abstract

Deep neural networks, trained with large amount of labeled data, can fail to generalize well when tested with examples from a target domain whose distribution differs from the training data distribution, referred as the source domain. It can be expensive or even infeasible to obtain required amount of labeled data in all possible domains. Unsupervised domain adaptation sets out to address this problem, aiming to learn a good predictive model for the target domain using labeled examples from the source domain but only unlabeled examples from the target domain. Domain alignment approaches this problem by matching the source and target feature distributions, and has been used as a key component in many state-of-the-art domain adaptation methods. However, matching the marginal feature distributions does not guarantee that the corresponding class conditional distributions will be aligned across the two domains. We propose co-regularized domain alignment for unsupervised domain adaptation, which constructs multiple diverse feature spaces and aligns source and target distributions in each of them individually, while encouraging that alignments agree with each other with regard to the class predictions on the unlabeled target examples. The proposed method is generic and can be used to improve any domain adaptation method which uses domain alignment. We instantiate it in the context of a recent state-of-the-art method and observe that it provides significant performance improvements on several domain adaptation benchmarks.

1 Introduction

Deep learning has shown impressive performance improvements on a wide variety of tasks. These remarkable gains often rely on the access to large amount of labeled examples \((x, y)\) for the concepts of interest \((y \in Y)\). However, a predictive model trained on certain distribution of data \((\{(x, y): x \sim P_s(x)\})\), referred as the source domain, can fail to generalize when faced with observations pertaining to same concepts but from a different distribution \((x \sim P_t(x))\), referred as the target domain. This problem of mismatch in training and test data distributions is commonly referred as domain or covariate shift [34]. The goal in domain adaptation is to address this mismatch and obtain a model that generalizes well on the target domain with limited or no labeled examples from
the target domain. Domain adaptation finds applications in many practical scenarios, including the scenario where source domain consists of simulated or synthetic data (for which labels are readily available from the simulator) and target domain consists of real world observations [43, 39, 5].

We consider the problem of unsupervised domain adaptation where the learner has access to only unlabeled examples from the target domain. The goal is to learn a good predictive model for the target domain using labeled source examples and unlabeled target examples. Domain alignment [13, 16] approaches this problem by extracting features that are invariant to the domain but preserve the discriminative information required for prediction. Domain alignment has been used as a crucial ingredient in numerous existing domain adaptation methods [18, 26, 40, 4, 17, 44, 42, 35]. The core idea is to align distributions of points in the feature spaces belonging to the same concept class across the two domains (i.e., aligning $g \mathcal{P}x \mathcal{P}(y)$ and $g \mathcal{P}x \mathcal{P}(y)$ where $g$ is a measurable feature generator mapping and $g \mathcal{P}$ denotes the push-forward of a distribution $\mathcal{P}$). The prediction performance of target domain directly depends on the correctness of this alignment. However, the right alignment of class conditional distributions can be challenging to achieve without access to any labels in the target domain. Indeed, there is still significant gap between the performance of unsupervised domain adaptation classifiers with existing methods and fully-supervised target classifier, especially when the discrepancy between the source and target domains is high.

In this work, we propose an approach to improve the alignment of class conditional feature distributions of source and target domains for unsupervised domain adaptation. Our approach works by constructing two (or possibly more) diverse feature embeddings for the source domain examples and aligning the target domain feature distribution to each of them individually. We co-regularize the multiple alignments by making them agree with each other with regard to the class prediction, which helps in reducing the search space of possible alignments while still keeping the correct set of alignments under consideration. The proposed method is generic and can be used to improve any domain adaptation method that uses domain alignment as an ingredient. We evaluate our approach on commonly used benchmark domain adaptation tasks such as digit recognition (MNIST, MNIST-M, SVHN, Synthetic Digit5) and object recognition (CIFAR-10, STL1, STL2) and observe significant improvement over state-of-the-art performance on these.

2 Formulation

We first provide a brief background on domain alignment while highlighting the challenges involved while using it for unsupervised domain adaptation.

2.1 Domain Alignment

The idea of aligning source and target distributions for domain adaptation can be motivated from the following result by Ben-David et al. [2]:

**Theorem 1 (2):** Let $\mathcal{H}$ be the common hypothesis class for source and target. The expected error for the target domain is upper bounded as

$$
\mathbb{E}_{t}(h) \leq \mathbb{E}_{s}(h) + \frac{\alpha}{2} \Delta_{DA}(\mathcal{P}, \mathcal{P}_t) + \lambda, \forall h \in \mathcal{H},
$$

where $\Delta_{DA}(\mathcal{P}, \mathcal{P}_t) = \max_{\mathcal{P}_s} \mathbb{E}_{s}(h) - \mathbb{E}_{t}(h)$, $\lambda = \min\{\epsilon_h(s), \epsilon_h(t)\}$, and $\epsilon_h(s)$ is the expected error of $h$ on the source domain.

Let $g_s : X \rightarrow \mathbb{R}^m$ and $g_t : X \rightarrow \mathbb{R}^m$ be the feature generators for source and target examples, respectively. We assume $g_s = g_t = g$ for simplicity but the following discussion also holds for different $g_s$ and $g_t$. Let $g \mathcal{P}x$ be the push-forward distribution of source distribution $\mathcal{P}_s$ induced by $g$ (similarity for $g \mathcal{P}x$). Let $H$ be a class of hypotheses defined over the feature space $[g \mathcal{P}x] : x \sim \mathcal{P}_s \cup \mathcal{P}_t$ and it is a necessary condition for $\Delta_{DA}$ to vanish and there may exist sets of $\mathcal{P}_s, \mathcal{P}_t$, and $H$ for which $\Delta_{DA}$ is zero without $g \mathcal{P}x$ and $g \mathcal{P}x$ being well aligned (Fig. 1a). However, for unaligned $g \mathcal{P}x$ and $g \mathcal{P}x$, it is difficult to choose the appropriate hypothesis class $H$ with small $\Delta_{DA}$ and small $\lambda$ without access to labeled target data.

On the other hand, if source feature distribution $g \mathcal{P}x$ and target feature distribution $g \mathcal{P}x$ are aligned well, it is easy to see that the $\Delta_{DA}$-distance will vanish for any space $H$ of sufficiently smooth hypotheses. A small $\Delta_{DA}$-distance alone does not guarantee small expected error on the target domain (Fig. 1b): it is also required to have source and target feature distributions such that there exists a hypothesis $h \in H$ with low expected error $\lambda$ on both source and target domains.

For well aligned marginal feature distributions, having a low $\lambda$ requires that the corresponding class conditional distributions $g \mathcal{P}x \mathcal{P}(y)$ and $g \mathcal{P}x \mathcal{P}(y)$ should be aligned for all $y \in Y$ (Fig. 1c). However, directly pursuing the alignment of the class conditional distributions is not possible as we do not have access to target labels in unsupervised domain adaptation. Hence most unsupervised domain adaptation methods optimize for alignment of marginal distributions $g \mathcal{P}x \mathcal{P}(y)$ and $g \mathcal{P}x \mathcal{P}(y)$, hoping that the corresponding class conditional distributions will get aligned as a result.

There is a large body of work on distribution alignment which becomes readily applicable here. The goal is to find a feature generator $g$ or a pair of feature generators $g_s$ and $g_t$ such that $g \mathcal{P}x = g \mathcal{P}x$ are close. Methods based on minimizing various distances between the two distributions (e.g., maximum mean discrepancy [18, 44], suitable divergences and their approximations [15, 4, 35]) or matching the moments of the two distributions [41, 40] have been proposed for unsupervised domain adaptation.

2.2 Co-regularized Domain Alignment

The idea of co-regularization has been successfully used in semi-supervised learning [37, 38, 31, 36] for reducing the size of the hypothesis class. It works by learning two predictors in two hypothesis classes $H_1$ and $H_2$, respectively, while penalizing the disagreement between their predictions on the unlabeled examples. This intuitively results in shrinking the search space by ruling out predictors from $H_1$ that don’t have an agreeing predictor in $H_2$ (and vice versa) [36]. When $H_1$ and $H_2$ are reproducing kernel Hilbert spaces, the co-regularized hypothesis class has been formally shown to have a reduced Rademacher complexity, by an amount that depends on a certain data dependent distance between the two views [31]. This results in improved generalization bounds comparing with the best predictor in the co-regularized class (reduces the variance).

Suppose the true labeling functions for source and target domains are given by $f_s : X \rightarrow Y$ and $f_t : X \rightarrow Y$, respectively. Let $X^S = \{x : f_s(x) = y, x \sim \mathcal{P}_s\}$ and $X^T = \{x : f_t(x) = y, x \sim \mathcal{P}_t\}$.

- Heavily-tuned manual data augmentation can be used to bring the two domains closer in the observed space $X$ [34] but it requires the augmentation to be tuned individually for every domain pair to be successful.
- Seidhnan and Karak [36] show that the bias introduced by co-regularization is small when each view carries sufficient information about $Y$ on its own (i.e., mutual information $I(Y; X^S)$ and $I(Y; X^T)$ are small), and the generalization bounds comparing with the Bayes optimal predictor are also tight.
be the sets which are assigned label $y$ in source and target domains, respectively. As discussed in the earlier section, the hope is that alignment of marginal distributions $g \circ F_1$ and $g \circ F_2$ will result in aligning the corresponding class non-distributions $g \circ P_1(y)$ and $g \circ P_2(y)$ but it is not guaranteed. There might be sets $A x_1 \subset X_s$ and $A x_2 \subset X_t$, for $y \neq y_0$ such that their images under $g(x, y), g(x, y_0)$ do not align $(g(x) \cdot x \in \mathbb{R}^d$ and $g(x_0) \cdot x \in \mathbb{R}^d$ get aligned in the feature space, which is difficult to detect or correct in the absence of target labels.

We propose to use the idea of co-regularization to trim the space of possible alignments without ruling out the desirable alignments of class conditional distributions from the space. Let $G_1, G_2$ be the two hypothesis spaces for the feature generators, and $H_1, H_2$ be the hypothesis classes of predictors defined on the output of the feature generators from $G_1$ and $G_2$, respectively. We want to learn a $g \in G_1$ and a $h \in H_2$ such that $h \circ g$ minimizes the prediction error on the source domain, while aligning the source and target feature distributions by minimizing a suitable distance $D(g \circ F_1, g \circ F_2)$ for ($i, j$). To measure the disagreement between the alignments of feature distributions in the two feature spaces ($g \circ P_s, P_t$, for $i = 1, 2$), we look at the distance between the predictions $(h_1 \circ g(x) | x \in X_s)$ and $(h_2 \circ g(x) | x \in X_t)$ on unlabeled target examples $x \in \mathcal{P}_t$.

If the predictions agree, it can be seen as an indicator that the alignment of source and target feature distributions is similar across the two feature spaces induced by $g_1$ and $g_2$ (with respect to the predictor $h_2$). Coming back to the example of erroneous alignment given in the previous paragraph, if there is a $g \in G_1$ which aligns $g_1(A^s)$ and $g_2(A^t)$ but does not have any agreeing $g \in G_2$ with respect to the classifier predictions, it will be ruled out of consideration. Hence, ideally we want to construct $G_1$ and $G_2$ such that they induce complementary erroneous alignments of source and target distributions while each of them still contains the set of desirable feature generators that produce the right alignments.

The proposed co-regularized domain alignment (referred as Co-DA) can be summarized by the following objective function (denoting $f_i = h_i \circ g_i | i = 1, 2$):

$$\min_{f_i, \lambda_i, \psi_i} \mathcal{L}_d(f_1; P_s) + \lambda_1 D_g (g, P_s) + \mathcal{L}_d(f_2; P_t) + \lambda_2 D_g (g, P_t),$$

where, $\mathcal{L}_d(f_1; P_s) = \mathbb{E}_{x \sim P_s} \max \{y | f_1(x) \}$ is the usual cross-entropy loss for the source examples (assuming $f_i$ outputs the probabilities of classes and $y$ is the label vector), $\mathcal{L}_d(\cdot)$ is the loss term measuring the distance between the two distributions, $D_g (g, P_s) = \mathbb{E}_{x \sim P_s} d_g (g(x), x)$ where $d_g(\cdot)$ measures the disagreement between the two predictions for a target sample, and $D_g (g, P_t)$ quantifies the diversity of $g$ and $P_t$.

2.2.1 Algorithmic Instantiation

We make our approach of co-regularized domain alignment more concrete by making the following algorithmic choices:

**Domain alignment.** Following much of the earlier work, we minimize the variational form of the Jensen-Shannon (JS) divergence [29, 19] between source and target feature distributions [15, 34, 35]:

$$\mathcal{L}_d(g \circ P_s, P_t) = \mathbb{E}_{x \sim P_s} \max \{y | g(x) \} + \mathbb{E}_{x \sim P_t} \max \{y | -g(x) \}.$$  

(3)

where $d_g$ is the domain discriminator, taken to be a two layer neural network that outputs the probability of the input sample belonging to the source domain.

**Target prediction agreement.** We use $\ell_1$ distance between the predicted class probabilities (twice the total variation distance) as the measure of disagreement (although other measures such as JS-divergence are also possible):

$$\mathcal{L}_d(f_1, f_2; P_t) = \mathbb{E}_{x \sim P_t} \| f_1(x) - f_2(x) \|_1.$$  

(4)

Diverse $g_1$ and $g_2$. It is desirable to have $g_1$ and $g_2$ such that errors in the distribution alignments are different from each other and target prediction agreement can play its role. To this end, we encourage source feature distributions induced by $g_1$ and $g_2$ to be different from each other. There can be many ways to approach this; here we adopt a simpler option of pushing the minibatch means far apart:

$$D_g (g, P_t) := \mathbb{E}_{x \sim P_t} \min \{v \mid \sum_{i=1}^n (g_i(x) - g(x)) \}.$$  

(5)

The hyperparameter $\sigma$ is a positive real controlling the maximum disparity between $g_1$ and $g_2$. This is needed for stability of feature maps $g_1$ and $g_2$ during training; we empirically observed that having $\sigma$ as an initial value in their continued optimization from each other, helping the alignment of source and target distributions in both $G_1$ and $G_2$. Note that we only encourage the source feature distributions $g \circ P_s$ to be different in the sense of the corresponding target distributions $g \circ P_t$ to them will produce different alignments.

**Cluster assumption.** The large amount of target unlabeled data can be used to bias the classifier boundaries to pass through the regions containing low density data of points. This is referred to as the cluster assumption [7] which has been used for semi-supervised learning [20, 27] and was also recently used for unsupervised domain adaptation [35]. Minimization of the conditional entropy of $f_1(x)$ can be used to push the predictor boundaries away from the high density regions [20, 27, 35]. However, this alone may result in overfitting to the unlabeled examples if the classifier has high capacity. To avoid this, a virtual adversarial training (VAT) [27] has been successfully used in conjunction with conditional entropy minimization to smooth the classifier surface around the unlabeled points [27, 35]. We follow this line of work and add the following additional loss term for conditional entropy minimization and VAT to the objective in (2):

$$L_{vat}(f_1; P_s) = -\mathbb{E}_{x \sim P_s} \max (\mathcal{L}(f_1(x)), \mathcal{L}(f_2(x)) + \max_{b \neq \mathcal{L}(f_1(x))} \mathcal{L}(b(x))).$$

(6)

We also use VAT loss $L_{vat}(f_1; P_s)$ on the source domain examples following Shu et al. [35]. Our final objective is given as:

$$\min_{\mathcal{L}(f_1), D_g, \psi_i, \lambda_i} \mathcal{L}(f_1) + \lambda_1 D_g (g, P_s) + \mathcal{L}(f_2) + \lambda_2 D_g (g, P_t),$$

(7)

Remarks:

(1) The proposed co-regularized domain alignment (CoDA) can be used to improve any domain adaptation method that has a domain alignment component in it. We instantiate it in the context of a recently proposed method VADA [36], which has the same objective as $\mathcal{L}(f_1)$ in Eq. (7) and has shown state-of-the-art results on several datasets. Indeed, we observe that co-regularized domain alignment significantly improves upon these results.

(2) The proposed method can be naturally extended to more than two hypothesis classes, however we limit ourselves to two hypothesis classes in the empirical evaluations.

3 Related Work

**Domain Adaptation.** Due to the significance of domain adaptation in reducing the need for labeled data, there has been extensive activity on it during past several years. Domain alignment has almost become a representative approach for domain adaptation, acting as a crucial component in many recently proposed methods [18, 26, 41, 4, 17, 44, 42, 35]. The proposed co-regularized domain alignment framework is applicable in all such methods that utilize domain alignment as an ingredient. Perhaps most related to our proposed method is a recent work by Saito et al. [33], who proposed directly optimizing a proxy for $\text{KL}(\cdot, \cdot)$-distance [2] in the context of deep neural networks. Their model consists of a single feature generator that feeds to two different multi-layer NN classifiers $h_1$ and $h_2$. Their approach alternates between two steps: (i) For a fixed $g$, finding $h_1$ and $h_2$ such that the discrepancy or disagreement between the predictions $h_1(g(x))$ and $h_2(g(x))$ is maximized for $x \sim P_t$; (ii) For fixed $h_1$ and $h_2$, find $g$ which minimizes the discrepancy between the predictions $h_1(g(x))$ and $h_2(g(x))$ for $x \sim P_t$. Our approach also has a discrepancy minimization term over the predictions for target samples but the core idea in our approach is fundamentally different where we want to have diverse feature generators $g_1$ and $g_2$ that induce different alignments for source and target populations, and which can correct each other’s errors by minimizing disagreement between them as measured by target predictions. Further, unlike [33] where the discrepancy is maximized at the final predictions $(h_1 + h_2)(x)$ and $(h_2 + h_2)(x)$ (Step (i)), we maximize diversity at the output of
feature generators $g_1$ and $g_2$. Apart from the aforementioned approaches, methods based on image translations across domains have also been proposed for unsupervised domain adaptation [24, 28, 6].

Co-regularization and Co-training. The related ideas of co-training [3] and co-regularization [37, 36] have been successfully used for semi-supervised learning as well as unsupervised learning [22, 21]. Chen et al. [8] used the idea of co-training for semi-supervised domain adaptation (assuming a few labelled examples are available) by finding a suitable split of the two sets based on the notion of $\epsilon$-expansibility [1]. A related work [9] used the idea of co-regularization for semi-supervised domain adaptation but their approach is quite different from our method where they learn different classifiers for source and target, making their predictions agree on the unlabelled target samples. Trs-training [45] can be regarded as an extension of co-training [3] and uses the output of three different classifiers to assign pseudo-labels to unlabelled examples. Suo et al. [32] proposed Asymmetric tri-training for unsupervised domain adaptation where one of the three models is learned only on pseudo-labeled target examples. Asymmetric trs-training, similar to [33], works with a single feature generator $g$ which feeds to three different classifiers $h_1$, $h_2$ and $h_3$.

Ensemble learning. There is an extensive line of work on ensemble methods for neural nets which combine predictions from multiple models [11, 10, 30, 25, 23]. Several ensemble methods also combine predictions from multiple models [11, 10, 30, 25, 23]. However, ensemble methods have been shown to be quite popular for improving the diversity among models and can be used to improve the accuracy of the model. In our experiments we found that the use of a classifier which is more diverse can improve the accuracy of the model.

4 Experiments

We evaluate the proposed Co-regularized Domain Alignment (Co-DA) by instantiating it in the context of a recently proposed method VADA [35] which has shown state-of-the-art results on several benchmarks, and observe that Co-DA yields further significant improvements on it, establishing new state-of-the-art in several cases. For a fair comparison, we evaluate on the same datasets as used by Shu et al. [35], MNIST, SVHN, MNIST-M, Synthetic Digits, CINIC-10. We completely follow the implementation on the code released by the authors' to rule out incidental differences due to implementation specific details.

Network architecture. VADA [35] has three components in the model architecture: a feature generator $g$, a feature classifier $h$ that takes output of $g$ as input, and a domain discriminator $d$ for domain alignment (Eq. 3). Their data classifier $f = h \circ g$ consists of nine convolutional layers followed by a global pooling and a classifier, and is trained on the other datasets (referred as the DIRT-T phase) while keeping it from moving too far away. If $\lambda$ is the classifier at iteration $n$ (Eq. 5) over $10^{\lambda - 1}$ (the grid for $\lambda$ was chosen to be $[0, 10^{9}]$) [35]. The hyperparameter for target prediction alignment, $\lambda_t$, was obtained by a search over $[0.1, 10^{9}]$. For hyperparameters in the diversity term, we fix $\lambda_{div} = 10^{-2}$ and do a grid search for $\nu$ (Eq. 5) over $[1, 5, 10^{10}]$. The hyperparameters were randomly selected from the target labeled training data. For hyperparameters in validation, following [35, 32]. We completely follow [35] for training our model, using Adam Optimizer ($\nu = 0.001, \beta_1 = 0.5, \beta_2 = 0.999$) with Polyak averaging (i.e., an exponential moving average with momentum=0.999 on the parameter trajectory), and train the models in all experiments as in [35].

Baselines. We primarily compare with VADA [35] to show that co-regularized domain alignment can provide a further improvement over state-of-the-art results. We also show results for the diversity loss term (i.e., $\lambda_{div} = 0$) to test the failure of explicitly encouraging diversity through Eq. 5 (note that some diversity can arise even with $\lambda_{div} = 0$, due to different random seeds, and Gaussian noise in dropout layers present in [35]). Shu et al. [35] also propose to incrementally refine the learned VADA model by shifting the classifier boundaries to pass through low density regions of target domains (referred as the DIRT-T phase) while keeping it from moving too far away. If $f^*$ is the classifier at iteration $n$ (Eq. 5) being the solution of VADA, the new classifier is obtained as $f^{n+1} = \arg\min_{f^*} \lambda_t(L_{div}(f^{n+1}, f^*)) + L_{ce}(f^{n+1}, Y) + 3 \cdot E_{c}(\mathbb{E}_{D}(f^*)^{(i)}(x))$. We also perform DIRT-T refinement individually on each of the two trained hypotheses obtained
with Co-DA (i.e., $f^\star_1$, $f^\star_2$) to see how it compares with DIRT-T refinement on the VADA model [35]. Note that DIRT-T refinement phase is carried out individually for $f^\star_1$ and $f^\star_2$ and there is no co-regularization term connecting the two in DIRT-T phase. Again, following the evaluation protocol in [35], we train DIRT-T for $\{200, 400, 600, 800\}$ iterations, with number of iterations taken as a hyperparameter. We do not perform any hyperparameter search for $\beta$ and the values for $\beta$ are fixed to what were reported in [35] for all datasets. Apart from VADA, we also show comparisons with other recently proposed unsupervised domain adaptation methods for completeness.

### 4.1 Domain adaptation results

We evaluate Co-DA on the following domain adaptation benchmarks. The results are shown in Table 1. The two numbers A/B in the table for the proposed methods are the individual test accuracies for both classifiers which are quite close to each other at convergence.

**MNIST $\rightarrow$ SVHN.** Both MNIST and SVHN are digits datasets but differ greatly in style: MNIST consists of gray-scale handwritten digits whereas SVHN consists of house numbers from street view images. This is the most challenging domain adaptation setting in our experiments (many earlier domain adaptation methods have omitted it from the experiments due to the difficulty of adaptation). VADA [35] showed good performance (73.3%) on this challenging setting using instance normalization but without using any data augmentation. The proposed Co-DA improves it substantially $\sim 81.7\%$, even surpassing the performance of VADA+DIRT-T (76.5%) [35]. Figure 2 shows the test accuracy as training proceeds. For the case of no instance-normalization as well, we see a substantial improvement over VADA from 47.5\% to 52\% using Co-DA and 55.3\% using Co-DA*. Applying iterative refinement with DIRT-T [35] further improves the accuracy to 88\% with instance norm and $\sim 60\%$ without instance norm. This sets new state-of-the-art for MNIST $\rightarrow$ SVHN domain adaptation without using any data augmentation.

**SVHN $\rightarrow$ MNIST.** This adaptation direction is easier as MNIST as the test domain is easy to classify and performance of existing methods is already quite high (97.9\% with VADA). Co-DA is still able to yield reasonable improvement over VADA, of about $\sim 1\%$ for no instance-normalization, and $\sim 4\%$ with instance-normalization. The application of DIRT-T after Co-DA does not give significant boost over VADA+DIRT-T as the performance is already saturated with Co-DA (close to 95\%).

**MNIST $\rightarrow$ MNIST-M.** Images in MNIST-M are created by blending MNIST digits with random color patches from the BSDS500 dataset. Co-DA provides similar improvements over VADA as the earlier setting of SVHN $\rightarrow$ MNIST, of about $\sim 1\%$ for no instance-normalization, and $\sim 2\%$ with instance-normalization.

**Syn-DIGITS $\rightarrow$ SVHN.** Syn-DIGITS data consists of about 50k synthetics digits images of varying positioning, orientation, background, stroke color and amount of blur. We again observe reasonable improvement of $\sim 1\%$ with Co-DA over VADA, getting close to the accuracy of a fully supervised target model for SVHN (without data augmentation).

**CIFAR $\rightarrow$ STL.** CIFAR has more labeled examples than STL hence CIFAR $\rightarrow$ STL is easier adaptation problem than STL $\rightarrow$ CIFAR. We observe more significant gains on the harder problem of STL $\rightarrow$ CIFAR, with Co-DA improving over VADA by $\sim 3\%$ for both with- and without instance-normalization.

### 5 Conclusion

We proposed a co-regularization based domain alignment framework for unsupervised domain adaptation. Our framework is applicable to any domain adaptation method that relies on explicitly aligning the source and target domains in a feature space. We instantiated it in the context of a state-of-the-art domain adaptation method and observe that it provides improved performance on some commonly used domain adaptation benchmarks, with substantial gains in the more challenging tasks, setting new state-of-the-art in these cases.
References


Abstract

An important goal common to domain adaptation and causal reasoning is to make accurate predictions when the distributions for the target domain(s) and the source domain(s) differ. We consider the case in which the domains correspond to different contexts in which a system has been measured, for example, a purely observational context and several interventional contexts in which the system has been perturbed by external interventions. We consider a class of such causal domain adaptation problems, where data for multiple source domains are given, and the task is to predict the distribution of a certain target variable from measurements of other variables in one or more target domains. We propose an approach for solving these problems that exploits causal inference and does not rely on prior knowledge of the causal graph, not of the type of the interventions or the intervention targets. We propose a practical implementation of the approach and evaluate it on simulated and real world data.

1 Introduction

Predicting unknown values based on observed data is a problem central to many sciences, and well studied in statistics and machine learning. This problem becomes significantly harder if the training and test data do not have the same distribution because they come from different domains. Such a distribution shift will happen in practice whenever the circumstances under which the training data were gathered are different from those for which the predictions are to be made. A rich literature exists on this problem of domain adaptation, a particular task in the field of transfer learning, see e.g. Quiñonero-Candela et al. (2009); Pan and Yang (2010) for overviews.

When the domain changes, so may the relations between the different variables under consideration. While for some (sets of) variables $A$, a function $f : A \rightarrow Y$ learned in one domain may continue to offer good predictions for $Y \in Y$ in a different domain, this may not be true of other sets $A'$ of variables. Causal graphs (e.g., Pearl, 2009; Spirtes et al., 2000) allow us to reason about this in a principled way when the domains correspond to different external interventions on the system, or more generally, to different contexts in which a system has been measured. Knowledge of the causal graph that describes the data generating mechanism, and of which parts of the model are invariant...
Augmenting Models

Causal graph

(b) No distribution shift for \( \{X_1\} \):
\[
P(Y | X_1, C_1 = 0) = P(Y | X_1, C_1 = 1)
\]

Suppose further that the relation between \( X_1 \) and \( X_2 \) is about equally strong as the relation between \( X_2 \) and \( X_3 \), but considerably more noisy. Then a feature selection method using only available source domain data, and aiming to select the best subset of features to use for prediction of \( Y \) will prefer both \( \{X_1, X_2\} \) and \( \{X_1, X_3\} \) over \( \{X_2\} \) (because predicting \( Y \) from \( X_2 \) leads to larger variance than predicting \( Y \) from \( X_1 \)), and to a larger bias than predicting \( Y \) from both \( X_1 \) and \( X_3 \). However, under the intervention \( (C_1 = 1) \), \( P(Y | X_1) \) and \( P(Y | \{X_1, X_3\}) \) both change, \(^1\) so that using those features to predict \( Y \) in the target domain could lead to extreme bias, as illustrated in Figure 1(c).

This example provides an instance of a domain adaptation problem where feature selection methods that do not take into account the causal structure would pick a set of features that does not generalize to the target domain, and may lead to arbitrarily bad predictions (even asymptotically, as the number of data points tends to infinity). On the other hand, correctly taking into account the causal structure and the possible distribution shift from source to target domain allows to upper bound the prediction error in the target domain, as we will see in Section 2.3.

2.1 Problem Setting

Let us consider a system of interest described by a set of system variables \( \{X_j\}_{j \in J} \). In addition, we model the domain in which the system has been measured by context variables \( \{C_i\}_{i \in I} \) (we will use “context” as a synonym for “domain”). We will denote the tuple of all system and context variables as \( V = \{X_j, C_i\}_{j \in J, i \in I} \). System and context variables can be discrete or continuous. As a concrete example, the system of interest could be a mouse. The system variables could be blood cell phenotypes such as the concentration of red blood cells, the concentration of white blood cells, and the mean red blood cell volume. The context variables could indicate for example whether a certain gene has been knocked out, the dosage of a certain drug administered to the mice, the age and gender of the mice, or the lab in which the measurements were done. The important underlying assumption is that context variables are exogenous to the system, whereas system variables are endogenous. The interventions are not limited to the perfect (“surgical”) interventions modeled by the do-operator of Pearl (2000), but can also be other types of interventions such as mechanism changes (Tian and Pearl, 2002), activity interventions (Markowtiz et al., 2005), faith-based interventions (Koopman and Meng, 2007), activity interventions (Mooij and Heskes, 2013), and stochastic versions of all these. Knowledge of the intervention targets is not necessary (but is certainly helpful). For example, administering a drug to the mice may have a direct causal effect on an unknown subset of the system variables, but we can simply model it as a binary exogenous variable (indicating whether or not the drug was administered) or a continuous exogenous variable (describing the dosage of the administered drug) without specifying in advance on which variables it has a direct effect. We can now formally state the domain adaptation task that we address in this work.

Task 1 (Domain Adaptation Task). Given data for a single or multiple source domains, in each of which \( C_i = 0 \), and for a single or multiple target domains, in each of which \( C_i = 1 \), assume the source domains data is complete (i.e., no missing values), and the target domains data is complete with the exception of all values of a certain target variable \( Y \). The task is to predict these values of the target variable \( Y \) given the available source and target domains data.

More precisely, we should say that \( P(Y | X_j, C_i = 0) \) may differ from \( P(Y | X_j, C_i = 1) \), and similarly when conditioning on \( \{X_j, C_i\} \).
Assumption 2. Let \( G \) be a causal graph with variables \( V \) consisting of system variables \( \{X_i| i \in J\} \) and context variables \( \{C_i| i \in I\} \), and \( \mathbb{P}[V] \) be the corresponding distribution on \( V \). Let \( C_i \) be the source/target domain indicator and \( Y - X_i \) the target variable.

(i) The distribution \( \mathbb{P}[V] \) is Markov and faithful w.r.t. \( G \).

(ii) Any conditional independence involving \( Y \) in the source domains also holds in the target domains, i.e., if \( A \cup B \subseteq S \) contains \( Y \) but not \( C_i \) then:
\[
A \perp B \mid S \mid \{C_i| i \in I\} = 0 \quad \Rightarrow \quad A \perp B \mid S \mid \{C_i| i \in I\} = 1.
\]

(iii) \( C_i \) has no direct effect on \( Y \) w.r.t. \( V, i.e., C_i \to Y \not\in G \).

The Markov and faithfulness assumption is standard in constraint-based causal discovery on a single domain, we apply it here on the “meta-system” composed of system and context. Note that Assumption 2(i) does not exclude the possibility of additional independences holding in the target domains, e.g., when \( C_i \) models a perfect surgical intervention. Note further that Assumption 2(iii) gets weaker the more relevant system variables are observed. In the next subsections, we will discuss how these assumptions enable us to address the domain adaptation task.

2.3 Separating Sets of Features

Our approach to addressing Task 1 is based on finding a separating set \( A \subseteq \{X_i| i \in J\} \cup \{C_i| i \in I\} \) of (context and system) variables that satisfies \( C_i \perp Y \mid A \mid \mathcal{A}[G] \). If such a separating set \( A \) can be found, then the distribution of \( Y \) on \( \mathcal{A}[G] \) is invariant under transferring from the source domains to the target domains, i.e., \( \mathbb{P}[Y \mid A \mid \mathcal{A}[G], C_i = 0 \mid 0] = \mathbb{P}[Y \mid A \mid \mathcal{A}[G], C_i = 1 \mid 1] \). As the former conditional distribution can be estimated from the source domains data, we directly obtain a prediction for the latter, which then enables us to predict the values of \( Y \) from the observed values of \( A \) in the target domains.3

We will now discuss the effect of the choice of \( A \) on the quality of the predictions. For simplicity of the exposition, we make use of the squared loss function and ignore finite-sample issues. When predicting \( Y \) from a subset of features \( \mathcal{A} \subseteq \{X_i| i \in J\} \cup \{C_i| i \in I\} \), the optimal predictor is defined as the function \( Y \) mapping the domain of \( A \) to the domain of \( Y \) that minimizes the target domains error \( \mathbb{E}[\hat{Y} - Y]^2 \mid C_i = 1 \). and is given by the conditional expectation (regression function) \( \hat{Y}^0(a) := \mathbb{E}[Y \mid A = a, C_i = 1] \). Since \( Y \) is not observed in the target domains, we cannot directly estimate this regression function from the data.

Our approach that is often used in practice is to ignore the difference in distribution between source and target domains, and use instead, the predictor \( \hat{Y}^0(a) := \mathbb{E}[Y \mid A = a, C_i = 0] \), which maximizes the source domains error \( \mathbb{E}[\hat{Y} - Y]^2 \mid C_i = 0 \). This approximation introduces a bias \( \hat{Y}^1 - \hat{Y}^0 \) that we will refer to as the transfer bias (when predicting \( Y \) from \( A \)). When ignoring that source domains and target domains have different distributions, any standard machine learning method can be used to predict \( Y \) from \( A \). As the transfer bias can become arbitrarily large (as we have seen in Example 1), the prediction accuracy by this solution strategy may be arbitrarily bad.

Instead, we propose to only predict \( Y \) from \( A \) when the set \( A \) of features satisfies the following separating set property:
\( C_i \perp Y \mid A \mid \mathcal{A}[G] \), (2)

i.e., \( A \) separates \( C_i \) from \( Y \) in \( \mathcal{G}[G] \) by the Markov assumption, this implies \( C_i \perp Y \mid A \mid \mathbb{P}[V] \). In other words, for separating sets, the distribution of \( Y \) conditional on \( A \) is invariant under transferring from the source domains to the target domains, i.e., \( \mathbb{P}[Y \mid A \mid \mathcal{A}[G], C_i = 0 \mid 0] = \mathbb{P}[Y \mid A \mid \mathcal{A}[G], C_i = 1 \mid 1] \).

This assumption can be weakened further: in some circumstances one can infer from the source data and the other assumptions that \( C_i \) cannot have a direct effect on \( Y \). For example: if there exists a descendant \( D \in \mathcal{D}(Y) \), and if there exists a set \( S \subseteq \mathcal{D}(Y) \) for which \( Y \) and \( C_i \) are independent in the source domains distribution \( \mathbb{P}[V \mid S \mid \mathcal{A}[G], C_i = 0 \mid 0] = \mathbb{P}[Y \mid S \mid \mathcal{A}[G], C_i = 1 \mid 1] \), then \( C_i \) and \( Y \) are not a direct cause of \( Y \) w.r.t. \( V \). For some proposals on alternative assumptions that can be made when this assumption is violated, see e.g., (Schölkopf et al., 2012; Zhang et al., 2013; 2015; Gong et al., 2016).

This trivial observation is not novel; see e.g. (Ch. 7, p. 164, Spirtes et al., 2000). It also follows as a special case of (Theorem 2, Pearl and Bareinboim, 2011). The main novelty of this work is the proposed strategy to identify such separating sets.
Augmenting Models

Rather than characterising by hand all possible situations in which a separating set can be identified (like in Example 2), in this work we delegate the causal reasoning to an automatic theorem prover. Intuitively, the idea is to provide the automatic theorem prover with the conditional (in)dependencies that hold in the data, in combination with an encoding of Assumptions 1 and 2 into logical rules, and ask the theorem prover whether it can prove that $C_1 \perp \perp Y \mid A$ holds for a candidate set $A$ from the assumptions and provided conditional (in)dependencies. There are three possibilities: either it can prove the query (and then we can proceed to predict $Y$ from $A$ and get an estimate of the target domains risk), or it can disprove the query (and then we know $A$ will generically give predictions that suffer from an arbitrarily large transfer bias), or it can do neither (in which case hopefully another subset $A$ can be found that does provably satisfy (2)).

2.5 Implementation Details

A simple (brute-force) implementation of our proposed approach is the following: By using a standard feature selection method, produce a ranked list of subsets $A \subseteq \{Y, C_1\}$ ordered ascendingly with respect to the empirical source domains risks. Going through this list of subsets (starting with the one with the smallest empirical source domains risk), test whether the separating set property can be inferred from the data by querying the automated theorem prover. If (2) can be shown to hold, use that subset $A$ for prediction of $Y$ and stop; if not, continue with the next candidate subset $A$ in the list. If no subset satisfies (2), abstain from making a prediction.

An important consequence of Assumption 2(ii) is that it enables us to transfer conditional independence involving the target variable from the source domains to the target domains (proof provided in the Supplementary Material):

Lemma 1. Under Assumption 2,

$$A \perp B \mid S \setminus \{C_1\} \iff A \perp B \mid \{S \cup \{C_1\}\}$$

for subsets $A, B, S \subseteq V$ such that their union contains $Y$ but not $C_2$.

To test the separating set condition (2), we use the approach proposed by Hyttinen et al. (2014), where we simply add the ICA assumptions (Assumption 1) as constraints on the optimization problem, in addition to the domain-adaptation specific assumption that $C_1 \perp \perp Y \mid \{A\}$ (Assumption 2(iii)). As inputs we use all directly testable conditional independence test $p$-values $p_{A \perp B \mid S}$ in the pooled data and all those resulting from Lemma 1 when $S = \{A\}$ and $B \subseteq \{Y, C_1\}$ (when $A \subseteq \{Y, C_1\} \setminus S$). If background knowledge on intervention targets or the causal graph is available, it can easily be added as well. We use the method proposed by Magliacane et al. (2016) to query for the confidence of whether some statement (e.g., $Y \perp \perp A$) is true or false. The results of Magliacane et al. (2016) show that this approach is sound under oracle inputs, and asymptotically consistent whenever the statistical conditional independence tests used are asymptotically consistent.

In other words, in this way the probability of wrongly deciding whether a subset $A$ is a separating set converges to zero as the sample size increases. We chose this approach because it is simple to implement on top of existing open source code. Note that the computational cost quickly increases with the number of variables, limiting the number of variables that can be considered simultaneously.

One remaining issue is how to predict $Y$ when an optimal separating set $A$ has been found. As the distribution of $A$ may shift when transferring from source domains to target domains, this means that there is a covariate shift to be taken into account when predicting $Y$. Any method (e.g., least-squares regression) could in principle be used to predict $Y$ from a given set of covariates, but it is advisable to use a prediction method that works well under covariate shift, e.g., (Sugiyama et al., 2008).
3 Evaluation

We perform an evaluation on both synthetic data and a real-world dataset based on a causal inference challenge. The latter dataset consists of hematology-related measurements from the International Mouse Phenotyping Consortium (IMPC), which collects measurements of phenotypes of mice with different single-gene knockouts.

In both evaluations we compare a standard feature selection method (which uses Random Forests) with our method that builds on top of it and selects from its output the best separating set. First, we score all possible subsets of features by their out-of-bag score using the implementation of Random Forest Regressor from scikit-learn (Pedregosa et al., 2011) with default parameters. For the baseline we then select the best performing subset and predict $Y$. Instead, for our proposed method we try to find a subset of features $A$ that is also a separating set, starting from the subsets with the best scores. To test whether $A$ is a separating set, we use the method described in Section 2.5, using the ASP solver clingo. (Gebser et al., 2014). We provide as inputs the independence test results from a partial correlation test with significance level $\alpha = 0.05$ and combine it with the weighting scheme from (Magliacane et al., 2016). We then use the first subset $A$ in the ranked list of predictive sets of features found by the Random Forest method for which the confidence that $C_\perp Y \mid A$ holds is positive. If there is no set $A$ that satisfies this criterion, then we abstain from making a prediction.

For the synthetic data, we generate randomly 200 linear acyclic models with latent variables and Gaussian noise, each with three system variables, and sample $Y$ data points each for the observational and two experimental domains, where we simulate soft interventions on randomly selected targets, focusing on small, medium and large perturbations. We randomly select which intervention variable will be $C_1$ and which system variable will be $Y$. We disallow direct effects of $C_1$ on $Y$, and enforce that no intervention can directly affect all variables simultaneously. Figure 3a shows a boxplot of the $L_2$ loss of the predicted $Y$ values with respect to the true values for both the baseline and our method, considering the 120 cases out of 200 in which our method does produce an answer. In particular, Figure 3a considers the case of $N = 1000$ samples per regime and interventions that all produce a large perturbation. In the Supplementary Material we show that results improve with more samples, both for the baseline, but even more so for our method, since the quality of the conditional independence tests improves. We also show that, according to expectations, if the target distribution is very similar to the source distributions, i.e., the transfer bias is small, our method does not provide any benefit and seems to perform worse than the baseline. Conversely, the larger the intervention effect, the bigger the advantage of using our method.

For the real-world dataset, we select a subset of the variables considered in the CRM Causal Inference Challenge. Specifically, for simplicity we focus on 16 phenotypes that are not deterministically related to each other. The dataset contains measurements for 441 “wild type” mice and for about 10 “mutant” mice for each of 13 different single gene knockouts. We then generate 1000 datasets by randomly selecting subsets of 3 variables and 2 gene knockouts interventions, and always include also “wild type” mice. For each dataset we randomly choose $Y$ and $C_1$, and remove the values of $Y$ for $C_1 = 1$. Figure 3b shows a boxplot of the $L_2$ loss of the predicted $Y$ values with respect to the true values for the baseline and our method. Given the small size of the datasets, this is a very challenging problem. In this case, our method does not perform better than the baseline, and abstains from making a prediction for 170 cases out of 1000.

4 Discussion and Conclusion

We have identified a general class of causal domain adaptation problems and proposed a method that can identify sets of features that lead to transferable predictions. Our assumptions are very general and do not require the true causal graph or the intervention targets to be known. The method gives promising results on simulated and real-world data. More work remains to be done on the implementation side, for example, scaling up to more variables. We hope that this work will also inspire further research on the interplay between bias, variance and causality from a statistical learning theory perspective.

References


Supplementary material

Proofs

Proof of Lemma 1. First of all, \( A \perp B \mid S \mid \{C_1\} \) implies (by definition) \( A \perp B \mid S \cup \{C_1\} \). Second, \( A \perp B \mid S \mid \{C_1\} \) implies (by assumption) \( A \perp B \mid S \mid \{C_1\} \), and taken together, we get \( A \perp B \mid S \mid \{C_1\} \). By the Markov and faithfulness assumption (Assumption 2(i)), this holds iff \( A \perp B \mid S \cup \{C_1\} \).

\( \square \)

Proof of Example 2. In the JCI setting, we assume that in the full ADMG \( G \) over variables \( \{X_1, X_2, X_3, X_4\} \), \( C_1 \) and \( C_2 \) are confounded and not caused by system variables \( X_1, X_2, X_3 \). Furthermore, no pair of system variable and intervention variable is confounded.

In the context \( \{C_1 \mid C_2 \} \), if the conditional independences \( C_2 \perp X_2 \mid X_1 \mid \{C_1 \} \) and \( C_2 \perp X_2 \mid X_1 \mid \{C_1 \} \) hold, then we can also derive that \( C_2 \perp X_2 \mid X_1 \mid \{C_1 \} \), for example using Rule (9) from Maglaiuchie et al. (2016). Morever, we know that \( C_2 \) is not caused by \( X_2 \) and \( X_4 \), or in other words \( X_1 \notin \mathcal{I} \cap \mathcal{C} \) and \( X_3 \notin \mathcal{I} \cap \mathcal{C} \). Thus we conclude that \( (C_2, X_2, X_3) \) is an LCD triple (Cooper, 1997) in the context \( C_1 = 0 \). Since in addition, in this case \( C_2 \) and \( X_1 \) are unconfounded, the marginal ADMG \( G' \) on \( \{C_2, X_2, X_3\} \) (in the context \( C_1 = 0 \), and hence by Lemma 1 in all contexts) must be given by Figure 4a.

Therefore, the extended marginal ADMG \( G'' \) on variables \( \{C_1, C_2, X_2, X_3\} \) must also have a directed path from \( C_2 \) to \( X_1 \) and from \( X_1 \) to \( X_2 \). \( C_2 \) cannot be on these paths, as none of the variables causes \( C_1 \) and therefore \( G'' \) also contains the directed edges \( C_2 \rightarrow X_1 \) and \( X_1 \rightarrow X_2 \). Moreover, \( G'' \) cannot contain any edge between \( C_2 \) and \( X_2 \), nor a bidirected edge between \( X_1 \) and \( X_2 \), because that would violate the conditional independence. By construction, in the ICI setting there is a bidirected edge between \( C_1 \) and \( C_2 \), and that is the only bidirected edge connecting to \( C_1 \) or \( C_2 \). As we assumed there is no direct effect of \( C_1 \) on target \( X_1 \), there is no edge between \( C_1 \) and \( X_2 \) in \( G'' \). There is also no directed edge \( X_1 \rightarrow C_1 \) in \( G'' \), as the ICI assumption implies none of the other variables causes \( C_1 \). Therefore, the marginal ADMG \( G'' \) is given by Figure 4b, either with the directed edge \( C_1 \rightarrow X_1 \) present, or without that edge.

If it additionally holds that \( C_2 \perp X_2 \mid X_1 \mid \{C_1\} \), we have two possibilities:

1. If \( C_2 \perp X_1 \mid \{C_1\} \) holds, then \( X_3 \) is not caused by \( C_2 \). This means it cannot be on any directed path from \( C_2 \) to \( X_1 \), from \( X_1 \) to \( X_2 \), or be a descendant of \( X_2 \). Therefore the full ADMG \( G \) also necessarily contains the directed edges \( C_2 \rightarrow X_1 \) and \( X_1 \rightarrow X_2 \).

2. If \( C_2 \perp X_1 \mid \{C_1\} \) holds, then in conjunction with \( C_2 \perp X_2 \mid X_1 \mid \{C_1\} \) we can derive \( X_2 \rightarrow X_1 \), for example using Rule (5) from Maglaiuchie et al. (2016). This means \( X_3 \) must be a descendant of \( X_2 \) in the full ADMG \( G \), which implies it cannot be on the directed path from \( C_2 \) to \( X_1 \), or on the one from \( X_1 \) to \( X_2 \). Therefore the full ADMG \( G \) also necessarily contains the directed edges \( C_2 \rightarrow X_1 \) and \( X_1 \rightarrow X_2 \).

Because of the independence statements and ICI assumptions, there cannot be a bidirected edge between \( X_3 \) and \( X_4 \). Similarly, there cannot be directed edges from \( X_1 \) to one of those nodes. The edges \( X_1 \rightarrow X_2 \) and \( C_2 \rightarrow X_3 \) must also be absent.
In both cases, there can be a directed edge from $C_1$ to $X_3$. Therefore, the full ADMG $\mathcal{G}$ is given by Figure 4c. In all cases we see that $C_1 \perp X_2 \mid X_3$ [g], and we conclude that $\{X_1\}$ is a valid separating set.

If the ADMG is as in Figure 2, then a standard feature selection method would asymptotically prefer the subset $\{X_1, X_3\}$ to predict $X_2$ over the subset $\{X_1\}$ (note that the Markov blanket of $X_2$ in context $\{C_1\} = 0$ is $\{X_1, X_2\}$). As a result, any prediction method trained on all available features using source domain data (i.e., in context $\{C_1\} = 0$) may incur a possibly unbounded prediction error when used to predict $X_2$ in the target domain $\{C_1\} = 0$ (for example, if $X_2$ is an almost deterministic copy of $X_2$ if $C_1 = 0$, but has a drastically different distribution if $C_1 = 1$).

Additional results on synthetic data

We provide some additional results on the synthetic data. We generate randomly 200 linear acyclic models with a small random number of latent variables and Gaussian noise, with each having three system variables. Similarly to the evaluation in the main paper, we sample $N$ data points each for the observational and two experimental domains, and simulate soft interventions on randomly selected targets. These interventions have linear coefficients sampled from $X \sim [0, 0.8]$, for which we randomly select the sign. In order to scale the effect of these interventions we multiply the coefficients for all interventions by the parameter $\text{IFactor}$, varying it from 0.1 to 100. Moreover, we randomly select $C_1$ and $Y$ from intervention and system variables respectively. We disallow direct effects of $C_1$ on $Y$, and enforce that no intervention can directly affect all variables simultaneously.

As expected, our method performs well when the target distribution is significantly different than the source distributions. Figure 5 shows different settings with different scales of intervention effects. In Figure 5a the intervention effects are all scaled by 0.1, resulting in very similar distributions in all domains. In this case, using our method does not offer any advantage with respect to the baseline and it actually performs worse. In the other cases, using our method starts to pay off in terms of prediction accuracy, and the difference increases with the scale of the interventions, as seen in Figure 5d.

In Figure 6, we vary the number of samples $N$ for each regime. The results improve with more samples, especially for our method, since the quality of the conditional independence test improves, but also for the baseline. In particular, as shown in Figure 6a, the accuracy is low for $N=100$ samples, but it improves substantially with $N=1000$ samples (Figure 5b).

![Figure 5. Additional results when varying the causal effect of all interventions (IFactor).](image1)

![Figure 6. Additional results when varying the sample size per regime (N).](image2)
Jointly Optimize Data Augmentation and Network Training:
Adversarial Data Augmentation in Human Pose Estimation

Xi Peng
Rutgers University
xip13@cs.rutgers.edu

Zhiqiang Tang
Rutgers University
zt53@cs.rutgers.edu

Fei Yang
Facebook
yangfei@fb.com

Rogerio S. Feris
IBM T.J. Watson Research Center
rsferis@us.ibm.com

Dimitris Metaxas
Rutgers University
dnm@cs.rutgers.edu

Abstract
Random data augmentation is a critical technique to avoid overfitting in training deep neural network models. However, data augmentation and network training are usually treated as two isolated processes, limiting the effectiveness of network training. Why not jointly optimize the two? We propose adversarial data augmentation to address this limitation. The main idea is to design an augmentation network (generator) that competes against a target network (discriminator) by generating “hard” augmentation operations online. The augmentation network explores the weaknesses of the target network, while the latter learns from “hard” augmentations to achieve better performance. We also design a reward/penalty strategy for effective joint training. We demonstrate our approach on the problem of human pose estimation and carry out a comprehensive experimental analysis, showing that our method can significantly improve state-of-the-art models without additional data efforts.

1. Introduction
Deep Neural Networks (DNNs) have achieved significant improvements in many computer vision tasks [20, 10, 18, 9]. A key ingredient for the success of state-of-the-art deep learning models is the availability of large amounts of training data. However, data collection and annotation are costly, and for many tasks, only a few training examples may be available. In addition, natural images usually follow a long-tail distribution [46, 33]. Effective training examples that lead to more robust classifiers may still be rare even if a large amount of data have been collected.

A common solution for this problem is to perform random data augmentation [21, 37]. Prior to being fed into the network, training images are heuristically jittered by predefined transformations (e.g., scaling, rotating, occluding) to increase variations. This strategy is simple, but data augmentation and network training are still treated as isolated processes, leading to the following issues.

First, the entire training set is usually applied the same random data augmentation strategy without considering the individual difference. This may produce many ineffective variations that are either too “hard” or too “easy” to help the network training [32, 39]. Second, random data augmentations can hardly match the dynamic training status since they are usually sampled from static distributions. Third, Gaussian distribution are widely used, which cannot address the long-tail issue since there would be a small chance to sample rare but useful augmentations.

A natural question then arises: can data augmentation and network training be jointly optimized to improve model performance? We propose to bridge the two by generating adversarial augmentations online. The generations are conditioned to both training images and the status of the target network.
network training be jointly optimized, so that effective augmentations can be generated online to improve the training.

In this work, we answer the above question by proposing a new approach that leverages adversarial learning for joint optimization of data augmentation and network training (see Figure 1). Specifically, we investigate the problem of human pose estimation, aiming to improve the network training with bounded datasets. Note that our approach can be generalized to other vision tasks, such as face alignment [25] and instance segmentation [23, 13].

Given an off-the-shelf pose estimation network, our goal is to obtain improved training from a bounded dataset. Specifically, we propose an augmentation network that acts as a generator. It aims to create “hard” augmentations that intend to make the pose network fail. The pose network, on the other hand, is modeled as a discriminator. It evaluates the quality of the generations, and more importantly, tries to learn from the “hard” augmentations. The main idea is to generate adversarial data augmentations online, conditioned to both training images and training status of the pose network. In other words, the augmentation network explores the weaknesses of the pose network which, at the same time, learns from adversarial augmentations for better performance.

Joint training of the two networks is a non-trivial task. Our experiments indicate that a straightforward design, such as directly generating adversarial pixels [12, 30] or deformations [39, 18], would yield problematic convergence behaviors (e.g. divergence and model collapse). Instead, the augmentation network is designed to generate adversarial distributions, from which augmentation operations (e.g. scaling, rotating, occluding) are sampled to create new data points. Besides, we propose a novel reward and penalty policy to address the issue of missing supervisions during the joint training. Moreover, instead of a raw image, the augmentation network is designed to take the byproduct, i.e. hierarchical features, of the pose network as the input. This can further improve the joint training efficiency using additional spatial constraints. To summarize, our key contributions are:

1. To the best of our knowledge, we are the first to investigate the joint optimization of data augmentation and network training in human pose estimation.
2. We propose an augmentation network to play a minimax game against the target network, by generating adversarial augmentations online.
3. We take advantage of the widely used U-net design and propose a reward and penalty policy for the efficient training of our networks.
4. Strong performance on public benchmarks, e.g. MPII and LSP, as well as intensive ablation studies, validate our method substantially in various aspects.

2. Related Work

We provide a brief overview of previous methods that are most relevant to ours in three categories.

Adversarial learning. Generative Adversarial Networks (GANs) [12, 45, 47] are designed as playing minimax games between generator and discriminator. Yu and Grauman [44] use GANs to synthesize image pairs to overcome the sparsity of supervision when learning to compare images. A-Fast-RCNN [39] uses GANs to generate deformations for object detection. Recent applications of GANs in human pose estimation include [6] and [7]. They both treat the pose estimation network as the generator and use a discriminator to provide additional supervision. However, in our design, the pose estimation network is treated as a discriminator, while the augmentation network is designed as a generator to create adversarial augmentations.

Hard example mining. It is wildly used in training SVM models for object detection [38, 41, 32]. The idea is to perform an alternative optimization of model training and data selection. Hard example mining focuses on how to select hard examples from the training set for effective training. It cannot create new data that do not exist in the training set. In contrast, we propose an augmentation network (generator) to actively generate adversarial data augmentations. This will create new data points that may not exist in the training set to improve the pose network (discriminator) training.

Human pose estimation. DeepPose [37] proposed to use deep neural networks for human pose estimation. Since then, deep learning based methods started to dominate this area (e.g. [4, 35, 15, 28, 22, 11, 17, 40, 3, 24]). For instance, Tompson et al. [36] used multiple branches of convolutional networks to fuse the features from an image pyramid. They applied Markov Random Field for post-processing. Chen et al. [5] also tried to combine neural networks with the graphical model inference to improve the pose estimation accuracy.

Recently, cascade models become popular for human pose estimation. They usually connect a series of deep neural networks in cascade to perform the estimation in a stage-by-stage manner. For example, Convolutional Pose Machines [40] brings obvious improvements by cascading multiple networks and adding intermediate supervisions. Better performance is achieved by the stacked hourglass network architecture [24], which also relies on multi-stage pose estimation. More recently, Chu et al. [8] added some layers into the stacked hourglass network for attention modeling. Yang et al. [42] also enhanced its performance by using pyramid residual modules. In this paper, instead of designing a new pose estimation network, we are more interested in how to jointly optimize data augmentation and network training. So, we can obtain improved training effect on any off-the-shelf deep neural network without looking for more data.

3. Adversarial Data Augmentation

Given a pre-defined pose network, e.g. the stacked hourglass pose estimator [24], our goal is to improve its training without looking for more data. Random data augmentation is widely used in deep neural network training. However, random data augmentations that are sampled from static distributions can hardly follow the dynamic training status, which may produce many ineffective variations that are either too “hard” or too “easy” to help the network training [32, 39]. Instead, we propose to leverage adversarial learning to optimize the data augmentation and the network training jointly. The main idea is to learn an augmentation network \( G(\theta_G) \) that generates “hard” augmentations that may increase the pose network loss. The pose network \( D(\theta_D) \), on the other hand, tries to learn from the adversarial augmentations and, at the same time, evaluates the quality of the generations. Please refer to Figure 2 for an overview of our approach.

Generation path. The augmentation network is designed as a generator. It outputs a set of distributions of augmentations operations. Mathematically, the augmentation network \( G \) outputs adversarial augmentation \( \tau(\cdot) \) that may increase \( D \)’s loss, compared with random augmentation \( \tau_r(\cdot) \), by maximizing the expectation:

\[
\min_{\tau \in \Omega} \mathbb{E}_{x \sim p(x)} \mathbb{E}_{\tau \sim \tau(\cdot)} \mathbb{E}_{y \sim y(x)} [L(D(\tau(x), y); z) - L(D(y(x), y); z)]
\]

where \( \Omega \) is the training image set and \( D \) is the random augmentation space. \( \tau(\cdot) \) is a predefined loss function and \( y(x) \) is the image annotation. We highlight that \( G(\theta_G) \) to specify the generation function of the input image \( x \) and the current status of the target network \( D \).

Discrimination path. The pose network is designed as a discriminator. It plays two roles: 1) \( D \) evaluates the generation quality as indicated in Equation (1); 2) \( D \) tries to learn from adversarial generations for better performance by minimizing the expectation:

\[
\min_{\tau_r \in \Omega} \mathbb{E}_{x \sim p(x)} \mathbb{E}_{\tau_r \sim \tau_r(\cdot)} \mathbb{E}_{y \sim y(x)} [L(D(\tau_r(x), y); z) - L(D(y(x), y); z)]
\]

The main idea is to learn an augmentation network \( G \) and a target network \( D \) to generate adversarial data augmentations online, conditioned to both input images and the training status of the pose network. Hard example mining focuses on how to select hard examples from the training set to improve the pose network [32, 39]. Instead, the pose estimation network is treated as a discriminator, while the augmentation network is designed as a generator to create adversarial augmentations.

Figure 2: Left: Overview of our approach. We propose an augmentation network to help the training of the pose network. The former creates hard augmentations; the latter learn from generations and produces reward/penalty for model update. Right: Illustration of the augmentation network. Instead of raw images, it takes hierarchical features of a U-net as inputs.
Algorithm 1: Training scheme of a mini batch

Output: $G, D$.

1. Randomly and equally divide $X$ into $X_1$, $X_2$ and $X_3$.
2. Train $D$ using $X_1$.
3. Train $D, G$ using $X_2$ with ASR following Alg. 2.
4. Train $D, G$ using $X_3$ with AHO following Alg. 2.

AHO pre-training. Similar to scaling and rotating, the augmentation network predicts an occluding distribution instead of an instance occluding mask. The first task is to ground the coarse truth of the occluding distribution. The idea is to assign values into a grid of $w \times h$ ($w = h = 4$). The value indicates the importance of the features at the corresponding cell. To achieve this, we vote a joint to one of the $w \times h$ cells according to its coordinates. By counting all joints from all images and normalizing the sum of cells to 1, we generate a heat map $P_o \in \mathbb{R}^{w \times h}$, which approximates the ground truth of the occluding distribution.

Given the prediction $P_o$, we propose a KL-divergence loss to pre-train the AHO task:

$$L_{AHO} = \sum_{i=1}^{w \times h} P_o(i) \log \frac{P_o(i)}{P_m(i)},$$

where $P_m \in \mathbb{R}^{w \times h}$ is the heat map predicted by the augmentation network. To generate the occluding mask, we sample one or two cells according to $P_m$, which are labeled as $0$ while the rest are labeled as $1$.

Discussion. Intuitively, there are three ways to apply hierarchical occluding: (1) a single mask scales up from the lowest to the highest resolutions, (2) a single mask scales down from the highest to the lowest resolutions, and (3) independent masks are generated at different resolutions. We exclusively use the first design in our approach since it would occlude more than needed due to the large receptive field in the second case, and the occluded information may be compensated at other resolutions in the third case.

4.3. Joint Training of Two Networks

Once ASR and AHO are pre-trained, we can jointly optimize the augmentation network and the pose network. As we mentioned in Sec. 3, this is a non-trivial task since the augmentation ground truth is missing. A naive approach could be repeating the pre-training process as described in Section 4.1 and Section 4.2 online. However, it would be extremely time-consuming since there are a large number of independent tasks.

Reward and penalty. Instead, we propose a reward and penalty policy to address this issue. The key idea is, the prediction of the augmentation network should be updated according to the current status of the target network, while its quality should be evaluated by comparing with a reference.

Algorithm 2: Training scheme of one image.

Output: $G, D$.

1. Forward $D$ to get bridge features $F$.
2. Forward $G$ with $F$ to get a distribution $P_r$.
3. Sample an adversarial augmentation $x$ from $P_r$.
4. Forward $D$ with $x$ to compute loss $L$.
5. Random augment $x$ to get $x'$.
6. Forward $D$ with $x'$ to compute loss $L'$.
7. Compare $L$ with $L'$ to update $G$ using (3) and (4).
8. Update $D$.

To this end, we sample a pair of augmentations for each image: 1) an adversarial augmentation $r$, and 2) a random augmentation $s$, as indicated in Equation (1). If the adversarial augmentation is harder than the random one, we reward the augmentation network for increasing the predictability of the sampled bin (ASR) or cell (AHO). Otherwise, we penalize it by decreasing the probability accordingly.

Mathematically, let $\tilde{P} \in \mathbb{R}^k$ denote the predicted distribution of the augmentation network. $P \in \mathbb{R}^k$ denotes the ground truth we are looking for. $k$ is the number of bins (ASR) or cells (AHO) and $i$ is the sampled one.

If the adversarial augmentation $\tau_i$ leads to higher pose network loss (more “difficult”) compared with the reference (a random augmentation $\alpha_i$), we update $P$ by rewarding:

$$P_i = P_i + \alpha_i \frac{1}{\tau_i}; P_i = \frac{\tau_i}{k-1} \alpha_i \forall j \neq i.$$

Similarly, if $\tau_i$ leads to lower pose network loss (less “difficult”) compared with $\alpha_i$, we update $P$ by penalizing:

$$P_i = P_i - \beta_i \frac{1}{\tau_i}; P_i = \frac{\tau_i}{k-1} \beta_i \forall j \neq i,$$

where $0 < \alpha_i, \beta_i \leq 1$ are hyperparameters that controls the amount of reward and penalty. The augmentation network keeps updating online, regardless of being rewarded or penalized, generating adversarial augmentations that intend to improve the pose network.

Discussion. The pose network can learn from the ordinary random augmentation to maintain its regular performance. More importantly, it can also learn from the adversarial augmentations to achieve better performance. The adversary augmentations may become too hard for the pose network if we apply ASR and AHO simultaneously. Thus, we alternately apply ASR and AHO on different images.
5. Experiments

In this section, we first show the visualization of network training states to verify the motivation of doing adversarial dynamic augmentation. Then we quantitatively evaluate the effectiveness of different components in the method and further compare with state-of-the-art approaches.

5.1. Experiment Settings

We use stacked hourglass (24) as the pose network. The augmentation network takes the top-down part of an hourglass and only uses one cell module in each resolution block. To evaluate the generalization capability of the proposed adversarial augmentation, we tested two types of modules: Residual module [14] and Dense block [16]. The dense block provides direct connections among different layers, which helps the gradient flow in backpropagation.

Network design. We test both residual hourglass and dense hourglass in our component evaluation experiments. For residual hourglass, each residual module has a bottleneck structure of BN-ReLU-Conv(3x3)-BN-ReLU-Conv(3x3)-BN-ReLU-Conv(1x1). The input/output dimension of each bottleneck is 256. The two × 1 convolutions are used to halve and double the feature dimensions.

For dense hourglass, each module is a bottleneck structure of BN-ReLU-Conv(1x1)-BN-ReLU-Conv(3x3)-BN-ReLU-Conv(1x1). The input/output dimension of each bottleneck is 256. The two × 1 convolutions are used to halve and double the feature dimensions.

Datasets. We evaluate the proposed adversarial human pose estimation on two benchmark datasets: MPII Human Pose [1] and Leeds Sports Pose (LSP) [19]. MPII is collected from YouTube videos with a broad range of human activities. It has 25K images and 40K annotated persons (29K for training and 11K for testing). Following [16], we sample 3K samples from the training set for validation. Each person has 16 labeled joints.

The LSP dataset contains images from many sports scenes. Its extended version has 11K training samples and 1K testing samples. Each person in LSP has 14 labeled joints. Since there are usually multiple people in one image, we crop around each person and resize it to 256 × 256. Typically, random scaling (0.75-1.25), rotating (−/+30 °) and flipping is used to augment the data.

Training. We use PyTorch for the implementation. RM-SProp [34] is used to optimize the networks. The adversarial training contains three stages. We first train hourglass for a few epochs with a learning rate 2.5 × 10−4. Then we freeze the hourglass model and use it to train the AHO and ASR networks with learning rate 2.5 × 10−5. Once they are pre-trained, we lower the learning rates of AHO and ASR networks to 5 × 10−5 and jointly train the three networks. The learning rate of the target network is decayed to 5 × 10−5 after the validation accuracy plateaus. In all experiments, the Percentage of Correct Keypoints (PCK) [43] is used to measure the pose estimation accuracy.

5.2. Visualization of the Training Status

In this experiment, we use a single residual hourglass. Each residual block contains 3 residual modules. We are interested in knowing how the pose network handles human images with different data augmentations: rotating, scaling and occluding. Since our method treats these three variations in a similar way, we take rotating as an example. More specifically, we visualize the loss distribution of hourglass on images with different rotations.

Random data augmentation. We train the pose network using random rotating sampled from a zero-centered Gaussian distribution as shown in the last row of Figure 5. We then test the trained pose network by applying the same rotating distribution on the testing data. We find that, at different training stages (training epochs), the target network loss always presents an inverted Gaussian-like distribution.

Adversarial data augmentation. In the beginning, the loss distribution of the pose network is similar to the case of random data augmentation. Since the pose network is pre-trained by the random data augmentation. However, the distribution becomes flatter as the training continues, which means the pose network could better handle the rotated images. The pose network learns from the adversarial data augmentation generated by the augmentation network.

Augmentation network training status. The status can be visualized by applying the generated rotating augmentation. Comparing the first two rows in Figure 5, we can find that the generated rotating distribution is similar to the loss distribution of the pose network. This means that the augmentation network could track the training status of the target network and generate effective data augmentations.

5.3. Component Evaluation

In this experiment, we verify the effectiveness of ASR and AHO in both residual and dense hourglasses. We use 3 residual bottleneck layers in each block of residual hourglass. In dense hourglass, we use 6 densely connected bottlenecks in one dense block.

Table 1 also shows that ASR improves the accuracy of all the keypoints on both residual and dense hourglasses, with average improvements of 0.5% and 0.5% respectively. This indicates that the generated adversarial scaling and rotating augmentations are effective in training the pose network.

Dense hourglass vs Residual hourglass. Table 1 shows that the dense hourglass has comparable performance in terms of pose estimation accuracy, but much more parameter efficient than the residual hourglass. ASR and AHO. Applying both ASR and AHO can further improve the accuracy by 0.4%, compared with applying either of them. Figure 6 shows that ASR and AHO can significantly improve the localization accuracy especially for joints that are usually more difficult to localize, such as ankle, knee and wrist.

Table 1: Comparison of random and adversarial data augmentation on the MPII validation set using PCKh@0.5. Consistent improvements on a range of normalized distances could be observed on both residual modules (left) and dense blocks (right).

5.4. Comparing with State-of-the-art Methods

Quantitative comparison. To compare with state-of-the-art methods, we apply the proposed adversarial data augmentation to train the hourglass networks (totally 8 stacked) [24]. The bridge features generated by the first hourglass network in the stack are input into the adversarial network. The same hierarchical occluding masks are applied to every hourglass network in the stack. Table 2 compares
The proposed adversarial data augmentation can improve the baseline stacked HGs(8) [24]. Indeed, our method improves the baseline [24] by 1.5%, which significantly outperforms existing methods on LSP dataset. Again, our method can improve performance. Table 3 compares PCK@0.2 accuracy of different methods on MPII dataset. Interestingly, the comparisons. We compare the random and adversarial data augmentation. Interestingly, the top-ankle, elbow, wrist), and left-right confusion.

Table 3: PCK@0.2 on the LSP dataset. Clear improvements from the adversarial data augmentation. Figur 7 shows qualitative analysis. In ECCV, 2016.


table 3: PCK@0.2 on the LSP dataset. Clear improvements from the adversarial data augmentation. Augmenting Models.

Figur 7: Comparisons of the same Stacked HG network trained using random data augmentation (top) and adversarial data augmentation (bottom). Note the improvement on challenging joints (e.g. ankle, elbow, wrist), and left-right confusion.

7. Acknowledgment

This work is partly supported by the Air Force Office of Scientific Research (AFOSR) under the Dynamic Data-Driven Application Systems program. We also thank to NSF Computer and Information Science and Engineering (CISE) for the support of our research.

References


Augmenting Models

∆-encoder: an effective sample synthesis method for few-shot object recognition

Eli Schwartz*1,2, Leonid Karlinsky*1,
Joseph Shlom1,2, Sivan Harary1, Mattias Marder1, Abhishek Kumar1,
Rogério Feris1, Raja Giryes2 and Alex M. Bronstein3

1IBM Research AI
2School of Electrical Engineering, Tel-Aviv University, Tel-Aviv, Israel
3Department of Computer Science, Technion, Haifa, Israel

Abstract

Learning to classify new categories based on just one or a few examples is a long-standing challenge in modern computer vision. In this work, we propose a simple yet effective method for few-shot (and one-shot) object recognition. Our approach is based on a modified auto-encoder, denoted ∆-encoder, that learns to synthesize new samples for an unseen category just by seeing few examples from it. The synthesized samples are then used to train a classifier. The proposed approach learns to both extract transferable intra-class deformations, or “deltas”, between same-class pairs of training examples, and to apply those deltas to the few provided examples of a novel class (unseen during training) in order to efficiently synthesize samples from that new class. The proposed method improves the state-of-the-art of one-shot object-recognition and performs comparably in the few-shot case.

* The authors have contributed equally to this work

Corresponding author: Leonid Karlinsky (leonidka@il.ibm.com)

Figure 1: Visualization of two-way one-shot classification trained on synthesized examples. Correctly classified images are framed in magenta (Golden retriever) and yellow (African wild dog). The only two images seen at training time and used for sample synthesis are framed in blue. Note the non-trivial relative arrangement of examples belonging to different classes handled successfully by our approach. The figure is plotted using t-SNE applied to VGG features. Best viewed in color.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.
1 Introduction

Following the great success of deep learning, the field of visual classification has made a significant leap forward, reaching – and in some cases, surpassing – human levels of performance (usually when expertise is required) [24, 37]. Starting from AlexNet [23], followed by VGG [38], Google Inception [42], ResNet [18], DenseNet [20] and NASNet [53], the field has made tremendous advances in classification performance on large-scale datasets, such as ImageNet [5], with thousands of examples per category. However, it is known that we humans learn naturally, building upon good learning new categories on the go, from seeing just a few or even a single example [24]. This is especially evident in early childhood, when a parent points and names an object and a child can immediately start finding more of its kind in the surroundings.

While the exact workings of the human brain are very far from being fully understood, one can conjecture that humans are likely to learn from analogies. That is, we identify in new objects elements of some latent semantic structure, present in other, already familiar categories, and use this structure to construct our internal classifier for the new category. Similarly, in the domain of computer vision we assume that we can use the plentiful set of examples (instances) of relevant classes (represented in some latent semantic space), in order to learn to sample from the distributions of the new classes, the ones for which we are given just one or a few examples.

Teaching a neural network to sample from distributions of new visual categories, based on just a few observed examples, is the essence of our proposed approach. First, the proposed approach learns to extract and later to sample (synthesize) transferable non-linear deformations between pairs of existing examples (training) classes. We refer to these examples as “deltas” in the feature space.

Second, it learns to apply those deltas to the few provided examples of novel categories, unseen during training, in order to efficiently synthesize new samples from these categories. Thus, in the few-shot scenario, we are able to synthesize enough samples of each new category to train a classifier in the standard supervised fashion.

Our proposed solution is a simple, yet effective (in the light of the obtained experimental results), method for learning to sample from the class distribution after being provided with one or a few examples of that class. It exhibits improved performance compared to the state-of-the-art methods for few-shot classification on a variety of standard few-shot classification benchmarks.

2 Related work

Few-shot learning by metric learning: a number of approaches [47, 39, 36] use a large corpus of instances of known categories to learn an embedding into a metric space where some simple (usually L2) metric is then used to classify instances of new categories via proximity to existing examples embedded in the same space. In [13], a metric learning method based on graph neural networks, that goes beyond the L2 metric, has been proposed. The metric-learning-based approaches are either posed as a general discriminative metric learning (DML) scheme [36], or optimized to operate in the few shot scenario [39, 47, 13]. These approaches show great promise, and in some cases are able to learn embedding spaces with quite meaningful semantics embedded in them. Yet, their performance is in many cases inferior to the meta-learning and generative (synthesis) approaches that will be discussed next.

Few-shot meta-learning (learning-to-learn): these approaches are trained on few-shot tasks instead of specific object instances, resulting in models that once trained can “learn” on new tasks with relatively few examples. In Matching Networks [43], a non-parametric k-NN classifier is meta-learned such that for each few-shot task the learned model generates an adaptive few-shot learning space for which the task can be approached. In solutions such as MAML [10], Meta-SGD [26], DEMLe+Meta-SGD [51], Meta-Learn LSTM [34] and Meta-Networks [31], the meta-learned classifiers are optimized to be easily fine-tuned on new few-shot tasks using standard training data.

Generative and augmentation-based few-shot methods: in these few-shot methods, either generative models are trained to synthesize new data based on few examples, or additional examples are obtained by some other form of transferring external data. These approaches can be categorized as follows: (1) semi-supervised approaches using additional unlabeled data [6, 11]; (2) fine tuning from pre-trained models [25, 45, 46]; (3) applying domain transfer by borrowing examples from relevant categories [27] or using semantic vocabularies [2, 12]; (4) rendering synthetic examples [26, 7, 40]; (5) augmenting the training examples [23]; (6) example synthesis using Generative Adversarial Networks (GANs) [52, 23, 14, 35, 29, 8, 20]; and (6) learning to use additional contextual information (e.g. attribute vector) per-instance for example synthesis [4]. It is noteworthy that all the augmentation and synthesis approaches can be used in combination with the metric learning or meta-learning schemes, and can always synthesize more data before using those approaches and thus (hopefully) improve their performance.

Several insightful papers have recently emerged dealing with sample synthesis. In [17] it is conjectured that the relative linear offset in feature space between a pair of same-class examples conveys information on a valid deformation, and can be applied to instances of other classes. In their approach, similar (in terms of this offset) pairs of examples from different categories are mined during training and then used to train a generator optimized for applying the same offset to other examples. In our technique, we do not restrict our “deltas” to be linear offsets, and in principle can have the encoder and the generator to learn more complex deformations than offsets in the feature space.

In [44], a generator sub-network is added to a classification network in order to synthesize additional examples on the fly in a way that helps training the classifier on small data. This generator receives the provided training examples accompanied by noise vectors (source of randomness). At the learning stage, the generator is optimized to perform random augmentation, jointly with the meta-learner parameters, via the classification loss. In contrast, in our strategy the generator is explicitly trained, via the reconstruction loss, to transfer deformations between examples and categories. Similar idea of learning to randomly augment class examples in a way that will improve classification performance is explored in [1] using GANs. In [35], a few-shot class density estimation is performed with an autoregressive model augmented with an attention mechanism, that serves as a kind of meta-learning. In their approach, density estimates from a large corpus of observed examples, is the essence of our proposed approach. First, the proposed approach learns to extract and later to sample (synthesize) transferable non-linear deformations between pairs of examples (training) classes. We refer to these examples as “deltas” in the feature space.

Second, it learns to apply those deltas to the few provided examples of novel categories, unseen during training, in order to efficiently synthesize new samples from these categories. Thus, in the few-shot scenario, we are able to synthesize enough samples of each new category to train a classifier in the standard supervised fashion. Our proposed solution is a simple, yet effective (in the light of the obtained experimental results), method for learning to sample from the class distribution after being provided with one or a few examples of that class. It exhibits improved performance compared to the state-of-the-art methods for few-shot classification on a variety of standard few-shot classification benchmarks.

3 The $\Delta$-encoder

We propose a method for few-shot classification by learning to synthesize samples of novel categories (unseen during training) when only a single or a few real examples are available. The generated samples are then used to train a classifier. Our proposed approach, dubbed as the $\Delta$-encoder, learns to sample from the category distribution, while being seeded by only one or a few examples from that distribution. Doing so, it belongs to the family of example synthesis methods. Yet, it does not assume the existence of additional unlabeled data, e.g., transferable pre-trained models (on an external dataset) or any directly related examples from other categories or domains, and it does not rely on additional semantic information per-instance.

The proposed solution is to train a network comprised of an encoder and a decoder. The encoder learns to extract transferable deformations between pairs of examples of the same category. The decoder, on the other hand, learns how to apply these deformations to other examples in order to learn to sample from new categories. For the ease of notation, assume we have given a single example $Y$ belonging to a certain category $C$, and our goal is to learn to sample additional examples $X$ belonging to the same category $C$. In other words, we would like to learn to sample from the class posterior $P(X|C, Y)$. Notice that the conditioning on $Y$ implies that we may not learn to sample from the whole class posterior, but rather from its certain subset of “modes” that can be obtained from $Y$ using the deformations we learned to extract. Our method is inspired by the one used for zero-shot classification in [3], where the decoder is provided side information about the class, in the form of human-annotated attributes.

Our generative model is a variant of an Auto-Encoder (AE). Standard AE learns to reconstruct a signal $X$ by minimizing $\|X - \hat{X}\|_2$, where $\hat{X} = D(Y, X)$ is the signal reconstructed by the AE, and $D$ and $E$ are the encoder and decoder sub-networks, respectively. A common assumption for an AE is that the intermediate bottleneck representation $E(X)$, can be of much lower dimension than $X$. This is driven by assuming the ability to extract the “semantic essence” of $X$ – a minimal set of identifying features of $X$ necessary for the reconstruction. The simple idea of this work is to change the meaning of $E(X)$ from representing the “essence” of $X$, to representing the delta, or “additional information” needed to reconstruct $X$ from $Y$ (an observed example from the same category).

To this end, we propose the training architecture depicted in Figure 2a. The encoder gets as an input both the signal $X$ and the “anchor” example $Y$ and learns to compute the representation of the additional information $E(\Delta Y, X)$ needed by the decoder $D$ in order to reconstruct the $X$
We have evaluated the few-shot classification performance of the proposed method on multiple datasets, which are the benchmarks of choice for the majority of few-shot learning literature; namely:

Table 1: 1-shot/5-shot 5-way accuracy results

<table>
<thead>
<tr>
<th>Method</th>
<th>miniImageNet</th>
<th>CIFAR100</th>
<th>Caltech-256</th>
<th>CUB</th>
<th>CUB</th>
<th>CUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest neighbor (baseline)</td>
<td>44.1 / 55.1</td>
<td>56.1 / 68.3</td>
<td>51.3 / 67.5</td>
<td>52.4 / 66.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACO [19]</td>
<td>41.1 / 58.3</td>
<td>-</td>
<td>-</td>
<td>60.8 / 75.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meta-Learner LSTM [34]</td>
<td>43.4 / 60.6</td>
<td>-</td>
<td>-</td>
<td>40.4 / 49.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Matching Nets [43]</td>
<td>46.6 / 60.0</td>
<td>50.5 / 60.3</td>
<td>48.1 / 57.5</td>
<td>49.3 / 59.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Matching Nets [10]</td>
<td>45.7 / 63.1</td>
<td>40.3 / 58.3</td>
<td>46.3 / 54.6</td>
<td>38.4 / 59.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prototypical Networks [39]</td>
<td>49.4 / 68.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Siamese [28]</td>
<td>55.2 / 69.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELATION NET [41]</td>
<td>57.0 / 71.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DML-Meta-SCD [51]</td>
<td>58.5 / 73.5</td>
<td>61.6 / 77.9</td>
<td>62.2 / 79.5</td>
<td>66.9 / 77.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dual TSN [41]</td>
<td>58.1 / 76.5</td>
<td>63.4 / 76.4</td>
<td>63.8 / 80.5</td>
<td>69.8 / 84.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>∆-encoder*</td>
<td>59.9 / 69.7</td>
<td>66.7 / 79.8</td>
<td>73.2 / 83.6</td>
<td>69.8 / 82.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Also trained on a large external dataset</td>
<td>Using label embedding trained on large corpus</td>
<td>Using human-annotated class attributes</td>
<td>Using ResNet features</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Also trained on a large external dataset

\[ \Delta \text{-encoder} \]

Figure 2: Proposed ∆-encoder architecture. (a) Training phase: \( X \) and \( Y \) are a random pair of samples from the same seen class; the ∆-encoder learns to reconstruct \( X \). (b) Sample synthesis phase: \( X \) and \( Y \) are a random pair of samples from a random seen class, and \( Y' \) is a single example from a novel unseen class; the ∆-encoder generates a new sample \( \hat{X} \) from the new class.

4.1 Standard benchmarks

For miniImageNet, CIFAR100, CUB and Caltech-256 datasets, we evaluate our approach using a backbone network (for computing the feature vectors) trained from scratch on a subset of categories of each dataset. For few-shot testing, we use the remaining unseen categories. The proposed synthesis network is trained on the same set of categories as the backbone network. The experimental protocol used here is the same as in all compared methods.

The performance achieved by our approach is summarized in Table 1; it competes favorably to the state-of-the-art of few-shot classification on these datasets. The performance of competing methods is taken from [4]. We remark in the table whenever a method uses some form of additional external data, be it training on an external large-scale dataset, using word embedding applied to the category name, or using human-annotated class attributes.

All models are trained with Adam optimizer with the learning rate set to \( 10^{-4} \). Dropout with 50% rate is applied to all layers. In all experiments 1024 samples are synthesized for each unseen class. The ∆-encoder training takes about 20 seconds running on an Nvidia Tesla K40m GPU (48k training samples, batch size 128). The data generation phase takes around 0.1 seconds per 1024 samples. The code is available here.

4 Results

We evaluated the few-shot classification performance of the proposed method on multiple datasets, which are the benchmarks of choice for the majority of few-shot learning literature; namely:
4.2 Additional experiments using a shared pre-trained feature extracting model

For fair comparison, in the experiments described above in Section 4.1 we only trained our feature extractor backbone on the subset of training categories of the target dataset (same as in other works). However, it is nonetheless interesting to see how our proposed method performs in a realistic setting of having a single pre-trained feature extractor backbone trained on a large set of external data. To this end, we have conducted experiments on four public datasets (APY, AWA2, CUB and SUN), where we used features obtained from a VGG16 backbone pre-trained on ImageNet. The unseen test categories were verified to be disjoint from the ImageNet categories in [49] that dealt with dataset bias in zero-shot experiments. The results of our experiments as well as comparisons to some baselines are summarized in Table 2. The experiments in this section illustrate that the proposed method can strongly benefit from better features trained on more data. For CUB with “stronger” ImageNet features (last column in Table 2) we achieved more than 10% improvement over training only using a subset of CUB categories (last column in Table 1).

Table 2: 1-shot/5-shot 5-way accuracy with ImageNet model features (trained on disjoint categories)

<table>
<thead>
<tr>
<th>Method</th>
<th>AWA2</th>
<th>APY</th>
<th>SUN</th>
<th>CUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest neighbor (baseline)</td>
<td>65.9</td>
<td>57.9</td>
<td>76.4</td>
<td>72.7</td>
</tr>
<tr>
<td>Prototypical Networks</td>
<td>80.8</td>
<td>95.3</td>
<td>69.9</td>
<td>90.1</td>
</tr>
<tr>
<td>Δ-encoder</td>
<td>90.5</td>
<td>96.4</td>
<td>82.5</td>
<td>93.4</td>
</tr>
</tbody>
</table>

4.3 Ablation study: evaluating different design choices

In this section we review and evaluate different design choices used in the architecture and the approach proposed in this paper. For this ablation study we use the performance estimates for the AWA, APY, SUN and CUB datasets to compare the different choices. In [3] the authors suggested the usage of Denoisng-Autoencoder (DAE) for zero-shot learning (Fig. 3a). The noise is implemented as 20% dropout on the input. In training time, the DAE learns to reconstruct \(X \) from it’s noisy version, where the decoder uses the class attributes to perform the reconstruction. At test time, the decoder is used to synthesize examples from a novel class using its attributes vector and a random noise vector \(Z \). Average accuracy for the zero-shot task is 64.4% (first row in Table 4). As a first step towards one-shot learning, we have tested the same architecture but using another sample from the same class instead of the attributes vector (Figure 3b). The intuition behind this is that the decoder will learn to reconstruct the class instances by editing another instances from the same class instead of relying on the class attributes. This already yields a significant improvement to the average accuracy bringing it to 81.1%, hinting that even a single class instance conveys more information than the human chosen attributes for the class in these datasets.

Next, we replaced the random sampling of \(Z \) with a non-parametric density estimate of it obtained from the training set. Instead of sampling entries of \(Z \sim N(0, 1) \), we randomly sample an instance \(X^* \) belonging to a randomly chosen training class and run it through the encoder to produce \(Z = E(X^*) \). This variant assumes that the distribution of \(0 \) belonging to a randomly chosen training class and run it through the encoder to produce \(Z = E(X^*) \). This variant assumes that the distribution of \(0 \) and \(1 \) are different. In this case, we observed a slight improvement of 0.5% due to this change. We also tested a variant where no noise is injected to the input, i.e. replacing Denoising-Autoencoder with Autoencoder. Since we did not observe a change in performance we chose the Autoencoder for being simpler of the two. Finally, to get to our final architecture as described in Section 3 we add \(Y \) as input to the encoder. This improved the performance by 2.4%.

**Linear offset delta** To evaluate the effect of the learned non-linear \(Δ\)-encoder we also experimented with replacing it with a linear “delta” in the embedding space. In this experiment we set \(Z = E(\Delta X) \) and \(X = E(\Delta Y) = Y + Z \). That means we sample linear shifts from same-class pairs in the training set and use them to augment the single example of a new class \(Y \) that we have. For this experiment we got ~10 points lower accuracy compared to \(Δ\)-encoder, showing the importance of the learned non-linear “delta” encoding.

Table 4: Evaluating different design choices. All the numbers are one-shot accuracy in %.

<table>
<thead>
<tr>
<th>Method</th>
<th>AWA2</th>
<th>APY</th>
<th>SUN</th>
<th>CUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero-shot ((Y ) is attribute vector) [3] (Fig. 3a)</td>
<td>66.4</td>
<td>62.0</td>
<td>82.5</td>
<td>42.8</td>
</tr>
<tr>
<td>Zero-shot ((Y ) is attributes vector) [3] (Fig. 3b)</td>
<td>81.3</td>
<td>72.1</td>
<td>73.5</td>
<td>73.9</td>
</tr>
<tr>
<td>Zero-shot attribute vector (s ) sampled from training set, not random (Fig. 3c)</td>
<td>85.4</td>
<td>78.1</td>
<td>81.1</td>
<td>81.1</td>
</tr>
<tr>
<td>Transferring linear offsets in the embedding space</td>
<td>76.6</td>
<td>73.8</td>
<td>81.2</td>
<td>68.2</td>
</tr>
<tr>
<td>Autoencoder instead of denoising-autoencoder (Fig. 3d)</td>
<td>88.2</td>
<td>80.9</td>
<td>79.5</td>
<td>78.1</td>
</tr>
</tbody>
</table>

**Figure 3:** Different design choices tried. Classification accuracy for each architecture is presented in Table 4. The final chosen architecture is depicted in Fig. 2.

4.4 Are we synthesizing non-trivial samples?

We have observed a significant few-shot performance boost when using the proposed method for sample synthesis when compared to the baseline of using just the few provided examples. But are we learning to generate any significant new information on the class manifold in the feature space? Or are we simply augmenting the provided examples slightly? We chose two ways to approach this question. First, evaluating the performance as more samples are synthesized. Second, visualizing the synthesized samples in the feature space.

Figure 4 presents the accuracy as a function of number of generated samples in both 1-shot and 5-shot scenarios when evaluated on the miniImageNet dataset. As can be seen the performance improves with the number of samples synthesized, converging after 5/12 – 1/24 samples. For this reason we use 1/24 synthesized samples in all of our experiments. It is interesting to note that at convergence,
not only using the 5-shot synthesized samples is significantly better than the baseline of using just the five provided real examples, but also using the samples synthesized from just one real example is better than using five real examples (without variation). This may suggest that the proposed synthesis approach does learn something non-trivial surpassing the addition of four real examples.

To visualize the synthesized samples we plot them for the case of 12 classes unseen during training (Figure 5a). The samples were synthesized from a single real example for each class (1-shot mode) and plotted in 2d using t-SNE. As can be seen from the figure, the synthesized samples reveal a non-trivial density structure around the seed examples. Moreover, the seed examples are not the centers of the synthesized populations (we verified that same is true before applying t-SNE) as would be expected for naive augmentation by random perturbation. Hence, the classifier learned from the synthesized examples significantly differs from the nearest neighbors baseline classifier that is using the seed examples alone (improving its performance by 20 – 30% in tables 1 & 2). In addition, Figure 5b shows visualizations for some of the synthesized feature vectors obtained from a given seed example. The images displayed are the nearest neighbors of the synthesized ones in the feature space.

Figure 5: a. Generated samples for 12-way one-shot. The red crosses mark the original 12 single-samples. The generated points are colored according to their class. b. Synthesized samples visualization. The single image seen at training is framed in blue. All other images represent the synthesized samples visualized using their nearest “real image” neighbors in the feature space. The two-dimensional embedding was produced by t-SNE. Best viewed in color.

5 Summary and Future work

In this work, we proposed a novel auto-encoder like architecture, the $\Delta$-encoder. This model learns to generate novel samples from a distribution of a class unseen during training using as little as one example from that class. The $\Delta$-encoder was shown to achieve state-of-the-art results in the task of few-shot classification. We believe that this new tool can be utilized in a variety of more general settings challenged by the scarceness of labeled examples, e.g., in semi-supervised and active learning. In the latter case, new candidate examples for labeling can be selected by first generating new samples using the $\Delta$-encoder, and then picking the data points that are farthest from the generated samples. Additional, more technical, research directions include iterative sampling from the generated distribution by feeding the generated samples as reference examples, and conditioning the sampling of the “delays” on the anchor example for better controlling the set of transformations suitable for transfer. We leave these interesting directions for future research.

Acknowledgment: Part of this research was partially supported by the ERC-StG SPADE grant.

References

Section 1
Augmenting Models


Routing Networks: Adaptive Selection of Non-linear Functions for Multi-Task Learning

Clemens Rosenbaum
College of Information and Computer Sciences
University of Massachusetts Amherst
140 Governors Dr., Amherst, MA 01003
cgr@cs.umass.edu

Tim Klinger & Matthew Riemer
IBM Research AI
1101 Kitchawan Rd, Yorktown Heights, NY 10598
{tklinger,mdriemer}@us.ibm.com

ABSTRACT

Multi-task learning (MTL) with neural networks leverages commonalities in tasks to improve performance, but often suffers from task interference which reduces the benefits of transfer. To address this issue we introduce the routing network paradigm, a novel neural network and training algorithm. A routing network is a kind of self-organizing neural network consisting of two components: a router and a set of one or more function blocks. A function block may be any neural network – for example a fully-connected or a convolutional layer. Given an input the router makes a routing decision, choosing a function block to apply and passing the output back to the router recursively, terminating when a fixed recursion depth is reached. In this way the routing network dynamically composes different function blocks for each input. We employ a collaborative multi-agent reinforcement learning (MARL) approach to jointly train the router and function blocks. We evaluate our model against cross-stitch networks and shared-layer baselines on multi-task settings of the MNIST, mini-imagenet, and CIFAR-100 datasets. Our experiments demonstrate a significant improvement in accuracy, with sharper convergence. In addition, routing networks have nearly constant per-task training cost while cross-stitch networks scale linearly with the number of tasks. On CIFAR-100 (20 tasks) we obtain cross-stitch performance levels with an 85% reduction in training time.

1 Introduction

Multi-task learning (MTL) is a paradigm in which multiple tasks must be learned simultaneously. Tasks are typically separate prediction problems, each with their own data distribution. In an early formulation of the problem, (Caruana, 1997) describes the goal of MTL as improving generalization performance by “leveraging the domain-specific information contained in the training signals of related tasks.” This means a model must leverage commonalities in the tasks (positive transfer) while minimizing interference (negative transfer). In this paper we propose a new architecture for MTL problems called a routing network, which consists of two trainable components: a router and a set of function blocks. Given an input, the router selects a function block from the set, applies it to the input, and passes the result back to the router, recursively up to a fixed recursion depth. If the router needs fewer iterations then it can decide to take a PASS action which leaves the current state unchanged. Intuitively, the architecture allows the network to dynamically self-organize in response to the input, sharing function blocks for different tasks when positive transfer is possible, and using separate blocks to prevent negative transfer.

The architecture is very general allowing many possible router implementations. For example, the router can condition its decision on both the current activation and a task label or just one or the other. It can also condition on the depth (number of router invocations), filtering the function module choices to allow layering. In addition, it can condition its decision for one instance on what was historically decided for other instances, to encourage re-use of existing functions for improved compression. The function blocks may be simple fully-connected neural network layers or whole...
Routing networks as long as the dimensionality of each function block allows composition with the previous function block choice. They needn’t even be the same type of layer. Any neural network or part of a network can be “routed” by adding its layers to the set of function blocks, making the architecture applicable to a wide range of problems. Because the routers make the network learn a sequence of hard decisions, which are not differentiable, we use reinforcement learning (RL) to train them. We discuss the training algorithm in Section 3.1, but one way we have modeled this as an RL problem is to create a separate RL agent for each task (assuming task labels are available in the dataset). Each such task agent learns its own policy for routing instances of that task through the function blocks.

To evaluate we have created a “routed” version of the convnet used in (Ravi & Larocheille, 2017) and use three image classification datasets adapted for MTL learning: a multi-task MNIST dataset that we created, a Mini-imagenet data split as introduced in (Vinyals et al., 2016), and CIFAR-100 (Krizhevsky, 2009), where each of the 20 label superclasses is treated as different tasks. We conduct extensive experiments comparing against cross-stitch networks (Misra et al., 2016) and the popular strategy of joint training with layer sharing as described in (Caruana, 1997). Our results indicate a significant improvement in accuracy over these strong baselines with a speedup in convergence and often orders of magnitude improvement in training time over cross-stitch networks.

2 RELATED WORK

Work on multi-task deep learning (Caruana, 1997) traditionally includes significant hand design of neural network architectures, attempting to find the right mix of task-specific and shared parameters. For example, many architectures share low-level features like those learned in shallow layers of deep convolutional networks or word embeddings across tasks and add task-specific architectures in later layers. By contrast, in routing networks, we learn a fully dynamic, compositional model which can adjust its structure differently for each task.

Routing networks share a common goal with techniques for automated selective transfer learning using attention (Rajendran et al., 2017) and learning gating mechanisms between representations (Stollenga et al., 2014), (Misra et al., 2016), (Ruder et al., 2017). In the latter two papers, experiments are performed on just 2 tasks at a time. We consider up to 20 tasks in our experiments and compare directly to (Misra et al., 2016).

Our work is also related to mixtures of experts architectures (Jacobs et al., 1991), (Jordan & Jacobs, 1994) as well as their modern attention based (Rieimer et al., 2016) and sparse (Shazeer et al., 2017) variants. A routing network in a typical mixture of experts model learns to “route” an input to an appropriate expert which then processes it and produces an output representation vector. A special PASS action (see Appendix Section 7.2 for details) just skips to the next iteration. Some experiments don’t require a task label and in that case we just pass a dummy value. For simplicity we assume the algorithm has access to the router function and don’t include them explicitly in the input. This process is illustrated in Figure 1. The input to the routing network is an instance to be classified $(x, t) \in \mathbb{R}^d$ is a representative vector of the input, and $t$ is an integer task identifier in $[1, \ldots, n]$ and a depth $\epsilon(\cdot)$, the depth of the recursion, and selects from among a set of function block choices available at depth 1, $f_1(x)$, $f_2(x)$, $f_3(x)$, picking $f_3(x)$ which is indicated with a dashed line. $f_3(x)$ is applied to the input $(x, t)$ to produce an output activation. The router again chooses a function block from those available at depth 2 (if the function blocks are of different dimensions then the router is constrained to select dimensionally matched blocks to apply and so on. Finally the router chooses a function block from the last (classification) layer function block set and produces the classification.

Algorithm 1 gives the routing procedure in detail. The algorithm takes as input a vector $v$, task label $t$ and maximum recursion depth $n$. It iterates $n$ times choosing a function block on each iteration and applying it to produce an output representation vector. A special PASS action (see Appendix Section 7.2 for details) just skips to the next iteration. Some experiments don’t require a task label and in that case we just pass a dummy value. For simplicity we assume the algorithm has access to the router function and don’t include them explicitly in the input. This process is illustrated in Figure 1. The input to the routing network is an instance to be classified $(x, t) \in \mathbb{R}^d$ is a representative vector of the input, and $t$ is an integer task identifier in $[1, \ldots, n]$, and a depth $\epsilon(\cdot)$, the depth of the recursion, and selects from among a set of function block choices available at depth 1, $f_1(x)$, $f_2(x)$, $f_3(x)$, picking $f_3(x)$ which is indicated with a dashed line. $f_3(x)$ is applied to the input $(x, t)$ to produce an output activation. The router again chooses a function block from those available at depth 2 (if the function blocks are of different dimensions then the router is constrained to select dimensionally matched blocks to apply and so on. Finally the router chooses a function block from the last (classification) layer function block set and produces the classification.

Algorithm 1: Routing Algorithm

```python
1: input: v, t, n
2: for \epsilon = 1, \ldots, n do
3:     a = \text{router}(x, t, \epsilon)
4:     if a \neq \text{PASS} then
5:         x = \text{function_block}_{a}(x)
6:     \end{if}
7: end for
8: return v
```

need for human intervention to manually choose which parameters will be shared and which will be kept task-specific.

Also related to our work is the literature on minimizing computation cost for single-task problems by conditional routing. These include decisions trained with REINFORCE (Denoyer & Gallinari, 2014), (Bengio et al., 2013), (Hamrick et al., 2017), Q-learning (Lin & Dong, 2017), and adaptive methods (McGill & Peirina, 2017). Our approach differs however in the introduction of several novel elements. Specifically, our work explores the multi-task learning setting, it uses a multi-agent reinforcement learning training algorithm, and it is structured as a recursive decision process.

There is a large body of related work which focuses on continual learning, in which tasks are presented to the network one at a time, potentially over a long period of time. One interesting recent paper in this setting, which also uses the notion of routes ("paths"), but uses evolutionary algorithms instead of RL is (Fernando et al., 2017). While a routing network is a novel artificial neural network formulation, the high-level idea of task-specific "routing" as a cognitive function is well founded in biological studies and theories of the human brain (Ginney et al., 2001), (Buschman & Miller, 2010), (Stocco et al., 2010).
If the routing network is run for \( d \) invocations then we say it has depth \( d \). For \( N \) function blocks a routing network run to a depth \( d \) can select from \( N^d \) distinct trainable functions (the paths in the network). Any neural network can be represented as a routing network by adding copies of its layers as routing network function blocks. We can group the function blocks for each network layer and constrain the router to pick from layer 0 function blocks at depth 0, layer 1 blocks at depth 1, and so on. If the number of function blocks differs from layer to layer in the original network, then the router may accommodate this by, for example, maintaining a separate decision function for each depth.

### 3.1 ROUTER TRAINING USING RL

We can view routing as an RL problem in the following way. The states of the MDP are the triples \((v, t, i)\) where \( v \in \mathbb{R}^v \) is a representation vector (initially the input), \( t \) is an integer task label for \( v \), and \( i \) is the depth (initially 1). The actions are function block choices (and \( \text{PASS} \)) in \([1, \ldots, k, \text{PASS}]\) for \( k \), the number of function blocks. Given a state \( s = (v, t, i) \), the router makes a decision about which action to take. For the non-PASS actions, the state is then updated \( s' = (v', t, i + 1) \) and the process continues. The \( \text{PASS} \) action produces the same representation vector again but increments the depth, so \( s' = (v, t, i + 1) \). We train the router policy using a variety of RL algorithms and settings which we will describe in detail in the next section.

Regardless of the RL algorithm applied, the router and function blocks are trained jointly. For each instance we route the instance through the network to produce a prediction \( \hat{y} \). Along the way we record the action \( a_i \) and the action \( a_t \) taken as well as an immediate reward \( r_i \) for action \( a_i \). When the last function block is chosen, we record a final reward which depends on the prediction \( \hat{y} \) and the true label \( y \).

![Algorithm 2: Router-Trainer. Training of a Routing Network.](image)

We can view routing as an RL problem in the following way. The states of the MDP are the triples \((v, t, i)\) where \( v \in \mathbb{R}^v \) is a representation vector (initially the input), \( t \) is an integer task label for \( v \), and \( i \) is the depth (initially 1). The actions are function block choices (and \( \text{PASS} \)) in \([1, \ldots, k, \text{PASS}]\) for \( k \), the number of function blocks. Given a state \( s = (v, t, i) \), the router makes a decision about which action to take. For the non-PASS actions, the state is then updated \( s' = (v', t, i + 1) \) and the process continues. The \( \text{PASS} \) action produces the same representation vector again but increments the depth, so \( s' = (v, t, i + 1) \). We train the router policy using a variety of RL algorithms and settings which we will describe in detail in the next section.

Regardless of the RL algorithm applied, the router and function blocks are trained jointly. For each instance we route the instance through the network to produce a prediction \( \hat{y} \). Along the way we record the action \( a_i \) and the action \( a_t \) taken as well as an immediate reward \( r_i \) for action \( a_i \). When the last function block is chosen, we record a final reward which depends on the prediction \( \hat{y} \) and the true label \( y \).

![Figure 2: Training (backward) Example](image)

We train the selected function blocks using SGD/Backprop. In the example of Figure 1 this means computing gradients for \( f_{\alpha_0}, f_{\alpha_1} \), and \( f_{\alpha_2} \). We then use the computed trace to train the router using an RL algorithm. The high-level procedure is summarized in Algorithm 2 and illustrated in Figure 2. To keep the presentation uncluttered we assume the RL training algorithm has access to the router function block approximation function, loss function, and any specific hyper-parameters such as discount rate needed for the training and don’t include them explicitly in the input.

### 3.1.1 REWARD DESIGN

A routing network uses two kinds of rewards: immediate action rewards \( r_i \) given in response to an action \( a_i \), and a final reward \( r_{\text{final}} \), given at the end of the routing. The final reward is a function of the network’s performance. For the classification problems focused on in this paper, we set it to \( +1 \) if the prediction was correct (\( y = \hat{y} \)), and \(-1 \) otherwise. For other domains, such as regression domains, the negative loss \( -\langle L(y, \hat{y}) \rangle \) could be used.

We experimented with an immediate reward that encourages the router to use fewer function blocks when possible. Since the number of function blocks per-layer needed to maximize performance is not known ahead of time (we just take it to be the same as the number of tasks), we wanted to see whether we could achieve comparable accuracy while reducing the number of function blocks (or blocks of blocks) chosen by the router, allowing us to reduce the size of the network after training. We experimented with two such rewards, multiplied by a hyper-parameter \( p \in [0, 1] \): the average number of times that block was chosen by the router historically and the average historical probability of the router choosing that block. We found no significant difference between the two approaches and use the average probability in our experiments. We evaluated the effect of \( p \) on final performance and report the results in Figure 12 in the appendix. We see there that generally \( p \approx 0.0 \) (no collaboration reward) or a small value works best and that there is relatively little sensitivity to the choice in this range.

### 3.1.2 RL ALGORITHMS

We experiment with storing the policy both as a table and in form of an approximator. The tabular representation has the invocation depth as its row dimension and the function block as its column dimension with the entries containing the probability of choosing a given function block at a given depth. The approximator representation can consist of either one MLP that is passed the depth (represented in 1-hot), or a vector of \( d \) MLPs, one for each decision/depth.

Both the Q-Learning and Policy Gradient algorithms are applicable with tabular and approximation function policy representations. We use REINFORCE (Williams, 1992) to train both the approximation function and tabular representations. For Q-Learning the table stores the \( Q \)-values in the entries. We use vanilla Q-Learning (Watkins, 1989) to train tabular representation and train the approximators to minimize the \( L_2 \) norm of the temporal difference error.

Implementing the router decision policy using multiple agents turns the routing problem into a stochastic game, which is a multi-agent extension of an MDP. In stochastic games multiple agents interact in the environment and the expected return for any given policy may change without any action on that agent’s part. In this view incompatible agents need to compete for blocks to train, since negative transfer will make collaboration unattractive, while compatible agents can gain by
Augmenting Models

Published as a conference paper at ICLR 2018

CIFAR-100 has coarse and fine labels for its instances. We follow existing work (Krizhevsky, 2009) creating one task for each of the 20 coarse labels and include 500 instances for each of the corresponding fine labels. There are 20 tasks with a total of 2.5k instances per task; 2.5k for training and 500 for testing. All results are reported on the test set and are averaged over 3 runs. The data are summarized in Table 1.

Each of these datasets has interesting characteristics which challenge the learning in different ways. CIFAR-MTL is a “natural” dataset where tasks correspond to human categories. MIN-MTL is randomly generated so will have less task coherence. This makes positive transfer more difficult to achieve and negative transfer more of a problem. And MNIST-MTL, while simple, has the difficult property that the same instance can appear with different labels in different tasks, causing interference. For example, in the “0 vs other digits” task, “0” appears with a positive label but in the “1 vs other digits” task it appears with a negative label.

Our experiments are conducted on a convent architecture (SimpleConvNet) which appeared recently in (Ravi & Larochelle, 2017). This model has 4 convolutional layers, each consisting of a 3x3 convolution and 32 filters, followed by batch normalization and a ReLU. The convolutional layers are followed by 3 fully connected layers, with 128 hidden units each. Our routed version of the network routes the 3 fully connected layers and for each router layer we supply one randomly initialized function block per task in the dataset. When we use neural net approximators for the router agents they are always 2 layer MLPs with a hidden dimension of 64. A state (s, t, i) is encoded for input to the approximator by concatenating s with a 1-hot representation of t (if used). That is, encoding(s) = (s, one_hot(t)).

We also tried several variations to find the best learning rate and ρ value for each algorithm on each dataset. We use ρ = 0.0 (no collaboration reward) for CIFAR-MTL and MIN-MTL and ρ = 0.3 for MIN-MTL. The learning rate is initialized to 10−5 and annealed by dividing by 10 every 20 epochs. We tried both regular SGD as well as Adam (Kingma & Ba (2014), but choose SGD as it resulted in marginally better performance. The SimpleConvNet has batch normalization layers but we use no dropout. For one experiment, we dedicate a special “PASS” action to allow the agents to skip layers during training which leaves the current state unchanged (routing-all-fc recurrent+PASS). A detailed description of the PASS action is provided in the Appendix in Section 7.2.

All data are presented in Table 2 in the Appendix. In the first experiment, shown in Figure 4, we compare different RL training algorithms on CIFAR-MTL. We compare five algorithms: MARL-WPL, a single agent REINFORCE learner with a separate approximation function per layer; an agent-per-task REINFORCE learner which maintains a separate approximation function for each layer; an agent-per-task Q learner with a separate approximator; an agent-per-task Q learner with a separate table for each layer. The best performer is the WPL algorithm which outperforms the nearest competitor, tabular Q-Learning by about 4%. We can say that if the WPL algorithm works better than a similar uninformed version, it has trouble learning; (2) having multiple agents works better than having a single agent; and (3) the tabular versions, which just use the task and depth to make their predictions, work better here than the approximation versions, which all use the representation vector in addition predict the next action. The next experiment compares the best performing algorithm WPL against other routing approaches, including the already introduced REINFORCE: single agent (for which WPL is not applicable). All of these algorithms route the full-connected layers of the SimpleConvNet using the layering approach we discussed earlier. To make the next comparison clear we rename MARL-WPL to routing-all-6; in Figure 5 to reflect the fact that it routes all the fully connected layers of the SimpleConvNet, and REINFORCE: single agent to routing-all-6: single agent. We compare against several other approaches. One approach, routing-all-6: recurrent+PASS, has the same setup as routing-all-6: it does not constrain the router to pick only from layer 0 function blocks at depth 0, etc. It is allowed to choose any function block from two of the layers (since the first two routed layers

Algorithm 3: Weighted Policy Learner

\begin{align*}
&\text{input: } T = (S, A, R, f) \text{ a treatment}, \\
&\text{output: } \{\Delta(a)\}_{a \in A} \text{ the updates to policy}\end{align*}

\begin{align*}
&\text{1. Compute the return: } R_t = R_{t-1} + \gamma \cdot \rho \cdot \pi(s_t) \cdot (s_{t+1} - s_t) \\
&\text{2. Update the average return: } \hat{R}_t = \frac{1}{n} \sum_{i=1}^{n} R_{t+i} \\
&\text{3. Compute the return: } \tilde{R}_t = \frac{1}{n} \sum_{i=1}^{n} \rho \cdot \pi(s_t) \cdot (s_{t+1} - s_t) \\
&\text{4. Compute the loss: } L = \sum_{t=0}^{T-1} \tilde{R}_t \\
&\text{5. Compute the advantage: } A_t = \tilde{R}_t - R_t \\
&\text{6. Update the average advantage: } \Delta(a) = \frac{1}{n} \sum_{i=1}^{n} \rho \cdot \pi(s_t) \cdot (s_{t+1} - s_t) \\
&\text{7. Update the policy: } \pi(s_t) = \frac{1}{Z} e^{\lambda \cdot A_t} \\
&\text{8. Return: } \hat{\pi}(s_{t+1}) = \frac{1}{Z} e^{\lambda \cdot A_t} \\
&\text{9. End}
\end{align*}

4 Quantitative Results

We experiment with three datasets: multi-task versions of MNIST (MNIST-MTL) (LeCun et al., 1998), Mini-ImageNet (MIN-MTL) (Vinyals et al., 2016) as introduced by (Ravi & Larochelle, 2017), and CIFAR-100 (CIFAR-MTL) (Krizhevsky, 2009) where we treat the 20 superclasses as tasks. In the binary MIN-MTL dataset, the task is to differentiate instances of a given class from non-instances. We create 10 tasks and for each we use 1k instances of the positive class s and 1k each of the remaining 9 negative classes for a total of 10k instances per task during training, which we then test on 200 samples per task (2k samples in total). MIN-MTL is a smaller version of ImageNet (Ding et al., 2009) which is easier to train in reasonable time periods. For mini-ImageNet we randomly choose 50 labels and create tasks from 10 disjoint random subsets of 5 labels each chosen from these. Each label has 800 training instances and 50 testing instances – so 4k training and 250 testing instances per task. For all 10 tasks we have a total of 40k training instances. Finally, a Nash equilibrium is a set of policies for each agent where each agent’s expected return will be lower if that agent unilaterally changes its policy.
Augmenting Models

Published as a conference paper at ICLR 2018

The task-specific-1-fc baseline has a separate last fully connected layer for each task and shares the rest of the layers for all tasks. The task-specific-all-fc baseline has a separate set of all the fully connected layers for each task. These baseline architectures allow considerable sharing of parameters but also grant the network private parameters for each task to avoid interference. However, unlike routing networks, the choice of which parameters are shared for which tasks, and which parameters are task-private is made statically in the architecture, independent of task.

The results are shown in Figures 7, 8, and 8. In each case the routing net routing-all-fc performs consistently better than the cross-stitch networks and the baselines. On CIFAR-MTL, the routing net beats cross-stitch networks by 7% and the next closest baseline task-specific-1-fc by 11%. On MIN-MTL, the routing net beats cross-stitch networks by about 2% and the nearest baseline task-specific-1-fc by about 6%. We surmise that the results are better on CIFAR-MTL because the task instances have more in common whereas the MIN-MTL tasks are randomly constructed, making sharing less profitable.

On MNIST-MTL the random baseline is 90%. We experimented with several learning rates but were unable to get the cross-stitch networks to train well here. Routing nets beats the cross-stitch networks by 9% and the nearest baseline (task-specific-all-fc) by 3%. The soft version also had trouble learning on this dataset.

In all these experiments routing makes a significant difference over both cross-stitch networks and the baselines and we conclude that a dynamic policy which learns the function blocks to compose on a per-task basis yields better accuracy and sharper convergence than simple static sharing baselines or a soft attention approach.

In addition, routing training is much faster. On CIFAR-MTL for example, training time on a stable compute cluster was reduced from roughly 38 hours to 5.6, an 85% improvement. We have conducted a set of scaling experiments to compare the training computation of routing networks and cross-stitch networks trained with 2, 3, 5, and 10 function blocks. The results are shown in the appendix in Section 7.3. Routing networks consistently perform better than cross-stitch networks and the baselines across all these problems. Adding function blocks has no apparent effect on the computation involved in training routing networks on a dataset of a given size. On the other hand, cross-stitch networks have a soft routing policy that scales computation linearly with the number of function blocks. Because the soft policy backpropagates through all function blocks and the hard routing policy only backpropagates through the selected block, the hard policy can much more easily scale to many task learning scenarios that require many diverse types of functional primitives.

To explore why the multi-agent approach seems to do better than the single-agent, we manually compared their policy dynamics for several CIFAR-MTL examples. For these experiments we simply froze the policy network and compared the state-action values using a fixed state and action. The single agent did not learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was able to learn a network that was能够完成任务。
that this is not what the network learns on its own. Here we see that the agents have converged on a strategy which first uses 7 function blocks, then compresses to just 4, then again expands to use 5. It is not clear if this is an optimal strategy but it does certainly give improvement over the static baselines.

6 Future Work

We have presented a general architecture for routing and multi-agent router training algorithm which performs significantly better than cross-stitch networks and baselines and other single-agent approaches. The paradigm can easily be applied to a state-of-the-art network to allow it to learn to dynamically adjust its representations.

As described in the section on Routing Networks, the state space to be learned grows exponentially with the depth of the routing, making it challenging to scale the routing to deeper networks in their entirety. It would be interesting to try hierarchical RL techniques (Barto & Mahadevan (2003)) here. Our most successful experiments have used the multi-agent architecture with one agent per task, trained with the Weighted Policy Learner algorithm (Algorithm 3). Currently this approach is tabular but we are investigating ways to adapt it to use neural net approximators.

We have also tried routing networks in an online setting, training over a sequence of tasks for few shot learning. To handle the iterative addition of new tasks we add a new routing agent for each and overfit it on the few shot examples while training the function modules with a very slow learning rate. Our results so far have been mixed, but this is a very useful setting and we plan to return to this problem.

References


Augmenting Models

Published as a conference paper at ICLR 2018


7 Appendix

7.1 Impact of Rho

Figure 12: Influence of the “collaboration reward” $\rho$ on CIFAR-MTL. The architecture is routing-all-fc with WPL routing agents.

Figure 13: Comparison of per-task training cost for cross-stitch and routing networks. We add a function block per task and normalize the training time per epoch by dividing by the number of tasks to isolate the effect of adding function blocks on computation.

7.2 The PASS action

When routing networks, some resulting sets of function blocks can be applied repeatedly. While there might be other constraints, the prevalent one is dimensionality - input and output dimensions need to match. Applied to the SimpleConvNet architecture used throughout the paper, this means that of the fc layers - (convolution $\rightarrow$ 48), (48 $\rightarrow$ #classes), the middle transformation can be applied an arbitrary number of times. In this case, the routing network becomes fully recurrent and the PASS action is applicable. This allows the network to shorten the recursion depth.

7.3 Overview of Implementations

We have tested 9 different implementation variants of the routing architectures. The architectures are summarized in Tables 3 and 4. The columns are:

#Agents refers to how many agents are used to implement the router. In most of the experiments, each router consists of one agent per task. However, as described in 3.1, there are implementations with 1 and #tasks + 1 agents.

Table 2: Numeric results (in % accuracy) for Figures 4 through 8

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>50</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REINFORCE: approx</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Qlearning: approx</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>24</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>Qlearning: table</td>
<td>20</td>
<td>56</td>
<td>47</td>
<td>50</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>MARL-WPL: table</td>
<td>31</td>
<td>53</td>
<td>57</td>
<td>58</td>
<td>60</td>
<td>60</td>
</tr>
</tbody>
</table>

| arch (Figure 5) |     |     |     |     |     |     |
| routing-all-fc |     |     |     |     |     |     |
| routing-all-fc recursive | 31 | 43 | 45 | 48 | 48 | 46 |
| routing-all-fc dispatched | 20 | 23 | 28 | 37 | 42 | 41 |
| soft mixture-all-fc | 20 | 24 | 27 | 30 | 32 | 35 |
| routing-all-fc single agent | 20 | 23 | 33 | 42 | 44 | 44 |
| CIFAR (Figure 6) |     |     |     |     |     |     |
| task specific-all-fc | 21 | 29 | 33 | 36 | 42 | 42 |
| task specific-1-fc | 27 | 34 | 39 | 42 | 48 | 49 |
| cross-stitch-all-fc | 26 | 37 | 42 | 52 | 53 |     |

| arch (Figure 7) |     |     |     |     |     |     |
| routing-all-fc |     |     |     |     |     |     |
| routing-all-fc recursive | 34 | 54 | 57 | 58 | 58 | 57 |
| task specific-all-fc | 22 | 30 | 37 | 43 | 47 | 48 |
| task specific-1-fc | 29 | 38 | 43 | 46 | 51 |     |
| cross-stitch-all-fc | 29 | 41 | 48 | 53 | 56 | 55 |
| MIN (Figure 7) |     |     |     |     |     |     |
| task specific-all-fc | 90 | 90 | 98 | 99 | 99 | 99 |
| task specific-1fc | 90 | 91 | 94 | 95 | 95 | 96 |
| MNIST (Figure 8) |     |     |     |     |     |     |
| task specific-all-fc | 90 | 90 | 91 | 92 | 93 | 93 |
| task specific-1fc | 90 | 90 | 90 | 90 | 90 | 90 |
| soil mixture-all-fc | 90 | 90 | 90 | 90 | 90 | 90 |
| cross-stitch-all-fc | 90 | 90 | 90 | 90 | 90 | 90 |

Figure 15: Results on the first $n$ tasks of CIFAR-MTL.
Augmenting Models

Published as a conference paper at ICLR 2018

<table>
<thead>
<tr>
<th>Name</th>
<th>Num Agents</th>
<th>Policy Representation</th>
<th>Part of State = (v, t, d) Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARL:WPL</td>
<td>MARL:WPL</td>
<td>Tabular (num layers x num function blocks)</td>
<td>t, d</td>
</tr>
<tr>
<td>REINFORCE</td>
<td>REINFORCE</td>
<td>Vector (num layers) of approx functions</td>
<td>v, t, d</td>
</tr>
<tr>
<td>Q-Learning</td>
<td>Q-Learning</td>
<td>Tabular (num layers x num function blocks)</td>
<td>t, d</td>
</tr>
</tbody>
</table>

Table 3: Implementation details for Figure 4. All approx functions are 2 layer MLPs with a hidden dim of 64.

<table>
<thead>
<tr>
<th>Name</th>
<th>Num Agents</th>
<th>Policy Representation</th>
<th>Part of State = (v, t, d) Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>routing-all-fc</td>
<td>routing-all-fc</td>
<td>Tabular (num layers x num function blocks)</td>
<td>t, d</td>
</tr>
<tr>
<td>routing-all-fc non-layered</td>
<td>routing-all-fc</td>
<td>Tabular (num layers x num function blocks)</td>
<td>t, d</td>
</tr>
<tr>
<td>dispatched-routing-all-fc</td>
<td>dispatched-routing-all-fc</td>
<td>Vector (num layers) of approx functions + dispatcher</td>
<td>v, t, d</td>
</tr>
<tr>
<td>single-agent-routing-all-fc</td>
<td>single-agent-routing-all-fc</td>
<td>Vector (num layers) of approx functions)</td>
<td>v, t, d</td>
</tr>
</tbody>
</table>

Table 4: Implementation details for Figure 5. All approx functions are 2 layer MLP’s with a hidden dim of 64.

Policy Representation There are two dominant representation variations, as described in 3.1. In the first, the policy is stored as a table. Since the table needs to store values for each of the different layers of the routing network, it is of size num layers x num actions. In the second, it is represented either as vector of MLP’s with a hidden layer of dimension 64, one for each layer of the routing network. In this case the input to the MLP is the representation vector v concatenated with a one-hot representation of the task identifier.

Policy Input describes which parts of the state are used in the decision of the routing action. For tabular policies, the task is used to index the agent responsible for handling that task. Each agent then uses the depth as a row index into the table. For approximation-based policies, there are two variations. For the single agent case the depth is used to index an approximation function which takes as input concat(v, one-hot(t)). For the multi-agent (non-dispatched) case the task label is used to index the agent and then the depth is used to index the corresponding approximation function for that depth, which is given concat(v, one-hot(t)) as input. In the dispatched case, the dispatcher is given concat(v, one-hot(t)) and predicts an agent index. That agent uses the depth to find the approximation function for that depth which is then given concat(v, one-hot(t)) as input.

7.4 EXPLANATION OF THE WEIGHTED POLICY LEARNER (WPL) ALGORITHM

The WPL algorithm is a multi-agent policy gradient algorithm designed to help dampen policy oscillation and encourage convergence. It does this by slowly scaling down the learning rate for an agent after a gradient change in that agents policy. It determines when there has been a gradient change by using the difference between the immediate reward and historical average reward for the action taken. Depending on the sign of the gradient the algorithm is in one of two scenarios. If the gradient is positive then it is scaled by $1 - \pi(a_t)$). Over time if the gradient remains positive it will cause $\pi(a_t)$ to increase and so $1 - \pi(a_t)$ will go to 0, slowing the learning. If the gradient is negative then it is scaled by $\pi(a_t)$. Here again if the gradient remains negative over time it will cause $\pi(a_t)$ to decrease eventually to 0, slowing the learning again. Slowing the learning after gradient changes dampens the policy oscillation and helps drive the policies towards convergence.
Learning Deep Network Representations with Adversarially Regularized Autoencoders

Wenchao Yu, Cheng Zheng, Wei Cheng, Charu C. Aggarwal, Dongjin Song, Bo Zong, Haifeng Chen, and Wei Wang

1Department of Computer Science, University of California Los Angeles
2NEC Laboratories America, Inc.
3IBM Research AI

ABSTRACT

The problem of network representation learning, also known as network embedding, arises in many machine learning tasks assuming that there exist a small number of variables in the vertex representations which can capture the "semantics" of the original network structure. Most existing network embedding models, with shallow or deep architectures, learn vertex representations from the sampled vertex sequences such that the low-dimensional embeddings preserve the locality property and/or global reconstruction capability. The resultant representations, however, are difficult for model generalization due to the intrinsic sparsity of sampled sequences from the input network. As such, an ideal approach to address the problem is to generate vertex representations by learning a probability density function over the sampled sequences. However, in many cases, such a distribution in a low-dimensional manifold may not always have an analytic form. In this study, we propose to learn the network representations with adversarially regularized autoencoders (NetRA). NetRA learns smoothly regularized vertex representations that will capture the network structure through jointly considering both locality-preserving and global reconstruction constraints. The joint inference is encapsulated in a generative adversarial training process to circumvent the requirement of an explicit prior distribution, and thus obtains better generalization performance. We demonstrate empirically how well key properties of the network structure are captured and the effectiveness of NetRA on a variety of tasks, including network reconstruction, link prediction, and multi-label classification.

KEYWORDS

Network embedding, autoencoder, generative adversarial networks

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Research Track Paper
KDD 2018, August 19–23, 2018, London, United Kingdom

1 INTRODUCTION

Network analysis has been attracting many research interests with its enormous potential in mining useful information which benefits the downstream tasks such as link prediction, community detection and anomaly detection on social network [34], biological networks [31] and language networks [28], to name a few.

To analyze network data, one fundamental problem is to learn a low-dimensional vector representation for each vertex, such that the network structure is preserved in the learned vector space [23]. For this problem, there are two major challenges: (1) preservation of complex structure property. The objective of network embedding is to train a model to "fit" the training networks, that is, to preserve the structure property of networks [23, 26]. However, the latent structure of the network is too complex to be portrayed by an explicit form of probability density which can capture both the local neighborhood information and global network structure.

(2) Sparsity of network sampling. Current research on network embedding employs network sampling techniques, including random walk sampling, breadth-first search etc., to derive vertex sequences as training datasets. However the sampled data represent only a small proportion of all the vertex sequences. An alternative approach is to encode these discrete structures in a continuous code space [37]. Unfortunately, learning continuous latent representations of discrete networks remains a challenging problem since in many cases, the prior distribution may not exist in a low dimensional manifold [26].

Recent work on network embedding has shown fruitful progress in learning vertex representations of complex networks [23, 26, 37]. These representations employ nonlinear transformations to capture the "semantics" of the original networks. Most existing methods first employ a random walk technique to sample a bunch of vertex sequences from the input network, then feed a learning model with these sequences to infer the optimal low-dimensional vertex embedding. However, the sampling strategy suffers from the data sparsity problem since the total amount of vertex sequences is usually very large in real networks and it is often intractable to enumerate all.

Subsequently, learning on a sparse sample set tends to produce an overly complex model to explain the sampled dataset, which eventually causes overfitting. Though autoencoders are adopted to encode the inputs into continuous latent representations [37], regularizations are still desirable to force the learned representations
2.2 Generative Adversarial Networks

The Generative Adversarial Networks (GANs) [11] build an adversarial training platform for two players, namely generator $g(x)$ and discriminator $d(x)$, to play a minimax game.

$$\min_{g(x)} \max_{d(x)} \mathbb{E}_{x \sim p_{data}(x)} \left[ \log d(x) \right] + \mathbb{E}_{z \sim p_z(z)} \left[ \log (1 - d(g(z))) \right]$$

The generator $g(x)$ tries to map the image to the output space as closely as the true data, while the discriminator $d(x)$ represents the probability that $x$ came from the data rather than the noise. It aims to distinguish real data distribution $P_{data}(x)$ and fake sample distribution $P_{gen}(x)$, e.g., $x \sim N(0,1)$. Wasserstein GANs [7] overcome unstable training problem by replacing Jensen-Shannon divergence with Earth-Mover (Wasserstein-1) distance, which considers solving the problem

$$\min_{g(x)} \max_{d(x)} \mathbb{E}_{x \sim p_{data}(x)} \left[ d(x) \right] + \mathbb{E}_{z \sim p_z(z)} \left[ 1 - d(g(z)) \right]$$

The Lipschitz constraint \(W\) on discriminator has been kept by clipping the weights of the discriminator within a compact space \([-c, c]\).

2.3 Network Embedding

Network embedding approaches seek to learn representations that encode structural information about the network. These approaches learn embeddings for vertices as points in a low-dimensional space. Given the encoded vector set \(\{x_1, \ldots, x_n\}\), finding an embedding \(\{z_1, \ldots, z_n\}\) can be formalized as an optimization problem [39, 41].

$$\min_{\{z_1, \ldots, z_n\}} \sum_{i,j} \|f(z_i) - f(z_j)\|^2$$

where \(f(z_i) \in \mathbb{R}^d\) is the embedding result for a given input \(z_i\). \(L\) is the loss function between a pair of inputs. \(\phi\) is the weight between \(d_i\) and \(x_i\).

3 APPROACH

In this section, we present NiRVA, a deep network embedding model using adversarially regularized autoencoders, to learn smoothly regularized vertex representations with sequences of vertices as inputs. The resultant representations can be used in downstream tasks, such as link prediction, network reconstruction and multi-class classification.

3.1 Random Walk Generator

Given network \(G(V, E)\), the random walk generator in DeepWalk [23] is utilized to obtain truncated random walks (i.e., sequences of vertices) rooted on each vertex \(u \in V\) in \(G(V, E)\). A walk is sampled randomly from the neighbors of the last visited vertex until the pre-processing length is reached.
Augmenting Models

Directed 76,584 Avg. degree 8.38
Undirected 1,899 4.23

We can separate the training of generator and discriminator. As
(14)
(15)

The dual form of the Earth Mover’s

where $\lambda$ is a convex parameter (Step 20-23). After the training of

Before the training of $G$, the encoder representation and the output of generator will be fed

to each simulation. The latent vectors are in close proximity (Step 2-7 in Algorithm 1). The latent

Then we get

Let $x \in \mathbb{R}^d$ be any distribution. Let $W \in \mathbb{R}^{d \times d}$ be a square matrix whose diagonal matrix $\Theta\in \mathbb{R}^{d \times d}$ is equal to

Our encoder $\phi$ can be defined as,

Let $x \in \mathbb{R}^d$ be any distribution. Let $W \in \mathbb{R}^{d \times d}$ be a square matrix whose diagonal matrix $\Theta\in \mathbb{R}^{d \times d}$ is equal to

where $\phi(z_i, x_i) \in \mathbb{R}$ is the Lipschitz continuity constraint (with

We evaluate the performance of our model with extensive experi-

we conduct experiments on a variety of networks from dif-

between the users (vertices) and edges containing sent messages (edges) between the users (vertices).

in each domain. The vertices represent users and the edges represent messages.

portmanteau theorem [21], we have

and $\text{Var}(\text{dist}(\tilde{W}(\phi(x)), \psi_{\theta}(x))) = \text{Var}(\text{dist}(\tilde{W}(\phi(x)), \psi_{\theta}(x)))$

where $\phi(z_i)$ is the Lipschitz continuity constraint (with

The training process of RefA consists of the following steps:

1. As shown in [36], $\text{dist}(\tilde{W}(\phi(x)), \psi_{\theta}(x))$ converges to $\text{dist}(\tilde{W}(\phi(x)), \psi_{\theta}(x))$

2. According to the Portmannen Theorem [36], $\text{dist}(\tilde{W}(\phi(x)), \psi_{\theta}(x))$ holds if $F, \mathbb{R}^x, \psi_{\theta}(x)$ is a bounded continuous function.

and

the number of discriminator training generation iteration.

Table 1: Statistics of the real-world network datasets

Dataset | V | E | Avg. degree | Label Type
--------|---|---|-------------|------------
BlogCatalog | 105,760 | 382,752 | 4.23 | Undirected
DBLP | 4,382 | 7,932 | 14.69 | Undirected
Flickr | 1,677 | 184,812 | 14.69 | Undirected
LSTM autoencoders are $\mathcal{O}(\text{Spinh} \times n \times |\psi|)$. Similarly, for the generator and discriminator, each invocation of backpropagation is typically linear in the number of parameters $|D(\phi)|$ and $|E(\psi)|$.

Thus the computational complexity for generator and discriminator is $\mathcal{O}(\text{Spinh} \times n \times |\psi|)$. It is basically quadratic if the input and hidden layers are of roughly the same size. However, if we set the size of embedding layers much less than that of the inputs, the time complexity reduces to $\mathcal{O}(n)$.

4 EVALUATION

In each domain. The vertices represent users and the edges represent messages.

Note that the training size of the sampled pairs is $O(n)$, which is much smaller than $O(n^2)$ because real networks are sparse in real settings.

The computational complexity of learning LSTM autoencoders is proportional to the number of parameters $|\phi| \times |\psi|$ in each iteration. Therefore, the learning computational complexity

$LSTM$ autoencoders are $\mathcal{O}(\text{Spinh} \times n \times |\psi|)$.
4.2 Comparing Algorithms
To evaluate the performance of our network embedding model, the competitors used in this paper are summarized as follows.

- **Spectral Clustering (GC)** [19]: GC is an approach based on matrix factorization, generating the vertex representation with the smallest $d$ eigenvectors of the normalized Laplacian matrix of the graph.
- **DeepWalk** [23]: DeepWalk is a skip-gram [20] based model which learns the graph embedding with truncated random walks.
- **node2vec** [12]: This approach combines the advantage of breadth-first traversal and depth-first traversal algorithms. The random walks generated by node2vec can better represent the structural equivalence.
- **Structural Deep Network Embedding (SDNE)** [17]: SDNE is a deep learning based network embedding model which uses autoencoder and locality preserving constraint to learn vertex representations that capture the highly non-linear network structure.
- **Adversarial Network Embedding (ANE)** [6]: ANE proposes a deep generative adversarial network based framework to learn the vertex representation that is adversarially difficult to distinguish from the real representations.

As observed in Figure 3, three classes are presented: red points for $\mathcal{RA}_N$, green points for $\mathcal{NE}_N$, and blue points for class $\mathcal{org}$am. The green points belong to class $\mathcal{org}$am, the blue points belong to class $\mathcal{java}$, and $\mathcal{NE}_N$ performs best as it achieves the highest precision for most $k$. By learning smoothly regularized vertex representations using generative adversarial training process [15], our model well integrates the locality-preserving and global reconstruction constraints to learn embeddings that capture the semantic information.

### 4.3 Visualization
In order to demonstrate how well key properties of network structure are captured by the network embedding models, we visualize the embeddings of each compared method. We run different embedding algorithms described in Section 4.2 to obtain low dimensional representations and map vertex vectors onto a two dimensional space using t-SNE [19]. With vertex colored by its label, we perform the visualization task on JDK dependency network, as shown in Figure 3.

As observed in Figure 3, three classes are presented: red points for $\mathcal{RA}_N$, green points for $\mathcal{NE}_N$, and blue points for $\mathcal{org}$am. The green points belong to class $\mathcal{org}$am, the blue points belong to class $\mathcal{java}$, and $\mathcal{NE}_N$ performs best as it achieves the highest precision for most $k$. By learning smoothly regularized vertex representations using generative adversarial training process [15], our model well integrates the locality-preserving and global reconstruction constraints to learn embeddings that capture the semantic information.

### 4.4 Link Prediction
The objective of link prediction task is to infer missing edges given a network with a certain fraction of edges removed. We randomly remove 50% of edges from the network, which serve as positive samples, and select an equal number of vertex pairs without linkage between them as negative samples. With vertex representation learned by network embedding algorithms, we obtain the edge feature from the $l_2$ norm of two vertex vectors, and use it to predict missing edges. Because our focus in network embedding model, this simple experimental setup can evaluate the performance based on the assumption that the representations of two connected vertices should be closer in the Euclidean space. We use the area under curve (AUC) score for evaluation on link prediction task.

The results are shown in Table 2. Obviously, we observe that $\mathcal{NE}_N$ outperforms the baseline algorithms across all datasets by a large margin. It can be seen that $\mathcal{NE}_N$ achieves 5% to 32% improvement based on the AUC score on the four datasets. By comparing $\mathcal{NE}_N$, node2vec, and DeepWalk, which all use random walks as inputs, we can see the effectiveness of generative adversarial regularization for improving the generalization performance in network embedding. With random walk sequences, $\mathcal{NE}_N$ can overcome the sparsity issue from the sampled sequences of vertices.

We also plot the ROC curve of these four datasets, as shown in Figure 4(a)-(d). The ROC curve of $\mathcal{NE}_N$ dominates other approaches and is very close to the (0, 1) point. We train the $\mathcal{NE}_N$ model with different epochs for different datasets and embed the vertices to representations after each training epoch. The results are shown in Figure 4(e)-(h). Generally, we can observe that $\mathcal{NE}_N$ converges pretty fast with high AUC score almost after the first epoch. When comparing with DeepWalk, node2vec, SDNE and ANE, we can clearly see the better performance of $\mathcal{NE}_N$ on these datasets.

### 4.5 Network Reconstruction
Network embeddings are considered as effective representations of the original network. The vertex representations learned by networking embedding maintain the edge information for network reconstruction. We randomly select vertex pairs as edge candidates and calculate the Euclidean distance between the vertices. We use the precision@k, the fraction of correct predictions in the top k predictions, for evaluation.

$$\text{precision}_k = \frac{1}{k} \sum_{j=1}^{k} \mathbf{I}(\frac{\mathbf{E}_{\text{true}}(j)}{\mathbf{E}_{\text{true}}(j)} \cap \mathbf{E}_{\text{pred}}(j))$$

where $\mathbf{E}_{\text{true}}(j, k)$ represents the top k predictions and $\mathbf{E}_{\text{true}}$ represents observed edges in original network. In the evaluation, the UCI and Blogcatalog datasets have been utilized to illustrate the performance of $\mathcal{NE}_N$, with results shown in Figure 5. As can be seen from the precision@k curves, the $\mathcal{NE}_N$ model achieves higher precision in the network reconstruction task. The total number of edge candidates selected in this task is 8 for UCI and 3064 for Blogcatalog. The reconstruction given by $\mathcal{NE}_N$ is very accurate in predicting most positive samples (results on JDK and DBLP datasets show similar trends which haven’t been included here). DeepWalk and node2vec can give reasonable reconstructions but the results are worse than $\mathcal{NE}_N$ for most k. By learning smoothly regularized vertex representations using generative adversarial training process [15], our model well integrates the locality-preserving and global reconstruction constraints to learn embeddings that capture the semantic information.

### 4.6 Multi-label Classification
The task of predicting vertex labels with representations learned by networking embedding algorithms is widely used in recent studies for performance evaluation [12, 23, 37]. As an effective network embedding algorithm should capture network topology and extract most useful features for downstream machine learning tasks. In this section, we use vertex features as input to a one-vs-rest logistic regression using the LIBLINEAR [9] package to train the classifiers. For the Wikipedia and PPI datasets, we randomly sample 16% to 50% of the vertex labels as the training set and use the remaining vertices as the test set. We report Micro-F1 [37] evaluation metrics. Each result is averaged by five runs, as shown in Figure 6.

It is evident from the figure that $\mathcal{NE}_N$ outperforms the state-of-the-art embedding algorithms on multi-label classification task. In the PPI dataset, $\mathcal{NE}_N$ achieves higher Micro-F1 scores than the baseline models by over 10% in all experiment settings. In the...
Augmenting Models

Problem due to the difficulty in back-propagation through discrete random variables on GANs like GraphGAN [38] and ANE [6] for discrete data is either through the use of discrete structures [5, 40] or the improved autoencoders [16].

6 CONCLUSION

In this study we proposed NeXRa, a deep network embedding model for encoding each vertex in a network as a low-dimensional vector representation with adversarially regularized autoencoders. Our model demonstrated the ability of generative adversarial training process in extracting informative representations. The proposed model has better generalization capability, without requiring an explicit prior density distributions for the latent representations. Specifically, we leveraged LSTM autoencoders that take the sampled sequences of vertices as input to learn smooth vertex representations regularized by locality-preserving constraint and general adversarial training process. The resultant representations are robust to the sparse vertex sequences sampled from the network. Empirically, we evaluated the learned representations with a variety of network datasets on different tasks such as network reconstruction, link prediction and multi-label classification. The results showed substantial improvement over the state-of-the-art network embedding competitors.

ACKNOWLEDGEMENT

The work is partially supported by NSFC U1936204, NSFC R11013014, and NSF DBI-3113668. Research of the fourth author was sponsored by the Army Research Laboratory and was also performed under Cooperative Agreement Number W911NF-19-2-0033. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on. We thank the anonymous reviewers for their careful reading and insightful comments on our manuscript.

REFERENCES


ABSTRACT

We propose a new Integral Probability Metric (IPM) between distributions: the Sobolev IPM. The Sobolev IPM compares the mean discrepancy of two distributions for functions (critic) restricted to a Sobolev ball defined with respect to a dominant measure \( \mu \). We show that the Sobolev IPM compares two distributions in high dimensions based on weighted conditional Cumulative Distribution Functions (CDF) of each coordinate on a leave one out basis. The Dominant measure \( \mu \) plays a crucial role as it defines the support on which conditional CDFs are compared. Sobolev IPM can be seen as an extension of the one dimensional Von Mises Cramér statistics to high dimensional distributions. We show how Sobolev IPM can be used to train Generative Adversarial Networks (GANs). We then exploit the intrinsic conditioning implied by Sobolev IPM in text generation. Finally we show that a variant of Sobolev GAN achieves competitive results in semi-supervised learning on CIFAR-10, thanks to the smoothness enforced on the critic by Sobolev GAN which relates to Laplacian regularization.

1 INTRODUCTION

In order to learn Generative Adversarial Networks (Goodfellow et al., 2014), it is now well established that the generator should mimic the distribution of real data, in the sense of a certain discrepancy measure. Discrepancies between distributions that measure the goodness of the fit of the neural generator to the real data distribution has been the subject of many recent studies (Arjovsky & Bottou, 2017; Nowozin et al., 2016; Kaae Sønderby et al., 2017; Mao et al., 2017; Arjovsky et al., 2017; Gulrajani et al., 2017; Mroueh et al., 2017; Mroueh & Sercu, 2017; Li et al., 2017), most of which focus on training stability.

In terms of data modalities, most success was booked in plausible natural image generation after the introduction of Deep Convolutional Generative Adversarial Networks (DCGAN) (Radford et al., 2015). This success is not only due to advances in training generative adversarial networks in terms of loss functions (Arjovsky et al., 2017) and stable algorithms, but also to the representation power of convolutional neural networks in modelling images and in finding sufficient statistics that capture the continuous density function of natural images. When moving to neural generators of discrete sequences generative adversarial networks theory and practice are still not very well understood. Maximum likelihood pre-training or augmentation, in conjunction with the use of reinforcement learning techniques were proposed in many recent works for training GAN for discrete sequences generation (Yu et al., 2016; Che et al., 2017; Hjelm et al., 2017; Rajeswar et al., 2017). Other methods included using the Gumbel Softmax trick (Kusner & Hernández-Lobato, 2016) and the use of auto-encoders to generate adversarially discrete sequences from a continuous space (Zhao et al., 2017). End to end training of GANs for discrete sequence generation is still an open problem (Press et al., 2017). Empirical successes of end to end training have been reported within the framework of WGAN-GP (Gulrajani et al., 2017), using a proxy for the Wasserstein distance via a
pointwise gradient penalty on the critic. Inspired by this success, we propose in this paper a new Integral Probability Metric (IPM) between distributions that we coin Sobolev IPM. Intuitively an IPM (Muller, 1997) between two probability distributions looks for a witness function \( f \), called critic, that maximally discriminates between samples coming from the two distributions:

\[
\sup_{f \in \mathcal{F}} \mathbb{E}_{P \sim X} f(x) - \mathbb{E}_{Q \sim \hat{X}} f(x)
\]

Traditionally, the function \( f \) is defined over a function class \( \mathcal{F} \) that is independent to the distributions at hand (Stiefelhagen et al., 2012). The Wasserstein-1 distance corresponds for instance to an IPM where the witness functions are defined over the space of Lipschitz functions; The MMD distance (Gretton et al., 2012) corresponds to witness functions defined over a ball in a Reproducing Kernel Hilbert Space (RKHS).

We will revisit in this paper Fisher IPM defined in (Moulines & Segrou, 2017), which extends the IPM definition to function classes defined with norms that depend on the distributions. Fisher IPM can be seen as restricting the critic to a Lebesgue ball defined with respect to a dominant measure \( \mu \). The Lebesgue norm is defined as follows:

\[
\int_{\mathcal{X}} |f(x)| \mu(dx)
\]

where \( \mu \) is a dominant measure of \( P \) and \( Q \).

In this paper we extend the IPM framework to critics bounded in the Sobolev norm:

\[
\int_{\mathcal{X}} \|\nabla f(x)\|_2^2 \mu(dx)
\]

In contrast to Fisher IPM, which compares joint probability density functions of all coordinates between two distributions, we will show that Sobolev IPM compares weighted (coordinate-wise) conditional Cumulative Distribution Functions for all coordinates on a leave on basis. Matching conditional dependencies between coordinates is crucial for sequence modeling.

Our analysis and empirical verification show that the modeling of the conditional dependencies can be built in to the metric used to learn GANs as in Sobolev IPM. For instance, this gives an advantage to Sobolev IPM in comparing sequences over Fisher IPM. Nevertheless, in sequence modeling when we parametrize the critic and the generator with a neural network, we find an interesting tradeoff between the metric used and the architectures used to parametrize the critic and the generator as well as the conditioning used in the generator. The burden of modeling the conditional long term dependencies can be handled by the IPM loss function as in Sobolev IPM (more accurately the critic of Sobolev IPM satisfies an elliptic Partial Differential Equation (PDE)). We relate this diffusion to the Fokker-Planck equation and show the behavior of the gradient of the optimal Sobolev critic as a transportation plan between distributions.

We empirically study Sobolev GAN in character level text generation (Section 6.1). We validate that the conditioning implied by Sobolev GAN is crucial for the success and stability of GAN in text generation. As a take home message from this study, we see that text generation succeeds either by implicit conditioning i.e using Sobolev GAN (or WGAN-GP) together with convolutional critics and generators, or by explicit conditioning using Fisher IPM together with recurrent critic and generator and curriculum learning.

We finally show in Section 6.2 that a variant of Sobolev GAN achieves competitive semi-supervised learning results on CIFAR-10, thanks to the smoothness implied by the Sobolev regularizer.

2 OVERVIEW OF METRICS BETWEEN DISTRIBUTIONS

In this Section, we review different representations of probability distributions and metrics for comparing distributions that use those representations. These metrics are at the core of training GAN. In what follows, we consider probability measures with a positive weakly differentiable probability density functions (PDF). Let \( P \) and \( Q \) be two probability measures with PDFs \( P(x) \) and \( Q(x) \) defined on \( X \subset \mathbb{R}^d \). Let \( F_P \) and \( F_Q \) be the Cumulative Distribution Functions (CDF) of \( P \) and \( Q \) respectively. For \( x = (x_1, ..., x_d) \), we have:

\[
F_P(x) = \int_{-\infty}^{x_1} \int_{-\infty}^{x_2} \int_{-\infty}^{x_3} \cdots \int_{-\infty}^{x_d} P(u_1, ..., u_d)du_1 \cdots du_d
\]

The score function of a density function is defined as: \( s_P(x) = \nabla \log(P(x)) \in \mathbb{R}^d \).

In this work, we are interested in metrics between distributions that have a variational form and can be written as a suprema of mean discrepancies of functions defined on a specific function class. This type of metrics include: \( \mu \)-divergences as well as Integral Probability Metrics (Sriperumbudur et al., 2009) and have the following form:

\[
d(\mu, P, Q) = \sup_{f \in \mathcal{F}} \|\Delta f / P, Q\|
\]

where \( \mu \) is a function class defined on \( \mathcal{X} \) and \( \Delta \) is a mean discrepancy, \( \Delta : \mathcal{F} \times R. \) The variational form given above leads in certain cases to closed form expressions in terms of the PDFs \( P, Q \) or in terms of the CDFs \( F_P, F_Q \) or the score functions \( s_P, s_Q \).

In Table 1, we give a comparison of different discrepancies \( \Delta \) and function spaces \( \mathcal{F} \) used in the literature for GAN training together with our proposed Sobolev IPM. We see from Table 1 that Sobolev IPM, compared to Wasserstein Distance, imposes a tractable smoothness constraint on the critic on points sampled from a distribution \( \mu \), rather then imposing a Lipschitz constraint on all points in the space \( \mathcal{X} \). We also see that Sobolev IPM is the natural generalization of the Cramér Vio-Mises Distance from one dimension to high dimensions. We note that the Energy Distance, a form of Maximum Mean Discrepancy for a special kernel, was used in (Bellemare et al., 2017b) as a generalization of the Cramér distance in GAN training but still needed a gradient penalty in its algorithmic counterpart leading to a mis-specified distance between distributions. Finally it is worth noting that when comparing Fisher IPM and Sobolev IPM we see that while Fisher IPM compares joint PDF of the distributions, Sobolev IPM compares weighted (coordinate-wise) conditional CDFs. As we will see later, this conditioning nature of the metric makes Sobolev IPM suitable for comparing sequences. Note that the Stein metric (Liu et al., 2016; Liu, 2017) uses the score function to match distributions. We will show later how Sobolev IPM relates to the Stein discrepancy (Appendix A).
We define the unit Lebesgue ball as follows:

$$B_2(X, \mu) = \{ f \in \mathcal{L}_2(X, \mu) \mid \| f \mathbb{1}_{[X,\mu]} \|_2 \leq 1 \}$$

Fisher IPM defined in (Mroueh & Sercu, 2017), searches for the critic function in the Lebesgue Ball

$$B_2(X, \mu)$$ that maximizes the mean discrepancy between P and Q. Fisher GAN (Mroueh & Sercu, 2017) was originally formulated specifically for $$\mu = \frac{1}{2}(P + Q)$$. We consider here a general $$\mu$$ as long as it dominates P and Q. We define Generalized Fisher IPM as follows:

$$\mathcal{F}_F(P, Q) = \sup_{f \in B_2(X, \mu)} E_{x \sim P \mid \mu}(f(x)) - E_{x \sim Q \mid \mu}(f(x))$$

Note that

$$E_{x \sim P \mid \mu}(f(x)) - E_{x \sim Q \mid \mu}(f(x)) = \int f(x, \theta) \pi_{x \sim P \mid \mu, \theta}(dx) - \int f(x, \theta) \pi_{x \sim Q \mid \mu, \theta}(dx)$$

Hence Fisher IPM can be written as follows:

$$\mathcal{F}_F(P, Q) = \sup_{f \in B_2(X, \mu)} \int f(x, \theta) \pi_{x \sim P \mid \mu, \theta}(dx) - \int f(x, \theta) \pi_{x \sim Q \mid \mu, \theta}(dx)$$

We have the following result:

**Theorem 1 (Generalized Fisher IPM).** The Fisher distance and the optimal critic are as follows:

1. The Fisher distance is given by:

$$\mathcal{F}_F(P, Q) = \frac{P - Q}{\mu_x} \mu_x \|_{L_1(x)} = \| E_{x \sim P \mid \mu}(x) - Q(x) \|_{L_1(x)}$$

2. The optimal $$f_\star$$ achieving the Fisher distance $$\mathcal{F}_F(P, Q)$$ is:

$$f_\star = \frac{1}{\mathcal{F}_F(P, Q)} P - Q \mu_x$$ almost surely.

**Proof of Theorem 1.** From Equation (3), the optimal $$f_\star$$ belong to the intersection of the hyperplane that has normal $$n = \frac{P - Q}{\mu_x}$$ and the ball $$B_2(X, \mu)$$, hence $$f_\star$$ belongs to $$B_2(X, \mu)$$. Hence

$$\mathcal{F}_F(P, Q) = \| n \|_{L_1(x)}$$

We see from Theorem 1 the role of the dominant measure $$\mu$$: the optimal critic is defined with respect to this measure and the overall Fisher distance can be seen as an average weighted distance between probability density functions, where the average is taken on points sampled from $$\mu$$. We give here some choices of $$\mu$$:

1. For $$\mu = \frac{1}{2}(P + Q)$$, we obtain the symmetric chi-squared distance as defined in (Mroueh & Sercu, 2017).
2. $$\mu = P$$, the implicit distribution defined by the interpolation lines between P and Q as in (Galrani et al., 2017).
3. When $$\mu$$ does not dominate P and Q, we obtain a non symmetric divergence. For example for $$\mu = P$$, $$\mathcal{F}_F(P, Q) = \int f(x, \theta) \pi_{x \sim Q \mid \mu, \theta}(dx)$$. We see here that for this particular choice we obtain the Pearson divergence.

4. **Sobolev IPM**

In this Section, we introduce the Sobolev IPM. In a nutshell, the Sobolev IPM constrains the critic function to belong to a ball in the restricted Sobolev Space. In other words we constrain the norm of the gradient of the critic $$\nabla f(x)$$. We will show that by moving from a Lebesgue constraint as in Fisher IPM to a Sobolev constraint as in Sobolev IPM, the metric changes from a joint PDF matching to weighted (coordinate-wise) conditional CDFs matching. The intrinsic conditioning built in to the Sobolev IPM and the comparison of cumulative distributions makes Sobolev IPM suitable for comparing discrete sequences.
3. The optimal critic $f^*$ satisfies the following identity:
\[
\nabla_x f^*(x) = \frac{1}{\mathbb{E}_{\mu}[f^*(x) - f_\mu(x)]} \mu(x) \quad \mu \text{ almost surely.}
\]

Sobolev IPM Approximation. Learning in the whole Sobolev space $W^{2,2}$ is challenging hence we need to restrict our function class to a hypothesis class $\mathcal{H}$, such as neural networks. We assume in the following that functions in $\mathcal{H}$ vanish on the boundary of $\mathcal{X}$, and restrict the optimization to the function space $\mathcal{H}$ or $\mathcal{H}$. In $\mathcal{H}$ can be a Reproducing Kernel Hilbert Space as in the MMD case or parametrized by a neural network. Define the Sobolev IPM approximation in $\mathcal{H}$:
\[
S_{\mathcal{H}}(\mu, \nu) = \sup_{f \in \mathcal{H}, \|f\|_{W^{2,2}} \leq 1} \left\{ \mathbb{E}_\mu f(x) - \mathbb{E}_\nu f(x) \right\}
\]

The following Lemma shows that the Sobolev IPM approximation in $\mathcal{H}$ is proportional to Sobolev IPM. The tightness of the approximation of the Sobolev IPM is governed by the tightness of the approximation of the optimal Sobolev Critic $f^*$ in $\mathcal{H}$. This approximation is measured in the Sobolev sense, using the Sobolev dot product.

Lemma 1 (Sobolev IPM Approximation in a Hypothesis Class). Let $\mathcal{H}$ be a function space with functions vanishing at the boundary. For any $f \in \mathcal{H}$ and for $f^*$ the optimal critic in $W^{2,2}$, we have:
\[
S_{\mathcal{H}}(\mu, \nu) = S_{\mathcal{H}}(\mu, \nu) \sup_{f \in \mathcal{H}, \|f\|_{W^{2,2}} \leq 1} \left\{ \mathbb{E}_\mu f(x) - \mathbb{E}_\nu f(x) \right\}
\]

Note that this Lemma means that the Sobolev IPM is well approximated if the space $\mathcal{H}$ has an enough representation power to express $\nabla_x f^*(x)$. This is parallel to the Fisher IPM approximation (Montiel & Serac, 2017) where it is shown that the Fisher IPM approximation error is proportional to the critical approximation in the Lebesgue sense. Having in mind that the gradient of the critic is the information that is passed on to the generator, we see that this convergence in the Sobolev sense to the optimal critic is an important property for GAN training.

Relation to Fokker-Planck Diffusion. We show in Appendix 5 that the optimal critic solves the following elliptic PDE (with zero boundary conditions):
\[
\frac{P - Q}{S_{\mathcal{H}}(\mu, \nu)} = -\text{div}(\mu(x)\nabla_x f^*(x)).
\]

We further link the elliptic PDE given in Equation (5) and the Fokker-Planck diffusion. As we illustrate in Figure 3(b) the gradient of the critic defines a transportation plan for moving the distribution mass from $Q$ to $P$.

Discussion of Theorem 2. We make the following remarks on Theorem 2:

1. From Theorem 2, we see that the Sobolev IPM compares $d$ higher order partial derivatives of the cumulative distributions $F_\mu$ and $F_\nu$, while Fisher IPM compares the probability density functions.

2. The dominant measure $\mu$ plays a similar role to Fisher:
\[
S_{\mu}(\mu, \nu) = \frac{1}{d^2} \sum_{i=1}^d \mathbb{E}_{\mu} \left( D^2 F_\mu(x) - D^2 F_\nu(x) \right)^2
\]

the average distance is defined with respect to points sampled from $\mu$. 

A. Definition and Expression of Sobolev IPM in Terms of Coordinate Conditional CDFs

We will start by recalling some definitions on Sobolev Spaces. We assume in the following that $\mathcal{X}$ is compact and consider functions in the Sobolev space $W^{1,2}(\mu, \mu)$:
\[
W^{1,2}(\mu, \mu) = \left\{ f : \mathcal{X} \to \mathbb{R}, \int_\mathcal{X} \| \nabla_x f(x) \|^2 \mu(x) dx < \infty \right\}
\]

We restrict ourselves to functions in $W^{1,2}(\mu, \mu)$ vanishing at the boundary, and note this space $W^{1,2}_0(\mu, \mu)$. Note that in this case:
\[
\|f\|_{W^{1,2}_0(\mu, \mu)} = \left\{ \int_\mathcal{X} \| \nabla_x f(x) \|^2 \mu(x) dx \right\}^{1/2}
\]
develops a semi-norm. We can similarly define a dot product in $W^{1,2}_0(\mu, \mu)$ for $f, g \in W^{1,2}_0(\mu, \mu)$:
\[
(f, g)_{W^{1,2}_0(\mu, \mu)} = \int_\mathcal{X} \nabla_x f(x) \cdot \nabla_x g(x) \mu(x) dx
\]

Hence we define the following Sobolev IPM, by restricting the critic of the mean discrepancy to the Sobolev unit ball:
\[
S_{\mu}(P, Q) = \sup_{f \in W^{2,2}_0(\mu, \mu)} \left\{ \mathbb{E}_\mu f(x) - \mathbb{E}_\nu f(x) \right\}
\]

When compared to the Wasserstein distance, the Sobolev IPM given in Equation (3) uses a data dependent gradient constraint (depends on $\mu$) rather than a data independent Lipschitz constraint. Let $F_\mu$ and $F_\nu$ be the cumulative distribution functions of $P$ and $Q$ respectively. We have:
\[
P(x) = \prod_{i=1}^d x_i \cdot \prod_{i=1}^d \frac{df}{dx_i}
\]

and we define
\[
D_i = \frac{df}{dx_i} = \prod_{j=1, j \neq i}^d x_j \quad i = 1, \ldots, d
\]

$D_i$ computes the $(d-1)$-th order partial derivative excluding the variable $i$.
3. Comparison of coordinate-wise Conditional CDFs. We note in the following $x^i = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d)$. Note that we have:

$$D^{-1}F_{\theta}(x) = \frac{\partial^{-1}}{\partial x_1 \ldots \partial x_{i-1} \partial x_{i+1} \ldots \partial x_d} \int_{-\infty}^{x_1} \int_{-\infty}^{x_2} \ldots \int_{-\infty}^{x_i} \int_{-\infty}^{x_{i+1}} \ldots \int_{-\infty}^{x_d} \mathbb{P}(a_1 \ldots a_d) \, da_1 \ldots \, da_d$$

$$= \int_{-\infty}^{x_1} \int_{-\infty}^{x_2} \ldots \int_{-\infty}^{x_i} \int_{-\infty}^{x_{i+1}} \ldots \int_{-\infty}^{x_d} \mathbb{P}(x_1 \ldots x_i, u, x_{i+1}, \ldots, x_d) \, du$$

$$= \mathbb{P}_X(x_1, \ldots, x_i, u, x_{i+1}, \ldots, x_d) \int_{-\infty}^{x_i} \mathbb{P}(x_i | x_1 \ldots x_{i-1}, u, x_{i+1} \ldots, x_d) \, dx_i$$

(Using Bayes rule)

$$= \mathbb{P}_X(x_1, \ldots, x_i, u, x_{i+1} \ldots, x_d) \mathbb{P}(x_i | x_1 \ldots x_{i-1}, u, x_{i+1} \ldots, x_d) \, dx_i$$

Note that for each $i$, $D^{-1}F_{\theta}(x)$ is the cumulative distribution of the variable $X_i$ given the other variables $X^i = x^i$, weighted by the density function of $X^i$ at $x^i$. This leads us to the form given in Equation 5.

We see that the Sobolev IPM compares for each dimension $i$ the conditional cumulative distribution of each variable given the other variables, weighted by their density function. We refer to this as comparison of coordinate-wise CDFs on a leave one out basis. From this we see that we are comparing CDFs, which are better behaved on discrete distributions. Moreover, the conditioning built in to this metric will play a crucial role in comparing sequences as the conditioning is important in this context (See section 6.1).

4.2 ILLUSTRATIVE EXAMPLES

Sobolev IPM / Cramer Distance and Wasserstein-1 in one Dimension. In one dimension, Sobolev IPM is the Cramér Distance (for $\mu$ uniform on $X$, we note this $\mu \sim U(\mathbb{R})$). While Sobolev IPM in one dimension measures the discrepancy between CDFs, the one dimensional Wasserstein-$p$ distance measures the discrepancy between inverse CDFs:

$$S^p_{\mu_1, \mu_2}(P, Q) = \int (F_{\mu_1}(x) - F_{\mu_2}(x))^p \, dx$$

Recall also that the Fisher IPM for uniform $\mu$ is given by:

$$\mathcal{F}^p_{\mu_1, \mu_2}(P, Q) = \int (\log \frac{P(x)}{Q(x)})^p \, dx$$

Consider for instance two point masses $P = \delta_{x_1}$ and $Q = \delta_{x_2}$ with $x_1, x_2 \in \mathbb{R}$. The rationale behind using Wasserstein distance for GAN training is that since it is a weak metric, for far distributions Wasserstein distance provides some signal (Arjovsky et al., 2017). In this case, it is easy to see that $\mathcal{F}^p_{\mu_1, \mu_2}(P, Q) = \mathcal{S}^p_{\mu_1, \mu_2}(P, Q) = 2$. As we see from this simple example, CDF comparison is more suitable than PDF for comparing distributions on discrete spaces. See Figure 1, for a further discussion of this effect in the GAN context.

Sobolev IPM between two 2D Gaussians. We consider $P$ and $Q$ to be two dimensional Gaussians with means $\mu_1$ and $\mu_2$ and covariances $\Sigma_1$ and $\Sigma_2$. Let $(x, y)$ be the coordinates in 2D. We note $F_2$ and $F_3$ the CDFs of $P$ and $Q$ respectively. We consider in this example $\mu_1 = (2, 2)$ and $\mu_2 = (2, -2)$. We know from Theorem 2 that the gradient of the Sobolev optimal critic is proportional to the following vector:

$$\nabla^T f(x, y) \propto \frac{1}{\mathbb{P}(x, y)} \left( \frac{\partial}{\partial x} (F_2(x, y) - F_3(x, y)) \right)$$

(9)

In Figure 2 we consider $\mu_1 = [1, 0]$, $\Sigma_1 = \begin{bmatrix} 1.9 & 0.8 \\ 0.8 & 1.3 \end{bmatrix}$ and $\mu_2 = [1, 2]$, $\Sigma_2 = \begin{bmatrix} 1.9 & -0.8 \\ -0.8 & 1.3 \end{bmatrix}$. In Figure 2(a) we plot the numerical solution of the PDE satisfied by the optimal Sobolev critic given in Equation (9), using MATLAB solver for elliptic PDEs (more accurately we solve $-\text{div}((\nabla f(x, \mu_1))/\mathbb{P}(x)) = F(x) - \mathbb{Q}(x)$, hence we obtain the solution of Equation (8) up to a normalization constant $\left(\frac{1}{\mathbb{P}(x)}\right)$). We numerically solve the PDE on a rectangle with zero boundary conditions with non overlapping supports.

Figure 1: In the GAN context for example in text generation, we have to match a (smoothed) discrete real distribution and a continuous generator. In this case, the CDF matching enabled by Sobolev IPM gives non zero discrepancy between a (smoothed) discrete and a continuous density even if the densities have disjoint supports. This ensures non vanishing gradients of the critic.

conditions. We see that the optimal Sobolev critic separates the two distributions well. In Figure 2(b) we then numerically compute the gradient of the optimal Sobolev critic on a 2D grid as given in Equation 9 (using numerical evaluation of the CDF and finite difference for the evaluation of the paritital derivatives). We plot in Figure 2(b) the density functions of $P$ and $Q$ as well as the vector field of the gradient of the optimal Sobolev critic. As discussed in Section A.1, we see that the gradient of the critic (ver to the input), defines on the support of $\mu \sim \mathcal{U}(\mathbb{R})$ a transportation plan for moving the distribution mass from $Q$ to $P$.

5 Sobolev GAN

Now we turn to the problem of learning GANs with Sobolev IPM. Given the "real distribution" $P_r \in \mathcal{P}_r(X)$, our goal is to learn a generator $g_{\theta} : \mathbb{R}^n \rightarrow X$, such that for $x \sim p_r$, the distribution of $g_{\theta}(x)$ is close to the real data distribution $P_r$, where $p_r$ is a fixed distribution on $\mathbb{R}$ (for instance $z \sim \mathcal{N}(0, I_1)$). We note $Q_r$ for the "fake distribution" of $g_{\theta}(z)$. $z \sim p_r$. Consider $\{x_i, i = 1 \ldots N\} \sim P_r$, $\{z_i, i = 1 \ldots N\} \sim \mathcal{N}(0, I_1)$, and $\{x_i, i = 1 \ldots N\} \sim \mu$. We consider these choices for $\mu$:

1. $\mu = \frac{p_r + p_h}{2}$, i.e. $\mu \sim P_r$, or $z \sim g_{\theta}(z)$, $z \sim p_r$, with equal probability $\frac{1}{2}$.

2. $\mu = \frac{p_r + p_h}{2}$ is the implicit distribution defined by the interpolation lines between $P_r$ and $Q_r$ as in (Gulrajani et al., 2017) i.e.: $z = x + (1 - u)y$, $z \sim P_r$, $y \sim g_{\theta}(z)$, $z \sim p_r$ and $u \sim \text{Unif}[0, 1]$.

Sobolev GAN can be written as follows:

$$\min_{\theta} \max_{\theta'} \sup_{(f_{\theta}, f_{\theta'})} \left( \mathbb{E}[f_{\theta}(g_{\theta}(z))] - \mathbb{E}[f_{\theta'}(x)] \right)$$

For any choice of the parametric function class $\mathcal{F}_{\theta'}$, note the constraint by $\Omega_1(f_{\theta}, g_{\theta}) = \frac{1}{2} \sum_{i=1}^{N} (\nabla f_{\theta}(x_i))^T \nabla f_{\theta}(x_i)$. For example if $\mu = \frac{p_r + p_h}{2}$, $\Omega_1(f_{\theta}, g_{\theta}) = \frac{1}{2\beta} \sum_{i=1}^{N} (\nabla f_{\theta}(x_i))^T \nabla f_{\theta}(x_i)$. 

Augmenting Models

Sobolev GAN

the reader is invited to check Appendix A.

Note that in Algorithm 1, we obtain a biased estimate since we are using same samples for the cost function and the constraint, but the incurred bias can be shown to be small and vanishing as the number of samples increases as shown and justified in (Shivaswamy & Jebara, 2010).

Relation to WGAN-GP. WGAN-GP can be written as follows:

$$\min_{\mu} \max_{\varphi, \psi} \mathbb{E}_{\mu}\left[\mathcal{L}_{\varphi}(\psi, \varphi)\right] = \min_{\mu} \max_{\varphi} \mathbb{E}_{\mu}\left[\mathcal{L}_{\varphi}(\psi, \varphi)\right] - \mathbb{E}_{\mu}\left[\mathcal{L}_{\varphi}(\psi, \psi)\right]$$

The main difference between WGAN-GP and our setting, is that WGAN-GP enforces pointwise constraints on points drawn from \(\mu = \mu_{\mathcal{P}}\) via a point-wise quadratic penalty \(\mathbb{E}_{\mu}\left[\mathcal{L}_{\varphi}(\psi, \psi)\right] = \mathbb{E}_{\mu}\left[\mathcal{L}_{\varphi}(\psi, \psi)\right] - \mathbb{E}_{\mu}\left[\mathcal{L}_{\varphi}(\psi, \psi)\right]\) while we enforce that constraint on average as a Sobolev norm, allowing us the coordinate weighted conditional CDF interpretation of the IPM.

6 APPLICATIONS OF SOBOLEV GAN

Sobolev IPM has two important properties. The first stems from the conditioning built in to the metric through the weighted conditional CDF interpretation. The second stems from the diffusion properties that the critic of Sobolev IPM satisfies (Appendix A) that has theoretical and practical ties to the Laplacian regularizer and diffusion on manifolds used in semi-supervised learning (Belkin et al., 2006).

In this section, we exploit those two important properties in two applications of Sobolev GAN: Text generation and semi-supervised learning. First in text generation, which can be seen as a discrete sequence generation, Sobolev GAN (and WGAN-GP) enable training GANs without need to do explicit brute-force conditioning. We attribute this to the built-in conditioning in Sobolev IPM (for the sequence aspect) and to the CDF matching (for the discrete aspect). Secondly using GANs in semi-supervised learning is a promising avenue for learning using unlabeled data. We show that a variant of Sobolev GAN can achieve strong SSL results on the CIFAR-10 dataset, without the need of any form of activation normalization in the networks or any ad hoc tricks.

6.1 Text Generation with Sobolev GAN

In this section, we present an empirical study of Sobolev GAN in character level text generation. Our empirical study on end to end training of character-level GAN for text generation is articulated on four dimensions (loss, critic, generator, \(\mu\)) (1) the loss used (GP: WGAN-GP (Gulrajani et al., 2017), 2) Sobolev or Fisher (2) the architecture of the critic (Resnets or RNN) (3) the architecture of the generator (Resnets or RNN with curriculum learning) (4) the sampling distribution \(\mu\) in the constraint.

Text Generation Experiments. We train a character-level GAN on Google Billion Word dataset and follow the same experimental setup used in (Gulrajani et al., 2017). The generated sequence length is 12 and the evaluation is based on Jensen-Shannon divergence on empirical 4-gram probabilities (JS-4) of validation data and generated data. JS-4 may not be an ideal evaluation criteria, but it is a reasonable metric for current character-level GAN results, which is still far from generating meaningful sentences.

Annealed Smoothing of discrete \(P_x\) in the constraint \(\mu\). Since the generator distribution will always be defined on a continuous space, we can replace the discrete “real” distribution \(P_x\) with a smoothed version (Gaussian kernel smoothing) \(\tilde{P}_x \sim N(0, \sigma^2 I)\). This corresponds to doing the following sampling for \(P_x\) : \(x \sim \tilde{P}_x \sim \tilde{P}_x \sim \tilde{P}_x \sim \tilde{P}_x \sim \tilde{P}_x \sim \tilde{P}_x \). Note that we only inject noise to the “real” distribution with the goal of smoothing the support of the discrete distribution, as opposed to instance noise on both “real” and “fake” to stabilize the training, as introduced in (Kurita et al., 2017; Arjovsky & Bottou, 2017). As it is common in optimization by continuation (Mohabi & Liu, 2015), we also anneal the noise level \(\sigma\) as the training progresses on a linear schedule.

Sobolev GAN versus WGAN-GP with ResNets. In this setting, we compare (WGAN-GP:ResNet-D⇒Resnet-\(\mu\) is Gaussian) to (Sobolev GAN:ResNet-D⇒Resnet-\(\mu\)) where \(\mu\) is one of: (1) \(\mu_{\mathcal{P}}\), (2) the noise smoothed \(\mu_{\mathcal{P}}(\sigma) = \mathcal{N}(0, \sigma^2 I)\), and (3) noise smoothed with annealing \(\mu_{\mathcal{P}}(\sigma)\), with the initial noise level. We use the same architectures of Resnet with 1D convolution for the critic and the generator as in (Gulrajani et al., 2017) (4 resnet blocks with hidden layer size of 512). In order to implement the noise smoothing we transform the data into one-hot vectors. Each one hot vector \(x\) is transformed to a probability vector \(\mathbf{1}(x)\) with a 0.9 replacing the one and 0.1/(\#labels - 1) replacing the zero. We then sample from a Gaussian distribution \(\mathcal{N}(0, \sigma^2)\) and

Algorithm 1 Sobolev GAN

Input: \(\rho\) penalty weight, \(\eta\) learning rate, \(n_i\) number of iterations for training the critic, \(N\) batch size

Initialize \(\rho, \eta, \lambda = 0\)

repeat

for \(j = 1\) to \(n_i\) do

Sample a minibatch \(z = \mathcal{N}(0, \sigma I)\) (\(N\), \(z < \mathcal{N}(0, \sigma I)\))

end for

Sample \(z = \mathcal{N}(0, \sigma I)\) (\(N\), \(z < \mathcal{N}(0, \sigma I)\))

\(\theta = \theta - \eta \cdot \text{ADAM}(\theta, \delta\theta)\)

until \(\delta\theta\) converges

Remark 1. Note that in Algorithm 1, we obtain a biased estimate since we are using same samples for the cost function and the constraint, but the incurred bias can be shown to be small and vanishing as the number of samples increases as shown and justified in (Shivaswamy & Jebara, 2010).
use softmax to normalize $\log \sigma + \epsilon$. We use algorithm 1 for Sobolev GAN and fix the learning rate to $10^{-4}$ and $\rho = 10^{-4}$. The noise level $\sigma$ was annealed following a linear schedule starting from an initial noise level $\sigma_0$ at iteration $i$, $\sigma_i = \sigma_0 (1- \frac{i}{\text{Maxiter}})$, Maxiter=30K. For WGAN-GP we used the open source implementation with the penalty $\lambda = 10$ as in (Gulrajani et al., 2017). Results are given in Figure 3(a) for the JS-4 evaluation of both WGAN-GP and Sobolev GAN for $\mu = \mu_{\text{GP}}$. In Figure 3(b) we show the JS-4 evaluation of Sobolev GAN with the annealed noise smoothing $\sigma_i(\nu_{\text{GP}})$ for various values of the initial noise level $\sigma_0$. We see that the training succeeds in both cases. Sobolev GAN achieves slightly better results than WGAN-GP for the annealing that starts with high noise level $\sigma_0 = 1.5$. We note that without smoothing and annealing i.e. using $\mu = \mu_{\text{GP}}$, Sobolev GAN is behind. Annealed smoothing of $\bar{P}_r$ helps the training as the real distribution is slowly going from a continuous distribution to a discrete distribution. See Appendix C (Figure 6) for a comparison between annealed and non annealed smoothing.

We give in Appendix C a comparison of WGAN-GP and Sobolev GAN for a ResNet generator architecture. The RNN has degraded performance due to optimization difficulties.

Fisher GAN Curriculum Conditioning versus Sobolev GAN: Explicit versus Implicit conditioning.

As mentioned before, the Sobolev norm as a regularizer for the Sobolev IPM draws connections with the Laplacian regularization in manifold learning (Belkin et al., 2006). In the Laplacian framework, the classifier satisfies a smoothness constraint imposed by controlling the Laplacian regularization in manifold learning (Belkin et al., 2006). In the Laplacian framework of semi-supervised learning, the classifier satisfies a smoothness constraint imposed by controlling in Sobolev norm $\int \nabla f(x)^\top \nabla f(x) \mu^n(x) dx$ (Alaoui et al., 2016). In this Section, we present a variant of Sobolev GAN that achieves competitive performance in semi-supervised learning on the CIFAR-10 dataset Krizhevsky & Hinton (2009) without using any internal activation normalization in the critic, such as batch normalization (BN) (Ioffe & Szegedy, 2015), layer normalization (LN) (Izra et al., 2016), or weight normalization (Salimans & Kingma, 2016).

In this setting, a convolutional neural network $\Phi : \mathbb{R}^m \rightarrow \mathbb{R}^n$ is shared between the cross entropy (CE) training of a $K$-class classifier ($\mathbb{R}^m \rightarrow \mathbb{R}^K$) and the critic of GAN (See Figure 5). We have the following training equations for the critic and classifier:

$$\text{Critic + Classifier:} \quad \max_{K \in \mathbb{R}^m} \mathcal{L}_{\text{C}} = \mathcal{L}_{\text{GAN}}(f, g_\theta) - \lambda \mathcal{L}_{\text{CE}} \sum_{x \in \mathcal{S}_T} \mathcal{L}_{\text{CE}}(p(y|x), y) \quad (11)$$

$$\text{Generator:} \quad \max_{K \in \mathbb{R}^m} \mathcal{L}_{\text{G}} = \delta(f, g_\theta) \quad (12)$$

where the main GAN objective with N samples: $\delta(f, g_\theta) = \frac{1}{N} \sum_{x \in \mathcal{S}_N} f(g_\theta(x))$.

Following (Moseh & Sercu, 2017) we use the following $K+1$ parameterization for the critic (See Figure 5):

$$f(x) = \sum_{k=1}^{K+1} p(x|\omega_k) \Phi_k(x) + \left[ (z, \Phi_k(z)) \right] f_k \text{real critic}$$

Note that $p(x|\omega_k) = \text{Softmax}(\mathbb{S}, \Phi_k(x))$, appears both in the critic formulation and in the Cross-Entropy term in Equation (11). Intuitively this critic uses the $K$ class directions of the classifier $\Phi_k$ to define the “real” direction, which competes with another $K+1$ direction $\phi$ that indicates fake samples. This parameterization adapts the idea of (Salimans et al., 2016), which was formulated specifically for the classic KL/JSD based GANs, to IPM-based GANs. We saw consistently better results with the $K+1$ formulation over the regular formulation where the classification layer $S$...
Propose the following two schemes for constraining the K+1 critic $f(x) = f^+(x) - f^-(x)$:

1. Fisher constraint on the critic: We restrict the critic to the following:
   \[
   f \in \left\{ f = f_s - f_o, \quad \hat{\Omega}(f, g_o) = \frac{1}{2N} \sum_{i=0}^{N-1} f^2(x) + \sum_{i=0}^{N-1} g^2(y_i) - 1 \right\}
   \]
   This constraint translates to the following ALM objective in Equation (11):
   \[
   \min f \quad \mathbb{E}_{x \sim \hat{p}} \left[ \| \nabla_x f(x) - \lambda (\hat{\Omega}(f, g_o) - 1) \|^2 \right]
   \]
   where the Fisher constraint ensures the stability of the training through an implicit whitened mean matching.

2. Fisher+ Sobolev constraint: We impose 2 constraints on the critic:
   \[
   f \in \left\{ f = f_s - f_o, \quad \hat{\Omega}(f, g_o) = 1 \right\}
   \]
   where $\hat{\Omega}(f, g_o) = \frac{1}{2N} \sum_{i=0}^{N-1} \| \nabla_x f(x) \|^2 + \sum_{i=0}^{N-1} \| \nabla_y f(x) \|^2$.
   This constraint translates to the following ALM in Equation (11):
   \[
   \min f \quad \mathbb{E}_{x \sim \hat{p}} \left[ \| \nabla_x f(x) - \lambda (\hat{\Omega}(f, g_o) - 1) \|^2 + \frac{\lambda^2}{2} \| \hat{\Omega}(f, g_o) - 1 \|^2 \right]
   \]
   where $\hat{\Omega}(f, g_o)$ enforces smoothness of the "fake" critic and thus the shared CNN $\Phi_\omega(x)$.

This is related to the classic Laplacian regularization in semi-supervised learning (Belkin et al., 2006).

7 Conclusion

We introduced the Sobolev IVM and showed that it amounts to a comparison between weighted (coordinate-wise) CDFs. We presented an ALM algorithm for training Sobolev GAN. The intrinsic conditioning implied by the Sobolev IVM explains the success of gradient regularization in Sobolev GAN and WGAN-GP on discrete sequence data, and particularly in text generation. We highlighted the important tradeoffs between the implicit conditioning introduced by the gradient regularizer in Sobolev IVM, and the explicit conditioning of Fisher IVM via recurrent critics and generators in conjunction with the curriculum conditioning. Both approaches succeed in text generation. We showed that Sobolev GAN achieves competitive semi-supervised learning results without the need of any normalization, thanks to the smoothness induced by the gradient regularizer. We think the Sobolev IVM point of view will open the door for designing new regularizers that induce different types of conditioning for general structured/discrete/graph data beyond sequences.

REFERENCES


Table 2: CIFAR-10 error rates for varying number of labeled samples in the training set. Mean and standard deviation computed over 5 runs. We only use the $K+1$ formulation of the critic. Note that we achieve strong SSL performance without any additional tricks, and even though the critic does not have any batch, layer or weight normalization. Baselines with * use either additional models like PixelCNN, or do data augmentation (translations and flips), or use a much larger model, either of which gives an advantage over our plain simple training method. ||[ ]|| is the result we achieved in our experimental setup under the same conditions but without “K+1” critic (see Appendix 3), since (Gulrajani et al., 2017) does not have SSL results.

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of labeled examples</th>
<th>2000</th>
<th>2000</th>
<th>4000</th>
<th>4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>CaGAN (Springenberg, 2015)</td>
<td>10.58</td>
<td>10.58</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FM (Salimans et al., 2016)</td>
<td>20.83 ± 0.20</td>
<td>19.63± 2.09</td>
<td>18.63 ± 2.32</td>
<td>17.72± 1.82</td>
<td></td>
</tr>
<tr>
<td>ALI (Dumoulin et al., 2017)</td>
<td>19.98 ± 0.32</td>
<td>19.09± 0.15</td>
<td>17.99 ± 0.54</td>
<td>17.6± 0.56</td>
<td></td>
</tr>
<tr>
<td>Tangents Reg (Kumar et al., 2017)</td>
<td>20.66 ± 0.5</td>
<td>16.76± 0.69</td>
<td>16.55 ± 0.29</td>
<td>16.6± 0.29</td>
<td></td>
</tr>
<tr>
<td>II-model (Lanre &amp; Arial, 2016)</td>
<td>14.87</td>
<td>14.87</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAT (Szegedy et al., 2017)</td>
<td>11.15</td>
<td>11.15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAT+EntMin+Large (Miyato et al., 2017)</td>
<td>11.29</td>
<td>11.29</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Vincent Dumoulin, IshmaelBelghazi, BenPoole, AlexLamb, MartinArjovsky, OlivierMastropietro, and AaronCourville. Adversarially learned inference. ICLR, 2017.


A Theory: Approximation and Transport Interpretation

In this Section we present the theoretical properties of Sobolev IPM and how it relates to distributions transport theory and other known metrics between distributions, notably the Stein distance.

A.1 Distribution Transport Perspective on Sobolev IPM

In this Section, we characterize the optimal critic of the Sobolev IPM as a solution of a non linear PDE. The solution of the variational problem of the Sobolev IPM satisfies a non linear PDE that can be derived using standard tools from calculus of variations (Ekeland & Turnbull, 1983; Alaoui et al., 2016).

Theorem 3 (PDE satisfied by the Sobolev Critic). The optimal critic of Sobolev IPM $f^*$ satisfies the following PDE:

$$\Delta f^*(x) + \langle \nabla_x \log \mu(x), \nabla_x f^*(x) \rangle \sum_{Q}(P, Q)P(x) - Q(x) = 0.$$  \hfill (13)

Define the Stein Operator: $T(\mu)(x) = \frac{1}{2} \left( \langle \nabla_x \log(\mu(x)), g(x) \rangle + \text{div}(g(x)) \right)$. Hence we have the following Transport Equation of $P$ to $Q$:

$$Q(x) = P(x) + 2S_n(P, Q)\mu(x)T(\mu)v_n^*(x).$$

Recall the definition of Stein Discrepancy $S_n(\mu, \nu) = \sup_{g \in C} |\mathbb{E}_{\mu} [T(\mu)(x)] - \mathbb{E}_{\nu} [g : X \rightarrow \mathbb{R}^d]|$.

Theorem 4 (Sobolev and Stein Discrepancies). The following inequality holds true:

$$\mathbb{E}_{\mu} \left[ \frac{Q(x) - P(x)}{\mu(x)} \right] \leq 2 \frac{S_n(\mu, \nu)}{S_n(P, Q)},$$

where $\mu = P$ and sequence $Q_n$. If the Sobolev distance goes $S_n(P, Q_n) \rightarrow 0$, the ratio $\mathbb{E}_{\mu} \left[ \frac{Q(x) - P(x)}{\mu(x)} \right]$ converges in expectation (w.r.t to $\mu$) to 1. The speed of the convergence is given by the Stein Discrepancy $S_n(\mu, P)$.

Relation to Fokker-Planck Diffusion Equation and Particles dynamics. Note that PDE satisfied by the Sobolev critic given in Equation (13) can be equivalently written:

$$\frac{\partial \rho_n}{\partial t}(x) = -\text{div}(\rho_n \nabla \phi_n(x)),$$

written in this form, we draw a connection with the Fokker-Planck Equation for the evolution of a density function $\rho_0$ that is the density of particles $X_t \in \mathbb{R}^d$ evolving with a drift ($a$ velocity field) $V(x, t) : \mathbb{R}^d \times [0, \infty) \rightarrow \mathbb{R}^d$:

$$dX_t = V(X_t, t)dt,$$

where the density of $X_t$ is given by $\rho(t) = Q(x)$. The Fokker-Planck Equation states that the evolution of the particles density $\rho_t$ satisfies:

$$\frac{\partial \rho}{\partial t}(x) = -\text{div}(\rho V(x, t)).$$

Comparing Equation (15) and Equation (16), we identify then the gradient of Sobolev critic $\phi_n$ as a drift. This suggests that one can define “Sobolev descent” as the evolution of particles along the gradient flow:

$$dX_t = \nabla \phi_n(X_t)dt,$$

where $\rho_t$ is the Sobolev critic between $\rho_0$ and $P$. One can show that the limit distribution of the particles is $P$. The analysis of “Sobolev descent” and its relation to Stein Descent (Liu & Wang, 2016; Liu, 2017) is beyond the scope of this paper and will be studied in a separate work. Hence we see that the gradient of the Sobolev critic defines a transportation plan to move particles whose distribution is $Q$ to particles whose distribution is $P$ (See Figure 3). This highlights the role of the gradient of the critic in the context of GAN training in term of transporting the distribution of the generator to the real distribution.
B PROOFS

Proof of Theorem 2. Let $F_1$ and $F_2$ be the cumulative distribution functions of $P$ and $Q$ respectively. We have:

$$P(x) = \frac{\partial^d}{\partial x_1 \cdots \partial x_d} F_2(x),$$  \hspace{2cm} (17)

We note $D = \frac{\partial^d}{\partial x_1 \cdots \partial x_d}$, and $D^{-i} = \frac{\partial^{d-i}}{\partial x_1 \cdots \partial x_d}$, for $i = 1 \ldots d$.

$D^{-i}$ computes the $i$-1 partial derivative excluding the variable $i$.

In the following we assume that $F_2$ and $F_1$ and its $d$ derivatives exist and are continuous meaning that $F_2$ and $F_1 \in C^d(X)$. The objective function in Equation (3) can be written as follows:

$$\mathbb{E}_{x \sim f}(x) - \mathbb{E}_{x \sim Q}(x) = \int f(x) D F_2(x) - F_2(x) dx$$

Let $D^{-i} = (D^{-1}, \ldots, D^{-d})$ it follows that:

$$\mathbb{E}_{x \sim f}(x) - \mathbb{E}_{x \sim Q}(x) = \frac{1}{d} \sum_{i=1}^{d} \int f_i (x) D F_2(x) - F_2(x) dx$$

Let us define $\mathcal{D}_p^2(X, \mu)^{\otimes d}$ the space of measurable functions from $X \rightarrow \mathbb{R}^d$. For $g, h \in \mathcal{D}_p^2(X, \mu)^{\otimes d}$ the dot product is defined as follows:

$$(g, h)_{\mathcal{D}_p^2(X, \mu)^{\otimes d}} = \int_X (g(x), h(x))_d \mu(dx)$$

and the norm is given:

$$\|g\|_{\mathcal{D}_p^2(X, \mu)^{\otimes d}} = \left( \int_X \|g(x)\|_2^2 \mu(dx) \right)^{1/2}$$

We can write the objective in Equation (18) in terms of the dot product in $\mathcal{D}_p^2(X, \mu)^{\otimes d}$:

$$\mathbb{E}_{x \sim f}(x) - \mathbb{E}_{x \sim Q}(x) = \frac{1}{d} \int (\nabla_x f \cdot D F_2(x) - F_2(x)) \mu(dx)$$

On the other hand the constraint in Equation (3) can be written in terms of the norm in $\mathcal{D}_p^2(X, \mu)^{\otimes d}$:

$$\|f\|_{L^2_p(X, \mu)} = \|\nabla_x f\|_{\mathcal{D}_p^2(X, \mu)^{\otimes d}}$$  \hspace{2cm} (20)

Replacing the objective and constraint given in Equations (19) and (20) in Equation (3), we obtain:

$$S(P, Q) = \frac{1}{d} \sup_{f \in \mathcal{D}_p^2(X, \mu)^{\otimes d}} \left\{ \nabla_x f \cdot D F_2(x) - F_2(x) \right\} \|x \|_{\mathcal{D}_p^2(X, \mu)^{\otimes d}}$$

$$- \frac{1}{d} \sup_{f \in \mathcal{D}_p^2(X, \mu)^{\otimes d}} \left\{ \nabla_x f \cdot D F_2(x) - F_2(x) \right\} \|x \|_{\mathcal{D}_p^2(X, \mu)^{\otimes d}}$$

Proof of Lemma 3.

$$\mathbb{E}_{x \sim f}(x) - \mathbb{E}_{x \sim Q}(x) = \frac{1}{d} \int (\nabla_x f \cdot D F_2(x) - F_2(x)) \mu(dx)$$

$$= S_p(P, Q) \left\{ \nabla_x f(x) \cdot D F_2(x) - F_2(x) \right\} \mu(dx)$$

$$= S_p(P, Q) \left\{ \nabla_x f(x) \cdot D F_2(x) - F_2(x) \right\} \mu(dx)$$

Hence we have:

$$S_{\mathcal{D}_p^2}(P, Q) = S_p(P, Q) \sup_{f \in \mathcal{D}_p^2(X, \mu)^{\otimes d}} \left( f, f^* \right)_{L^2_p}$$

It follows therefore that:

$$S_{\mathcal{D}_p^2}(P, Q) = S_p(P, Q) \sup_{f \in \mathcal{D}_p^2(X, \mu)^{\otimes d}} \left( f, f^* \right)_{L^2_p}$$

We conclude that the Sobolev IPM can be approximated in arbitrary space as long as it has enough capacity to approximate the optimal critic. Interestingly the approximation error is measured now with the Sobolev semi-norm, while in Fisher it was measured with the Lebesgue norm. Approximations with Sobolev Semi-norms are stronger then Lebesgue norms as given by the Poincare inequality ($\|f\|_{L^2_{\mathcal{D}_p^2}} \leq C \|f\|_{L^1_{\mathcal{D}_p^2}}$), meaning if the error goes to zero in Sobolev sense it also goes to zero in the Lebesgue sense, but the converse is not true.

Proof of Theorem 3. The proof follows similar arguments in the proofs of the analysis of Laplacian regularization in semi-supervised learning studied by (Alaoui et al., 2016).

$$S_{\mathcal{D}_p^2}(P, Q) = \sup_{f \in \mathcal{D}_p^2(X, \mu)^{\otimes d}} \left\{ \mathbb{E}_{x \sim P}(f(x)) - \mathbb{E}_{x \sim Q}(f(x)) \right\} \|x \|_{\mathcal{D}_p^2(X, \mu)^{\otimes d}}$$

$$\|\nabla_x f\|_{\mathcal{D}_p^2(X, \mu)^{\otimes d}} \leq 1, \hspace{2cm} (21)$$
Note that this problem is convex in $f$ (Ekeland & Turnbull, 1983). Writing the lagrangian for equation (21) we get:

$$L(f, \lambda) = \mathbb{E}_{x \sim \mathcal{D}}[f(x)] - \mathbb{E}_{x \sim q}[f(x)] + \frac{\lambda}{2} \left(1 - \mathbb{E}_{x \sim \mu}[\|\nabla f(x)\|^2]\right)$$

$$= \int_x f(x) \left(P(x) - Q(x)\right) dx + \frac{\lambda}{2} \left(1 - \int_x \left\|\nabla f(x)\right\|^2 \mu(x) dx\right)$$

$$= \int_x f(x) \mu_1(x) dx + \frac{\lambda}{2} \left(1 - \int_x \left\|\nabla f(x)\right\|^2 \mu(x) dx\right)$$

We denote $\{P(x) - Q(x)\}$ as $\mu_1(x)$. To get the optimal $f$, we need to apply KKT conditions on the above equation:

$$L(f, \lambda) = \int_x f(x) \mu_1(x) dx + \frac{\lambda}{2} \left(1 - \int_x \left\|\nabla f(x)\right\|^2 \mu(x) dx\right)$$

From the calculus of variations:

$$\frac{\partial L(f, \lambda)}{\partial f} = \int_x \left\{\nabla_x \left[\mathbb{E}_{x \sim \mathcal{D}}[f(x)] - \mathbb{E}_{x \sim q}[f(x)] + \frac{\lambda}{2} \left(1 - \mathbb{E}_{x \sim \mu}[\|\nabla f(x)\|^2]\right)\right]\right\} \mu(x) dx$$

$$= \int_x \left\{f(x) - P(x) + Q(x) + \frac{\lambda}{2} \left(1 - \left\|\nabla f(x)\right\|^2\right)\right\} \mu(x) dx$$

$$= \int_x \left\{f(x) - P(x) + Q(x) + \frac{\lambda}{2} \left(1 - \left\|\nabla f(x)\right\|^2\right)\right\} \mu(x) dx$$

Adding equation (28) for all $i$, we get:

$$\sum_i \frac{\partial f(x)}{\partial x_i} + \frac{\partial}{\partial x_i} \left(\mu(x) \nabla f(x)\right) + \frac{\partial}{\partial x_i} \left(\frac{P(x) - Q(x)}{\lambda^2 \mu(x)}\right) = 0$$

Hence, we have:

$$\frac{\partial f(x)}{\partial x_i} + \mu(x) \nabla f(x) + \frac{P(x) - Q(x)}{\lambda^2 \mu(x)} = 0$$

For the optimal $f^*$, the first order optimality condition gives us:

$$\mu(x) + \lambda^* \mu(x) \nabla f^*(x) = 0$$

and

$$\int_x \left\|\nabla f^*(x)\right\|^2 \mu(x) dx = 1$$

Note that (See for example (Alaoui et al., 2016)):

$$\nabla f^*(x) = \mu(x) \Delta f^*(x) + (\nabla \mu(x), \nabla f^*(x)),$$

since $\nabla f^*(x) = \Delta f^*(x)$. Hence from equation (22)

$$\mu(x) + \lambda^* \mu(x) \nabla f^*(x) = 0$$

$$\Rightarrow \mu(x) + \lambda^* \mu(x) \Delta f^*(x) + (\nabla \mu(x), \nabla f^*(x)) = 0$$

$$\Rightarrow \mu(x) + \lambda^* \mu(x) \Delta f^*(x) + (\nabla \mu(x), \nabla f^*(x)) = 0$$

Hence $f^*$, $\lambda^*$ satisfies:

$$\Delta f^*(x) + (\nabla \mu(x), \nabla f^*(x)) + \frac{P(x) - Q(x)}{\lambda^2 \mu(x)} = 0$$

and

$$\int x \nabla f^*(x) dx = 1.$$
As a result, the solution $f^*$ of the partial differential equation given in equation (25) satisfies the following:
\[
\frac{\partial^* f^*(x)}{\partial t} = -D^*F_2(x) + D^*F_3(x) \quad \forall x \in [d]
\]

Using the constraint in (26) we can get the value of $\lambda^*$:
\[
\int \left| \nabla f^*(x) \right|^2 \mu(x) \, dx = 1
\]
\[
\Rightarrow \int \sum_{i=1}^d \left( \frac{\partial^* f^*(x)}{\partial x_i} \right)^2 \mu(x) \, dx = 1
\]
\[
\Rightarrow \lambda^* = \frac{1}{d} \sum_{i=1}^d \left( \frac{D^*F_2(x) + D^*F_3(x)}{\mu(x)} \right) \, dx = S_\mu(P, Q).
\]

Proof of Theorem 4. Define the Stein operator (Oates et al., 2017):
\[
T(\mu)\nabla f(x) = \frac{1}{2} \left( \nabla \log \mu(x) + \nabla \log P(x) \right)
\]
\[
= \frac{1}{2} \left( \nabla \log \mu(x) + \nabla \log P(x) \right)
\]
This operator was later used in defining the Stein discrepancy (Gorham & Mackey, 2015; Liu et al., 2016). Recall that Barbour generator theory provides us a way of constructing such operators that produce
mean zero function under $\mu$. It is easy to verify that:
\[
\mathbb{E}_{x \sim \mu} T(\mu)\nabla f(x) = 0.
\]
Recall that this operator arises from the overdamped Langevin diffusion, defined by the stochastic differential equation:
\[
dx = -\nabla \log \mu(x) + dW_t
\]
where $(W_t)_{t \geq 0}$ is a Wiener process. This is related to plug and play networks for generating samples of the distribution is known, using the stochastic differential equation.

From Theorem 3, it is easy to see that the PDE the Sobolev Critic ($f^*, \lambda^* = S_\mu(P, Q)$) can be written in terms of Stein Operator as follows:
\[
T(\mu)\nabla f^*(x) = \frac{1}{2\lambda^*} \left( Q(x) - P(x) \right)
\]
Taking absolute values and the expectation with respect to $Q$:
\[
\left| \mathbb{E}_{x \sim \mu} T(\mu)\nabla f^*(x) \right| = \frac{1}{2\lambda^*} \frac{1}{S_\mu(P, Q)} \mathbb{E}_{x \sim \mu} \left[ \frac{|Q(x) - P(x)|}{\mu(x)} \right]
\]
Recall that the definition of Stein Discrepancy:
\[
S(Q, \mu) = \sup_{\|g\|_2 \leq 1} \left| \mathbb{E}_{x \sim \mu} T(\mu) g(x) \right|
\]
It follows that Sobolev IPM critic satisfies:
\[
\left| \mathbb{E}_{x \sim \mu} T(\mu)\nabla f^*(x) \right| \leq S(Q, \mu).
\]
Hence we have the following inequality:
\[
\frac{1}{2S_\mu(P, Q)} \mathbb{E}_{x \sim \mu} \left[ |Q(x) - P(x)| \right] \leq S(Q, \mu)
\]
This is equivalent to:
\[
\left| \mathbb{E}_{x \sim \mu} \left[ Q(x) - P(x) \right] \right| \leq 2S(Q, \mu) S_\mu(P, Q)
\]
Similarly we obtain:
\[
\left| \mathbb{E}_{x \sim \mu} \left[ Q(x) - P(x) \right] \right| \leq 2S(P, \mu) S_\mu(P, Q)
\]
For instance consider $\mu = P$, we have therefore:
\[
\frac{1}{2} \mathbb{E}_{x \sim \mu} \left[ \frac{Q(x)}{P(x)} \right] = 1 \leq S(Q, P) S_\mu(P, Q).
\]
Note that the left hand side of the inequality is not the total variation distance.
Hence for a sequence $Q_n$, if the Sobolev distance goes $S_\mu(P, Q_n) \to 0$, then the ratio $r_n(x) = \frac{Q_n(x)}{P(x)}$ converges in expectation (w.r.t to $Q$) to 1. The spread of the convergence is given by the Stein Discrepancy $S(Q_n, P)$.

One important observation here is that convergence of PDF ratio is weaker than the conditional CDF as given by the Sobolev distance and of the good fitness of score function as given by Stein discrepancy.

C. Text Experiments: Additional Plots

Comparison of annealed versus non annealed smoothing of $P_r$ in Sobolev GAN.

Figure 6: Comparison of annealed versus non annealed smoothing of $P_r$ in Sobolev GAN. We see that annealed smoothing outperforms the non annealed smoothing experiments.

Sobolev GAN versus WGAN-GP with RNN. We fix the generator architecture to Resnets. The experiments using RNN (GRUs) as the critic architecture for WGAN-GP and Sobolev is shown in Figure 7 where we used $\mu = \mu_{\text{CP}}$ for both cases. We only apply gradient clipping to stabilize the performance without other tricks. We can observe that using RNN degrades the performance. We think that this is due to an optimization issue and a difficulty in training RNN under the GAN objective without any pre-training or conditioning.
Figure 7: Result of WGAN-GP and Sobolev with RNNs.

Figure 8: Text samples from various GANs considered in this paper.

D SSL: HYPERPARAMETERS AND ARCHITECTURE

For our SSL experiments on CIFAR-10, we used Adam with learning rate $\eta = 2e^{-4}$, $\beta_1 = 0.5$ and $\beta_2 = 0.999$ both for critic $\phi$ (without BN) and Generator (with BN). We selected $\lambda_{GP} = 1.5$ from $\{0.8, 1.5, 3.0, 5.0\}$. We train all models for 350 epochs. We used some L2 weight decay: $1e^{-8}$ for $\phi$, $1e^{-5}$ for $D$, $1e^{-3}$ weight decay on the last layer $v$. For formulation 1 (Fisher only) we have $\rho_{\phi} = 1e^{-7}$, modified critic learning rate $\rho_{D} = 1e^{-4}$–$6$, critic iters $n_{v} = 2$. For formulation 2 (Sobolev + Fisher) we have $\rho_{\phi} = 5e^{-8}$, $\rho_{D} = 2e^{-8}$, critic iters $n_{v} = 1$. For the WGAN-GP (Gulrajani et al., 2017) baseline SSL experiment we followed the original paper with critic iters $n_{v} = 5$, $\rho_{\phi} = 1e^{-4}$–$6$, Adam $\beta_1=0.9$ and GP weight $\lambda_{GP} = 10.0$. Architectures are as below. We determined $\lambda_{GP} = 0.3$ to be optimal from $\{0.03, 0.1, 0.3, 1.0, 3.0\}$. As mentioned in Section 6.2, the K+1 critic formulation is not able to fit the training set with the GP constraint, so we fall back to the plain critic formulation where the critic $\langle v, \Phi_\omega(x) \rangle$ does not interact with the classifier $\langle S, \Phi_\omega(x) \rangle$.

Architecture:

```
for (i: 1; i < 30; i++) {
    Sequential {
        Conv2d(3, 96, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        LeakyReLU (0.2, inplace)
        Conv2d(96, 192, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        LeakyReLU (0.2, inplace)
        Conv2d(192, 192, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        LeakyReLU (0.2, inplace)
        Conv2d(192, 192, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        ReLU (inplace)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        ReLU (inplace)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        ReLU (inplace)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        ReLU (inplace)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        ReLU (inplace)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        ReLU (inplace)
        Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    }
}
```

```
for (i: 1; i < 30; i++) {
    Sequential {
        Conv2d(100, 256, kernel_size=(4, 4), stride=(1, 1), padding=(1, 1), bias=False)
        LeakyReLU (0.2, inplace)
        Conv2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
        Conv2d(128, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
        LeakyReLU (inplace)
        BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
    }
}
```

Development of the GANs considered in this paper.
Augmenting Models

Non-parametric estimation of
Jensen-Shannon Divergence in
Generative Adversarial Network training

Mathieu Sinn
IBM Research – Ireland
Mulhuddart, Dublin 15, Ireland
mathsinn@ie.ibm.com

Ambrish Rawat
IBM Research – Ireland
Mulhuddart, Dublin 15, Ireland
ambrish.rawat@ie.ibm.com

Abstract

Generative Adversarial Networks (GANs) have become a widely popular framework for generative modelling of high-dimensional datasets. However their training is well-known to be difficult. This work presents a rigorous statistical analysis of GANs providing straightforward explanations for common training pathologies such as vanishing gradients. Furthermore, it proposes a new training objective, Kernel GANs, and demonstrates its practical effectiveness on large-scale real-world datasets. A key element in the analysis is the distinction between training with respect to the (unknown) data distribution, and its empirical counterpart. To overcome issues in GAN training, we pursue the idea of smoothing the Jensen-Shannon Divergence (JSD) by incorporating noise in the input distributions of the discriminator. As we show, this effectively leads to an empirical version of the JSD in which the true and the generator densities are replaced by kernel density estimates, which leads to Kernel GANs.

1 INTRODUCTION

Generative Adversarial Networks (GANs), introduced by Goodfellow et al. [9], have become a widely popular framework for generative modeling using deep neural networks. While practitioners find that GANs – particularly for image data – produce sharp and realistic samples, it is well recognized that GANs are difficult to train. Key challenges are: vanishing gradients, local optima leading to mode collapse, high sensitivity to hyperparameters, and finding the right balance between generator and discriminator training in the adversarial set-up (Dinh et al. [5], Goodfellow [8], Goodfellow et al. [10], Metz et al. [21], Radford et al. [24], Salimans et al. [26]). Various authors have proposed practical modifications of GAN training to address these issues. However, only recently have authors began to analyze them mathematically and develop principled solutions. An important step in this direction was the work by Arjovsky and Bottou [1], which led to the idea of Wasserstein GANs elaborated in Arjovsky et al. [2] and further developed by Gulrajani et al. [12]. Two important insights were: 1) training the discriminator in GANs till optimality may provably result in vanishing gradients, and 2) the Jensen-Shannon Divergence (JSD) doesn’t yield meaningful information about convergence of distributions if their intersection with the support of the limit-distribution has measure zero. Another important contribution was the work by Metz et al. [21], who proposed to unroll discriminators in the GAN training objective in order to avoid degenerate optima and vanishing gradients.

Our contributions. This work has three major contributions.

- First, a rigorous mathematical framework to analyze GANs, which yields a remarkably simple explanation of the vanishing gradient problem.
Second, a novel training objective, Kernel GANs, backed with a principled theoretical analysis along with an empirical study that highlights practical aspects of Kernel GAN training.

Finally, experimentation with different training setups that scale Kernel GANs to large datasets and establish their practical usefulness.

Our rigorous mathematical framework allows us to formulate the results in Arjovsky and Bottou [1] more generally, and it addresses obscurities in the original theory developed in [9]. It appears that a major source of confusion has been the missing distinction between the GAN training objective with respect to the (unknown) target density, and its empirical counterpart used in practical training. This was recently independently pointed out by Arora et al. [3], however, without deriving the implications stated in the present work.

To overcome GAN training pathologies, we analyze approaches for smoothing the JSD in the GAN training objective. We pursue the idea of adding noise terms to the inputs of the discriminator. As a consequence, \( X \) is the support of \( X \), i.e., there doesn't exist an \( x \in X \) with an open neighborhood \( B_x \) in the topology of \( X \) such that \( P(X \in B_x) = 0 \).

Theorem 2.1. Suppose that (A1)-(A4) hold. Moreover suppose that \( Z = [0, 1] \), \( \mathcal{G} \) is the Borel \( \sigma \)-algebra on \([0, 1]\), and \( Z \) is uniformly distributed on \([0, 1]\). Then there exists a continuous surjection \( g: Z \rightarrow X \) such that \( P(g(Z) \in A) = P(X \in A) \) for all \( A \in \mathcal{A} \).

Note that, equivalently, one could have assumed \( Z \) follows any real-valued distribution which is absolutely continuous with respect to the Lebesgue measure. There has been some confusion in the GAN literature about the exact conditions that are required to obtain this result. For example, Goodfellow [8] states the “the only requirements” for \( p(d) \) to have “full support” on \( X \) are that the dimension of \( Z \) be “at least as large as the dimension of \( X \)”, and \( g \) be differentiable. This isn’t accurate as \( Z \) may have smaller dimension, as long as its cardinality is not smaller than the one of \( X \), and the distribution of \( Z \) is non-atomic. Differentiability of \( g \) is not required in theory. To obtain an invertible and differentiable mapping \( g \), the dimension of \( Z \) must not be smaller than the dimension of \( X \). The result in Theorem 2.1 relies on a construction using space-filling curves, which aren’t differentiable.

2.2 GAN Training – Theoretical Case

The GAN approach (Goodfellow et al. [9]) for learning \( g \) is as follows: for \( d \in \mathcal{D} \) and \( g \in \mathcal{G} \) let \( V(d, g) = \mathbb{E}[\log(d(X))] + \mathbb{E}[\log(1 - d(g(Z)))] \).

The relation of \( V(d, g) \) to density ratio estimation (which becomes apparent in equation (4) below) is discussed in [22]. Intuitively, we wish the discriminator \( d(x) \) to be close to 1 if \( x \) is more likely under the distribution of \( X \), and close to 0 if \( x \) is more likely under the distribution of \( g(Z) \). Hence, the optimal \( g \) given a fixed generator \( g \) would attempt to maximize \( V(d, g) \), and the optimal \( g \) is the one which solves the minmax problem

\[
q = \arg \min_{g \in \mathcal{G}} \left\{ \max_{d \in \mathcal{D}} V(d, g) \right\}.
\]

The following theorem, which generalizes Proposition 1 and Theorem 1 in [9], shows that the max and \( \arg \min \) in (2) are well-defined. Note that our formulation neither requires \( g \) to be differentiable, nor \( X \) to be continuous.

Theorem 2.2. Suppose (A1)-(A2) hold. Then

\[
V(d, g) = \int_{\mathcal{D}} \log(d(x))d(x) + \log(1 - d(g(z)))d(z)
\]

for all \( d \in \mathcal{D} \) and \( g \in \mathcal{G} \). Hence, for any fixed \( g \in \mathcal{G} \), any \( d \in \mathcal{D} \) which maximizes \( V(d, g) \) has the form

\[
d(x) = \frac{p(x)}{p(x) + p(\mu)}
\]

for \( \mu \)-almost every \( x \in X \), implying that

\[
\max_{d \in \mathcal{D}} V(d, g) = \int_{\mathcal{D}} \log\left( \frac{p(x)}{p(x) + p(\mu)} \right) d(x)
\]

Assuming that (A3)-(A4) also hold, any generator \( g \in \mathcal{G} \) that minimizies (5) is such that \( p(\mu) = p \)-almost everywhere, and

\[
\min_{g \in \mathcal{G}} \max_{d \in \mathcal{D}} V(d, g) = -\log(4).
\]

The next theorem establishes further properties of the optimal discriminator \( d \) in (4). It generalizes Theorem 2.1 and 2.2 in [11], which were stated for the special case of \( P \) and \( D^{\text{ideal}} \) being not-perfectly-aligned submanifolds of \( \mathbb{R}^d \).
Theorem 2.3. Suppose (A1)-(A3) hold. For fixed $g \in G$, let $P, P^{(i)} \subset X$ be such that $x \in X \setminus \{g(x) \mid x \in P\}$ and $x \in X \setminus \{g(x) \mid x \in P^{(i)}\}$. Suppose $\mu(P \cap P^{(i)}) = 0$, $\rho(P, P^{(i)}) = 0$ (where $\rho$ denotes the topological boundary) and $\mu(P \cap P^{(i)}) = 0$. Then the optimal $d$ in (4) satisfies $\mathbb{P}(d(X) = 1) = 1$ and $\mathbb{P}(d(Y) = 1) = 1$. Moreover, without loss of generality, $d$ is continuous $\mu$-almost everywhere and, in the special case $X = \mathbb{R}^d$, the gradient $\nabla d(x)$ exists and $\nabla d(x) = 0$ for $\mu$-almost every $x \in X$.

In practice, the discriminator being constant on $P$ and $P^{(i)}$ poses problems. In particular, when $g$ is fixed and $d$ is trained till optimality, the gradients $\nabla d(x)$ may vanish and further updates of $g$ become impossible. In their Lemma 1 and 2, Arjovsky and Bottou [1] establish that this $g$ is almost surely going to occur whenever the dimension of $X$ is smaller than the dimension of $Y$, and $y$ is parameterized by a standard neural network. As we show next, it is more directly an inevitable consequence of using an empirical version of the objective (1) in practical GAN training.

2.3 GAN Training – Empirical Case

Let $X^*_n$ be a random variable following the empirical distribution of $X_1, \ldots, X_n$. By (1) we denote the indicator function which evaluates to 1 if the statement in brackets is true, and 0 otherwise.

Note that, conditionally on $X_1, \ldots, X_n$, the distribution of $X^*_n$ in

$$
P(X^*_n \in A \mid X_1, \ldots, X_n) = \frac{1}{n} \sum_{i=1}^n 1_{X_i \in A}
$$

for $A \subset \mathcal{A}$, and an analogous statement holds for the distribution of $g(Z_i)$ conditional on $Z_1, \ldots, Z_n$. It is important to note that practical GAN training (such as in Algorithm 1 in [9]) is not with respect to the theoretical objective (1), but with respect to its empirical counterpart

$$
V_{\text{emp}}(d, g) := \mathbb{E} [\log(d(X^*_n))] \mid X_1, \ldots, X_n
$$

for $d, g \in \mathcal{D}$ and $\mathbb{E}$ denotes the empirical expectation.

It appears there has been a widespread belief among GAN practitioners that optimizing $V_{\text{emp}}(d, g)$ leads to discriminators and generators with the same properties as stated in Theorem 2.3. As the following theorem shows, this isn’t true in general. We add subscript $d_n$ and $g_n$ to emphasize the dependence of discriminator and generator on the sample size $n$.

Theorem 2.4. Suppose (A1)-(A4) hold. For fixed $g \in G$, any $d_n \in \mathcal{D}$ maximizing $V_{\text{emp}}(d_n, g)$ in (6) has the form

$$
d_n(x) = \frac{\sum_{i=1}^n 1_{X_i = x}}{\sum_{i=1}^n 1_{X_i = x} + \sum_{i=1}^n 1_{g(Z_i) = x}}
$$

for $x \in \{X_1, \ldots, X_n \} \cup \{g(Z_1), \ldots, g(Z_n)\}$ (for all other $x \in X$, the value $d_n(x)$ is arbitrary). If the cardinality of $\{Z_1, \ldots, Z_n\}$ is greater than or equal to the cardinality of $\{X_1, \ldots, X_n\}$, then any $g_n \in \mathcal{G}$ minimizing (6) for $d_n$ is such that $g_n(Y(Z_1), \ldots, Y(Z_n)) = X_1, \ldots, X_n$.

Theorem 2.4 reveals two insights: First, if $X$ and $Z$ both have continuous distributions, then $d_n$ has the same properties as $d$ in Theorem 2.3. This suggests the primary reason for vanishing gradients in GAN training is the discrete nature of the empirical objective (6)—which is a remarkably simple explanation.

The second insight is that, when training with respect to (6), there is no theoretical guarantee that $p^{(i)} = p$ $\mu$-almost everywhere for the optimal generator $g_n$ — which contradicts Proposition 2 in [9]. The only guarantee is that, when applied to $Z_1, \ldots, Z_n$, $g_n$ should reproduce the training samples $X_1, \ldots, X_n$. Note: this does not imply that $g_n$ will solely reproduce training samples; in theory, the samples generated on $X_n \setminus \{Z_1, \ldots, Z_n\}$ are arbitrary. Hence, in contrast to the reasoning in [21] and [2], the optimal $g_n$ is not necessarily a Dirac function at the $Z_i$ to which $d_n$ assigns the highest values.

In practice, these undesirable properties could be mitigated for the following reasons: 1) the discriminator and generator function spaces $\mathcal{D}$ and $\mathcal{G}$ have limited capacity, hence the properties of $d_n$ and $g_n$ may hold approximately; 2) similarly, alternate training of the generator and discriminator, or not training till optimality could alter the form of $d_n$ and $g_n$, thereby circumventing pathologies. Limiting the capacity of the networks or finding the tight balance between training the generator and discriminator, however, is challenging. This is why GAN training has been regarded as extremely difficult among practitioners.

2.4 Smoothing the Training Objective

A natural approach to avoid the issues pointed out in Theorem 2.3 and Theorem 2.4 is to smooth the Jensen-Shannon Divergence (JSD) in the GAN training objective by adding noise to the input distributions of the optimal discriminator.

In the following, let $\varepsilon$ be a fixed random variable on $(X, \mathcal{A})$ which is absolutely continuous with respect to $\mu$, hence $x$ has a $\mu$-density $p(x)$. We use the following assumption:

(A5) In addition to (A3), $(X, +)$ is a topological group.

This allows us to consider the convolutions $p \ast p^{(\varepsilon)}$ and $p^{(\varepsilon)} \ast p^{(\varepsilon)}$, which are the $\mu$-densities of $X + \varepsilon$ and $g(Z) + \varepsilon$, respectively. The idea is to use, instead of the discriminator in (4), a modified version

$$
d'_{\varepsilon}(x) = \frac{p \ast p^{(\varepsilon)}(x)}{p \ast p^{(\varepsilon)}(x) + p^{(\varepsilon)} \ast p^{(\varepsilon)}(x)}
$$

If the support of $p^{(\varepsilon)}$ is sufficiently large, then the supports of $p \ast p^{(\varepsilon)}$ and $p^{(\varepsilon)} \ast p^{(\varepsilon)}$ will overlap. Hence, it is not possible to construct an optimal $d'$ with the properties in Theorem 2.3. On the other hand, by the same arguments as in Theorem 2.2, the generator $g$ minimizing (8) is such that $p \ast p^{(\varepsilon)} = p \ast p^{(\varepsilon)}$ $\mu$-almost everywhere, which implies $p \ast p^{(\varepsilon)} = p \ast p^{(\varepsilon)}$ $\mu$-almost everywhere, i.e. the optimal generator $g(Z)$ with respect to the theoretical objective will recover the distribution of $X$.

Next, we derive the form of the optimal discriminator for the modified empirical objective.

Theorem 2.5. Suppose (A1)-(A5) hold and let $g \in G$ be fixed. If we replace $X^*_n$ and $g(Z_i)$ in (6) by $X^*_n \ast \varepsilon$ and $g(Z_i) \ast \varepsilon$, respectively, then the discriminator minimizing the objective has the form

$$
d'_{\varepsilon}(x) = \frac{\sum_{i=1}^n p^{(\varepsilon)}(x - X_i)}{\sum_{i=1}^n p^{(\varepsilon)}(x - X_i) + \sum_{i=1}^n p^{(\varepsilon)}(x - g(Z_i))}
$$

for $x \in X$. Same as in Theorem 2.4, if the cardinality of $\{Z_1, \ldots, Z_n\}$ is greater than or equal to the cardinality of $\{X_1, \ldots, X_n\}$, then any $g_n \in \mathcal{G}$ minimizing the objective (6) for $d'_{\varepsilon}$ is such that $g_n(Z_1, \ldots, g(Z_n)) = X_1, \ldots, X_n$.

Note that the smoothing of distributions outlined here is not equivalent to adding noise to the samples $X_1, \ldots, X_n$ or $g(Z_1), \ldots, g(Z_n)$ before optimizing the empirical objective, which would lead to the same result as in (7).

As Theorem 2.5 shows, smoothing the empirical distributions $X^*_n$ and $g(Z_i)$ results in an optimal discriminator $d'_{\varepsilon}$ which, if the support of $p^{(\varepsilon)}$ is sufficiently large, won’t cause vanishing gradients. However, there is still no guarantee that the optimal generator $g(Z)$ recovers the distribution of $X$ apart from reproducing training samples. In the following section we discuss a new training objective which addresses this issue.

2.5 Kernel GANs

Throughout the rest of the paper we assume $X = \mathbb{R}^d$, $\mu$ is absolutely continuous with respect to the Lebesgue measure on $\mathbb{R}^d$, $\mathcal{A}$ is the Borel $\sigma$-algebra, and $\mathcal{G}$ is a bounded and square-integrable function (kernel)
σ > 0 (bandwidth). For x ∈ X consider the kernel density estimates

\begin{align}
\hat{p}_\theta(x) &= \frac{1}{\sigma^d n} \sum_{i=1}^n K \left( \frac{x - X_i}{\sigma} \right) \\
\hat{p}_\theta^D(x) &= \frac{1}{\sigma^d n} \sum_{i=1}^n K \left( \frac{x - \hat{g}_\theta(Z_i)}{\sigma} \right)
\end{align}

of p(x) and p(φ(z)). Choosing K = ρ(·,·), we can regard (9) as a kernel estimate of the density ratio p(x)/p(φ(z) + p(φ(z)). Our key idea is to plug the optimal discriminator (9) back into the empirical training objective (6), i.e., consider \( V_\lambda (d, q) \) with \( d = d^\ast \). This results in the Kernel GAN training objective:

\begin{align}
K_G(\theta, \sigma, \varphi) := \int p(x) \log \left( \frac{\hat{p}_\theta(x) + \varphi}{\hat{p}_\theta(x) + \hat{p}_\theta^D(x) + 2\varphi} \right) + \frac{1}{n} \sum_{i=1}^n \log \left( \frac{\hat{p}_\theta^D(\hat{g}_\theta(Z_i)) + \varphi}{\hat{p}_\theta^D(\hat{g}_\theta(Z_i)) + \hat{p}_\theta(\hat{g}_\theta(Z_i)) + 2\varphi} \right)
\end{align}

where \( \varphi \geq 0 \) is a regularizer to avoid underflow issues. In contrast to conventional GAN training, only the generator is explicitly updated when optimizing (12). The discriminator \( \delta \) is updated implicitly through changes in the density estimates (11). Note that plugging the optimal discriminator (9) into the training objective (6) can be regarded as unravelling the discriminator same as in Metz et al. [21], where in our case the unravelling is controlled explicit-training.

The following theorem establishes convergence of the objective (12).

**Theorem 2.6.** Suppose (A1)-(A5) hold. Moreover, suppose \( p \) and \( p(\varphi) \) are bounded and uniformly continuous for all \( \theta \in \Theta \), and K has compact support and is of the form \( K(x) = \phi(q(x)) \), where \( q \) is a polynomial and \( \alpha \) a bounded non-negative function with bounded variation. Let \( \sigma_0 > 0 \) be a sequence asymptotically equivalent to \( C n^{\frac{1}{d+1}} \) for some finite constant \( C \) and \( \delta \in (0, 1) \).

Then

\[ \lim_{n \to \infty} K_G(\theta, \sigma, \varphi) = \int p(x) \log \left( \frac{p(x) + \varphi}{p(x) + p(\varphi) + 2\varphi} \right) + \frac{1}{n} \sum_{i=1}^n \log \left( \frac{p(\varphi) + \varphi}{p(x) + p(\varphi) + 2\varphi} \right) \] (13)

\( P \)-almost surely for all \( \theta \) and \( \varphi > 0 \).

The regularizer \( \varphi > 0 \) is required for establishing the convergence in (13). It results in estimates of the theoretical RSD that are asymptotically biased. In particular, while \( K_G(\theta, \sigma, \varphi) \) converges to \( -\log(4) \) (which is the minimum value of JSD) if \( \varphi \) is such that \( p(\varphi) = p \), it may converge to smaller values for other values of \( \theta \). Hence, minimizing \( K_G(\theta, \sigma, \varphi) \) would not result in a generator \( p(\varphi)^\ast(Z) \) recovering \( X \) (although \( \varphi \) can be chosen arbitrarily small, hence the practical difference might be negligible). However, as we show in Appendix A.2, if \( P(X) < \infty \), then \( K_G(\theta, \sigma, \varphi) \) can be modified such that its limit is minimized by \( \theta \) recovering the distribution of \( X \).

### 3 EXPERIMENTS

In this section, we discuss practical learning of Kernel GANs. First, we demonstrate the learning on small and mid-sized datasets – a Mixture-of-Gaussian (MOG) toy dataset (212) and MNIST (116). Further, we study the effect of kernel bandwidth along with practical approaches such as generating in a lower-dimensional feature space that is independently learned using an autoencoder.

Second, we establish practical usefulness of Kernel GANs by scaling them to two high-dimensional datasets: CIFAR-10 ([151]) and CelebA ([203]). We enable this with a modified training setup that involves kernel learning, similar to [18].

---

Figure 1: MOG toy dataset. Blue: training points. Green: samples produced by the generator. The leftmost figure is for the initial generator. Training phases for bandwidths (left to right): 0.8, 0.4, 0.2, 0.1, 0.05, 0.025 (10,000 iterations were performed for each σ).

Finally, we conduct various evaluations of the performance of the trained generators. In a quantitative evaluation, we compare Kernel GANs with MMD-based models (Li et al. [19], Dziugaite et al. [6], Li et al. [18]), which also use kernel-based statistics, but in a non-adversarial fashion. Full details on the implementation and all the experiments can be found in Appendix A.3 and A.4.

### 3.1 Learning Kernel GANs

Algorithm 1 outlines our general training protocol for learning the generator parameters \( \theta \) that minimize the training objective (12).

**Algorithm 1 Training Protocol**

**Input:** Training samples \( X \), distribution of latent variable \( Z \), initial kernel parameters \( \theta \), initial generator parameters \( \theta \), regularizer \( \psi \).

**1:** while stopping criterion not met do

**2:** Sample a minibatch \( X_1, \ldots, X_n \) from \( X \).

**3:** Generate iid samples \( Z_1, \ldots, Z_n \) from \( Z \).

**4:** Generate iid samples \( Z_1, \ldots, Z_n \) from \( Z \).

**5:** Update generator parameters \( \theta \) according to gradients \( \nabla_\theta K_G(\theta, \sigma, \varphi) \).

**6:** Update kernel parameters \( \sigma \).

**7:** end while

**Output:** Trained generator parameters \( \theta \).

**Hyperparameters.** Previously, Li et al. [19] and Dziugaite et al. [6] used RBF kernels in their training objectives for generative models. While [19] deploys a mixture of RBF kernels, [6] uses a Bayesian Optimisation to determine a suitable bandwidth. Moreover, both works suggest to use the median-trick (Gretton et al. [11]) as a method to choose kernel bandwidths for computing MMD statistics. Intuitively, small bandwidths push the generator towards producing samples that are similar to the training set. However, initial bandwidths that are too small will not give gradients in areas that are far from the modes of the training set. We therefore explored gradual reductions of the bandwidth during training, similar to cooling schedules in simulated annealing (e.g., Hjäck [13], Nourani and Andreesen [23]). While the regularizer \( \varphi \) is required for deriving Theorem 2.6, we don’t find it to play a crucial role in the practical experiments and therefore set it equal to zero. A further investigation of the practical effect of \( \varphi \) will be part of future work.

**MOG Toy Dataset.** For the MOG dataset, \( Z \) was a 100-dimensional standard normal distribution, and the generator used was a three-layer fully connected network (128-relu-128-relu-128-tanh).

Figure 1 shows the evolution of the generator during the training, as the bandwidth \( \sigma \) is gradually decreased. Initially the generated samples \( q(Z) \) are dispersed randomly. As the bandwidth \( \sigma \) is decreased, they begin to concentrate around the modes of the MOG distribution.

**MNIST.** We successfully trained three different generative models for MNIST. Two of these were trained to sample directly in the space of (28 × 28) greyscale images. The third model used an autoencoder to map the images onto a lower-dimensional feature space, in which the generator was trained.

The three models used following architectures: a fully connected network (FC); a deconvolutional network with batch normalisation (DC); a fully connected network for the feature space (FC-PS).
Augmenting Models

Table 1: Quantitative evaluation of different generators on MNIST

<table>
<thead>
<tr>
<th></th>
<th>FC</th>
<th>DC</th>
<th>FC-FS</th>
<th>GMMN-AE</th>
<th>GMMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE</td>
<td>0.408</td>
<td>0.289</td>
<td>0.365</td>
<td>0.361</td>
<td>0.293</td>
</tr>
<tr>
<td>MMD</td>
<td>0.019</td>
<td>0.017</td>
<td>0.084</td>
<td>0.091</td>
<td>0.005</td>
</tr>
<tr>
<td>JSD</td>
<td>6.601</td>
<td>7.464</td>
<td>6.916</td>
<td>6.948</td>
<td>7.408</td>
</tr>
<tr>
<td>JSD-F</td>
<td>-0.740</td>
<td>-0.740</td>
<td>-0.769</td>
<td>-0.776</td>
<td>-0.689</td>
</tr>
<tr>
<td>JSD-S</td>
<td>0.644</td>
<td>-0.645</td>
<td>-0.603</td>
<td>-0.596</td>
<td>-0.694</td>
</tr>
<tr>
<td>MMD</td>
<td></td>
<td></td>
<td></td>
<td>0.011</td>
<td>0.017</td>
</tr>
</tbody>
</table>

Figure 2: Training of generators for MNIST. (a): Fully connected network (FC). (b): Deep convolutional architecture (DC). (c): Fully connected network in feature space (FC-FS).

Figure 3: Kernel GAN generated samples. (a): CIFAR10. (b): CelebA.

We adopted the architectures proposed in [19] for FC and FC-FS, and the architecture proposed in [24] for DC. As latent variable \( Z \), all models used samples from a 10-dimensional uniform distribution. We used a mixture of RBF kernels for training these models (see the appendix for details). For FC-FS, we also experimented with different bandwidths in simple RBF kernels. Generated samples from FC, DC and FC-FS are shown in Figure 2 (a)-(c). We found that samples from FC-FS had a very smooth appearance. DC generated sharper samples than FC, but still produced some artifacts. The sharpness of the FC-FS samples with simple RBF kernels and different bandwidths is evaluated in the appendix. Quantitative measures of sample fidelity and diversity are discussed below.

We observed that training randomly initialized networks can be numerically unstable for very small bandwidths, leading to artifacts in the produced images. For very large bandwidths, we occasionally found the generator to collapse and produce undersampled images like mean images. We noticed, however, that the generator model recovered when we increased or decreased the bandwidth appropriately in subsequent training iterations. An analysis is provided in the appendix. This suggests that kernel bandwidths can be used as “knobs” for correcting over- or underfitting of generative models during the training process.

3.2 Scaling Kernel GANs

Although being sufficient in theory, we found it difficult to train Kernel GANs for colored images using plain RBF kernels. To impose more structure in kernel-based training of generative models, [19] had suggested to use convolutional autoencoders to learn a lower dimensional feature space for colored images. [18] used this approach for training GMMN models on colored images both in feature and data space, however they observed quality issues in the generated samples. Instead they pursued an alternative approach where, much like a GAN setup, they learn a network which transforms the original space into a lower dimensional space over which the kernel is computed. We adopt their approach, leading to a modified Kernel GAN training objective, \( \mathcal{L}_{\psi, \theta, \sigma, \phi} \), which is the same as (12), except that the kernels operate on the space \( f_{\psi}(X) \) instead of \( X \).

\[
\mathcal{L}_{\psi, \theta, \sigma, \phi}(z) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{K} \left( f_{\sigma}(x_i) - f_{\sigma}(y(z_i)) \right),
\]

\[
\mathcal{L}_{\psi, \theta, \sigma, \phi}(z) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{K} \left( f_{\psi}(x_i) - f_{\psi}(y(z_i)) \right).
\]

The parameters \( \{\theta, \psi\} \) are learned in a min-max fashion: \( \min_{\{\theta, \psi\}} \max_{\phi} \mathcal{L}_{\psi, \theta, \sigma, \phi} \). Similar to conventional GAN training, \( \psi \) and \( \theta \) are optimized alternately. In practice, we observed that additional regularization was required for stable learning. We used the experimental setup of [18], which models the function \( f_{\psi} \) as the encoder of an autoencoder and regularizes the objective function with the autoencoder reconstruction loss.

CIFAR10 and CelebA. We used this setup to successfully train Kernel GANs for the CIFAR10 and CelebA datasets. We adopted and appropriately rescaled hyperparameters and regularization weights of [18]. We trained a Deep Convolutional architecture for both datasets. The dimension of the encoded space \( f_{\psi}(X) \) was fixed to 100. While CIFAR10 was trained with a 128-dimensional standard normal distribution for \( Z \), CelebA was trained with a 64-dimensional \( Z \). Samples obtained from the trained generators are shown in Figure 3. We found that they were qualitatively comparable to the results in [18].

3.3 Quantitative Evaluation

MNIST. Quantifying the performance of generative networks – particularly their ability to generalize and produce diverse samples – remains a challenging task ([28, 31]). In this paper, we report the following metrics:

- Expected entropy (EE): As proposed in [26], we trained a probabilistic classifier (LeNet [17]) and computed the expected entropy of the classifier probabilities for samples \( g(Z) \). For all metrics, we used Monte-Carlo estimates of expected values, based on 10,000 samples from \( g(Z) \). Expected nearest-neighbour distance (ENN): To assess the similarity of generated samples with samples in the training set, we determined the expected value of the Euclidean distance between samples from \( g(Z) \) and their nearest neighbor in the train set. LeNet score (LS): Similar to the Inception score proposed in [26], we computed the exponential of the expected Kullback-Leibler divergence between the predicted class probabilities for samples \( g(Z) \), and the frequency of classes (±digits) in the MNIST train set. Jensen-Shannon divergence (JSD): We estimated the JSD between the unknown data distributions and \( g(Z) \) by computing (12) over the MNIST test set and samples produced by \( g(Z) \).

We also report the corresponding values JSD-F and JSD-S of the first and second term in (12). Maximum Mean Discrepancy (MMD): Finally, we also report the MMD statistic ([11]).
Table 1 shows a comparison of different generators. GMMN and GMMN-AE are the data- and code-space Generative Moment Matching Networks (GMMN) proposed in [19]. The numbers in the MNIST Test column are obtained by using the MNIST test set instead of generated samples; hence they can be regarded as the performance of an ideal generator, with optimal trade-off between fidelity (EE), diversity (ENN, LS), and overall consistency (JSD). In this regard, we found DC performed the best among all trained generators: it achieved the lowest EE, comparable ENN, and the closest LS in comparison with MNIST Test. FC-FS and GMMN-AE also achieved high fidelity, but seemed to exhibit less diversity. Interestingly, the first and second term of the JSD were observed the closest LS in comparison with MNIST Test performed the best among all trained generators: it achieved the lowest EE, comparable ENN, and the closest LS in comparison with MNIST Test. FC-FS and GMMN-AE also achieved high fidelity, but seemed to exhibit less diversity. Interestingly, the first and second term of the JSD were observed. We hypothesize that keeping JSD-F and JSD-S balanced during training is key to obtaining generators with good generalization capacity.

CIFAR10. We computed the Inception score [26] and mean and standard deviation for 5×10k samples obtained from a Kernel GAN that was trained for 5,000 iterations. The score for held-out CIFAR10 images (which can be regarded as gold standard) was 11.95 ± 0.20. Kernel GANs yielded a score of 4.22 ± 0.12, which is significantly higher than the scores for GMMN-AE and GMMN (3.94 ± 0.04 and 3.47 ± 0.03, respectively), but lower than for MMD-GAN (6.17 ± 0.07, see [18]). The latter finding can be explained by the fact that we did not optimize hyperparameters and regularization weights for Kernel GANs, which could lead to further improvements in future work.

4 CONCLUSIONS
We established a rigorous framework for analyzing statistical properties of Generative Adversarial Network training. To overcome potential pathologies (in particular, vanishing gradients), we introduced a novel training objective, which can be regarded as minimizing a non-parametric estimate of the Jensen-Shannon Divergence. We analyzed its asymptotic properties and showed its practical applicability. We see several directions for future work: 1) Advance the design of optimal kernels and strategies for annealing the bandwidths. 2) Further analyze statistical properties of the proposed training objective, in particular, the effect of the regularizer. 3) Investigate the effect of imbalances between the first and second term in the training objective; we believe this could lead to the design of adaptive training protocols which ensure both fidelity and diversity of generator samples.

References
A.1: Proofs

Proof of Theorem 2.1: This is an immediate consequence of Theorem 1 in Schoenfeld [27], which builds on the classical result from Hahn-Mazurkiewicz that a metric space is the continuous image of the unit interval if and only if the space is compact, connected and locally connected.

Proof of Theorem 2.2: The arguments are analogous to the proofs of Proposition 1 and Theorem 1 in Goodfellow et al. [9], using the change-of-variables formula for general pushforward measures (Bogachev [4]).

Proof of Theorem 2.3: This is an immediate consequence of Urysohn’s Lemma (Section 15 in Willard [30], and the assumption that $X$ is a compact Hausdorff space.

Proof of Theorem 2.4: We first note that
\[
V_d(d,g) = \int_X \log(d(x)) p_n(x) + \log(1-d(x)) \rho_{\theta}^{(1)}(x) dx,
\]
where $\mu$ denotes the counting measure on $(X, \mathcal{A})$, and $p_n$ and $\rho_{\theta}^{(1)}$ the $\mu$-densities of $X_n$ conditional on $X_1, \ldots, X_n$ and of $\theta(Z_n)$ conditional on $Z_1, \ldots, Z_n$, respectively:
\[
p_n(x) = \frac{1}{n} \sum_{i=1}^n I[Z_i = x] \quad \text{and} \quad \rho_{\theta}^{(1)}(x) = \frac{1}{n} \sum_{i=1}^n I[\theta(Z_i) = x]
\]
for $x \in X$. Hence, completely analogous to Theorem 2.2, we obtain that any $d_n \in D$ maximizing $V_d(d_n, g)$ has the form
\[
d_n(x) = \frac{p_n(x)}{\hat{p}_n(x)} + \hat{p}_n(x)
\]
for $x \in \{X_1, \ldots, X_n \} \cup \{\theta(Z_1), \ldots, \theta(Z_n)\}$, and hence the result in (7) follows. Again completely analogous to Theorem 2.2, any generator $\rho_{\theta}(\cdot, \cdot)$ minimizing the objective (6) for $d = d_n$ is such that $p_n(x) = \rho_{\theta}^{(1)}(x)$ for all $x \in \{X_1, \ldots, X_n \} \cup \{\theta(Z_1), \ldots, \theta(Z_n)\}$, which holds if and only if $(\theta(Z_1), \ldots, \theta(Z_n)) = (X_1, \ldots, X_n)$.

Proof of Theorem 2.5: We note that $X_n + \epsilon$ conditional on $X_1, \ldots, X_n$ and $\theta(Z_n) + \epsilon$ conditional on $Z_1, \ldots, Z_n$ have the following $\mu$-densities:
\[
(p_n(x) + \rho_{\theta}^{(1)}(x)) = \frac{1}{n} \sum_{i=1}^n \rho_{\theta}^{(1)}[x - X_i] \quad \text{and} \quad \rho_{\theta}^{(2)}(x) + \rho_{\theta}^{(3)}(x) = \frac{1}{n} \sum_{i=1}^n \rho_{\theta}^{(2)}(x - y_i)
\]
for $x \in X$, with $p_n$ and $\rho_{\theta}^{(1)}$ as in (16). Using the same arguments as in the proof of Theorem 2.4, we obtain both statements of this theorem.

Proof of Theorem 2.6: Let $\theta$ and $\phi$ be fixed. First we note that the kernel $K$, the densities $p_{\theta}^{(2)}$ and the sequence $\sigma_n$ satisfy conditions (K2), (D2), (W2) in Giné and Guillou [7]. Consequently, by Theorem 3.3 in [7],
\[
\lim_{n \to \infty} \sup_{x \in X} |\hat{p}_{\sigma_n}(x) - p_{\sigma_n}(x)| = 0
\]
$\mathbb{P}$-almost surely. Since $p$ is uniformly continuous and $K$ has compact support, it is easy to obtain from equation (1.3) in [7],
\[
\lim_{n \to \infty} \sup_{x \in X} |\mathbb{E}[\hat{p}_{\sigma_n}(x)] - p(x)| = 0.
\]

$\mathbb{P}$-almost surely. Consequently, by the triangle inequality,
\[
\lim_{n \to \infty} \sup_{x \in X} |\hat{p}_{\sigma_n}(x) - p(x)| = 0.
\]

A.2: Asymptotically unbiased estimation of $p$

In order to establish (i), introduce the following function of $\theta$ and $\phi$,
\[
K(\theta, \phi) = \int \left( p(x) + \rho_{\theta}^{(1)}(x) \right) \log \left( p(x) + \rho_{\theta}^{(1)}(x) + 2\rho_{\theta}^{(3)}(x) \right) dx.
\]
Note that $K(\theta, \phi)$ is the Jensen-Shannon Divergence (multiplied by $2$) and
\[
\rho_{\theta}^{(3)}(x) := 1 + \rho_{\theta}^{(1)}(x) + \rho_{\theta}^{(2)}(x).
\]

Hence, $K(\theta, \phi)$ is minimized for $\theta$ such that $\rho_{\theta}^{(2)} = \rho_{\theta}^{(3)}$-almost everywhere, which is equivalent to $\rho_{\theta}^{(2)} = p_{\theta}$-almost everywhere. Next, observe that $K(\theta, \phi)$ is equal to the right-hand side in (13), plus the following two terms:
\[
K^{(1)}(\theta, \phi) := \int \log \frac{p(x)}{p(x) + \rho_{\theta}^{(1)}(x) + 2\rho_{\theta}^{(3)}(x)} dx
\]
and
\[
K^{(2)}(\theta, \phi) := \int \log \frac{p(x) + \rho_{\theta}^{(1)}(x) + 2\rho_{\theta}^{(3)}(x)}{p(x) + \rho_{\theta}^{(2)}(x) + 2\rho_{\theta}^{(3)}(x)} dx.
\]

Now suppose $\hat{X}_n, \ldots, \hat{X}_n$ are sampled independently according to the probability density $x \mapsto p(x)/\mu(X)$, and consider the estimators
\[
K^{(1)}(\theta, \sigma_n, \phi) := \frac{1}{n} \sum_{i=1}^n \log \frac{\hat{p}(X_i) + \phi}{\hat{p}(X_i) + \phi + 2\rho_{\theta}^{(3)}(X_i) + 2\rho_{\theta}^{(1)}(X_i) + 2\rho_{\theta}^{(3)}(X_i)}
\]
and
\[
K^{(2)}(\theta, \sigma_n, \phi) := \frac{1}{n} \sum_{i=1}^n \log \frac{\hat{p}(X_i) + \phi + 2\rho_{\theta}^{(3)}(X_i) + 2\rho_{\theta}^{(1)}(X_i) + 2\rho_{\theta}^{(3)}(X_i)}{\hat{p}(X_i) + \phi + 2\rho_{\theta}^{(2)}(X_i) + 2\rho_{\theta}^{(3)}(X_i)}.
\]

Under the same assumptions as in Theorem 2.6, we obtain
\[
\lim_{n \to \infty} \mathbb{E}[K^{(1)}(\theta, \sigma_n, \phi)] = K^{(1)}(\theta, \phi)
\]
$\mathbb{P}$-almost surely. Hence, $K^{(1)}(\theta, \sigma_n, \phi) + K^{(2)}(\theta, \sigma_n, \phi)$ converges to a limit which is minimized by $\theta$ such that the generator $p^{(2)}(Z)$ recovers $X$. 

P-almost surely. Now let $\alpha$ denote a finite upper bound both for $p$ and $\mu(\theta)$ (which exists by our assumptions). Note that $(\phi, \psi) \mapsto \log(\phi(x) + \psi(x) + 2\alpha)$ is uniformly continuous on $[0, \alpha] \times [0, \alpha]$.

Hence, with (18) and (19), we obtain
\[
\lim_{n \to \infty} \sup_{x \in X} \log \frac{\hat{p}_{\sigma_n}(x) + \phi + 2\alpha}{\hat{p}_{\sigma_n}(x) + \phi + 2\alpha + 2\rho_{\theta}^{(3)}(x) + 2\rho_{\theta}^{(1)}(x) + 2\rho_{\theta}^{(3)}(x) + 2\alpha} = 0
\]
$\mathbb{P}$-almost surely. Consequently,
\[
\lim_{n \to \infty} \sup_{x \in X} \log \frac{\hat{p}_{\sigma_n}(x) + \phi + 2\alpha}{\hat{p}_{\sigma_n}(x) + \phi + 2\alpha + 2\rho_{\theta}^{(3)}(x) + 2\rho_{\theta}^{(1)}(x) + 2\rho_{\theta}^{(3)}(x) + 2\alpha} = 0
\]
$\mathbb{P}$-almost surely. Moreover, by the Strong Law of Large Numbers,
\[
\lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^n \log \frac{p(X_i) + \phi + 2\alpha}{p(x) + \rho_{\theta}^{(3)}(x) + 2\rho_{\theta}^{(1)}(x) + 2\alpha} = 0
\]
$\mathbb{P}$-almost surely. Hence, using (19), (20) and the triangle inequality, we obtain
\[
\lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^n \log \frac{\hat{p}_{\sigma_n}(x) + \phi + 2\alpha}{\hat{p}_{\sigma_n}(x) + \phi + 2\alpha + 2\rho_{\theta}^{(3)}(x) + 2\rho_{\theta}^{(1)}(x) + 2\rho_{\theta}^{(3)}(x) + 2\alpha} = 0
\]
$\mathbb{P}$-almost surely which is the desired result for the first term in (13). The result for the second term follows by analogous arguments, which proves the theorem.
A.3: Implementation details

MNIST Autoencoder. The autoencoder for Feature Space (FS) based Kernel GANs of MNIST were trained to yield 32-dimensional feature vectors of images, as suggested in [19]. The architecture was was

$\text{FC-10-64-relu-256-relu-256-relu-1024-relu-784-sigmoid}(1)$,

Kernel: mixture of RBF ($\sigma \in \{0.05, 0.5, 1.0, 0.05, 0.1, 0.05, 0.01\}$).

$\text{FC-FS-10-64-relu-256-relu-32-sigmoid}(19)$

Kernel: mixture of RBF ($\sigma \in \{0.0, 0.5, 0.1, 0.05, 0.01\}$).

DC: the DCGAN ([25]) architecture with dimension as 64.

Kernel: mixture of RBF ($\sigma \in \{0.05, 0.5, 1.0, 0.05, 0.1, 0.05, 0.01\}$).

Training was performed with a minibatch size of $n = 1,000$ (cf Algorithm 1), and RMSProp [29] (learning rate of 0.001) for optimization.

Classifier for LeNet score. A LeNet-like classifier was trained with the following architecture: $(28,28,1)$-conv$(32,(3,3))$-relu-maxpool$(2,2)$-conv$(64,(3,3))$-relu-maxpool$(2,2)$-fc$(128)$-relu-10-maxout. Training was performed using Adam and with dropout for regularization.

Generative Moment Matching Networks. We trained the data-space and code-space networks of [19], which they define as GMMN and GMMN-ALLI respectively. We used a mixture of RBF kernels (GMMN: $\sigma \in \{100.0, 50.0, 10.0, 5.0, 1.0, 0.5, 0.1, 0.05, 0.01\}$). GMMN-ALLI: $\sigma \in \{1.0, 0.5, 0.1, 0.05, 0.01\}$.

CIFAR10 and CelebA. We trained models based on DCGAN ([25]) architecture for both CIFAR10 and CelebA. Similar to [18], we trained a network for $f$ which was modelled as an encoder of a convolutional autoencoder. The encoding dimension was fixed to 100. A mixture of RBF kernels ($\sigma \in \{1.0, 2.0, 10.0, 10.0, 5.0, 1.0, 0.5, 0.1, 0.05, 0.01\}$). GMMN-ALLI: $\sigma \in \{1.0, 0.5, 0.1, 0.05, 0.01\}$.

Hyperparameters including the model architecture, values of kernel bandwidth were not optimised for any of the experiments. Further, we also believe that longer schedules for optimisation can affect the performance of generators.

A.4: Experiments

MNIST generated samples. Figure 4–7 show the generator samples for MNIST. For comparison, we include a sample from the MNIST test set in Figure 8.

Effect of the kernel bandwidth. We conducted experiments with the FC-FS architecture and simple RBF kernels with different bandwidths. Figure 7 shows samples generated for the MNIST dataset. As can be seen, the sample quality is comparable to FC-FS; however, some digits appear to be over- and under-represented, respectively. Table 2 shows the EE and EEN metrics for on the MNIST dataset for different bandwidths. We note that smaller bandwidths result in generator samples with lower EE. Below we show EIL values for samples generated over 2-dimensional manifolds in the latent space, illustrating that lower EE stems both from generated samples with higher visual fidelity, and sharper transitions between low-entropy regions in the latent space. Similarly, also the EEN values decrease with the bandwidth. This can be regarded as a loss of diversity, as generated samples become more and more similar to instances in the training set.

We observed that, for very large values of the kernel bandwidth $\sigma$, the generated samples have a tendency to collapse to the mean of the training instances. Figure 9 analyzes the behaviour of the training objective in (12) as a function of $\sigma$. We compare two different generators: an “ideal” one (black line), which is able – given 100 samples from the MNIST training set, to produce 100 different samples. The red line shows objective function values of a generator which simply produces the mean of the 100 training samples. As can be seen in (a), for large values of $\sigma$ the latter generator performs better with regard to the training objective. For small $\sigma$, the order is reverse. Interestingly, for large $\sigma$ the objective function values of the generator producing the mean is below $-\log(\alpha)$. As (b) and (c) show, this is due to a large imbalance of the first and second term in the training objective. Hence, we hypothesize that the training of meaningful generators should not only aim to minimize (12), but also aims to keep the first and second term in (12) balanced. A deeper investigation of the trade-off between these two terms will be future work.

Entropy carpets. Given a trained GAN $g$ and points $z_1, z_2$ in the latent space $Z$, it is often instructive to inspect the generated images $g((1−x)z_1 + x z_2)$ for $x \in [0, 1]$. Recently, Dinh et al. [5] proposed an angle-based 2-dimensional manifold interpolation between four points $z_1, z_2, z_3, z_4 \in Z$. Here we explore more conventional convex combinations, given by

$$\{x \cdot y \cdot z_1 + (1−x) \cdot y \cdot z_2 + x \cdot (1−y) \cdot z_3 + (1−y) \cdot z_4 : x, y \in [0, 1]\}.$$ 

In practice, we let $x$ and $y$ vary along a mesh grid of a unit-length square. Figure 10–12 display the image manifolds generated by three different models, along with the “entropy carpets” which show the entropy of a probablistic classifier (we use LeNet, see above) at each point of the manifold. Bright colors correspond to high-entropy regions, dark colors to low entropy. Entropy carpets can be regarded as a semi-qualitative-semi-quantitative way to relate the manifolds of generated images to the Expected Entropy (EE) metric reported in Tables 2 and 1. Of particular interest are the transitions which coincide with high entropy. Typically, the output of the generator is uninterpretable in those regions. Hence, an ideal generator should have as few and as sharp transitions between different modes as possible.

| Table 2: Effect of the kernel bandwidth $\sigma$ on the FC-FS model |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| $\sigma$       | $\sigma = 5.0$  | $\sigma = 1.0$  | $\sigma = 0.5$  | $\sigma = 0.1$  | $\sigma = 0.05$ | $\sigma = 0.01$ |
| EE              | 2.117           | 1.388           | 1.096           | 1.079           | 0.783           | 0.406           |
Figures 13-15 show the entropy carpets for the FC-FS model trained with bandwidths $\sigma = 0.5, 0.05, 0.01$. Beyond the Expected Entropy metrics reported in Tables 2, the entropy carpets give an idea of the fraction of points $Z$ that result in meaningful versus non-meaningful images. Interestingly, even for $\sigma = 0.5$, significant parts of $Z$ result in high-fidelity images; however, there are large areas in between the modes in which the generator only generates "noise" from the point-of-view of the classifier.

Figure 13: Entropy carpet for $\text{FC-FS-RBF}(0.5)$
Figure 14: Entropy carpet for $\text{FC-FS-RBF}(0.05)$
Figure 15: Entropy carpet for $\text{FC-FS-RBF}(0.01)$

Figures 13-15 show the entropy carpets for the FC-FS model trained with bandwidths $\sigma = 0.5, 0.05, 0.01$. Beyond the Expected Entropy metrics reported in Tables 2, the entropy carpets give an idea of the fraction of points $Z$ that result in meaningful versus non-meaningful images. Interestingly, even for $\sigma = 0.5$, significant parts of $Z$ result in high-fidelity images; however, there are large areas in between the modes in which the generator only generates "noise" from the point-of-view of the classifier.
Learning, Knowledge and Reasoning
"A wise man, therefore, proportions his belief to the evidence."

David Hume

The next generation of AI systems will need to understand and reason about the world in human-like ways. Deep reasoning over complex, often implicit, knowledge is beyond the capabilities of today’s most advanced machine learning classification systems. In order to achieve human level interpretability, a machine needs to accrete and represent common sense knowledge and reason on that knowledge. In other words, a machine needs a knowledge base – a set of facts – and a means to deduce relevant consequences of these facts. In order to do that, knowledge must be first represented in a formal language that machines can understand.

This chapter highlights our papers on learning, knowledge induction and reasoning, including: identification of implicit knowledge between entities and minimization of time needed for domain adaptation [9], a semantic concept discovery engine to assist analysts in preparing deep analysis reports [10], a sophisticated question annotation and information retrieval interface for middle- and high-school science exams from the US educational system [11], and a visual question answering system that combines deep learning and symbolic reasoning [12].

9 — Discovering Implicit Knowledge with Unary Relations
p — 138

10 — Semantic Concept Discovery over Event Databases
p — 150

11 — A Systematic Classification of Knowledge, Reasoning, and Context within the ARC Dataset
p — 168

12 — Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding
p — 180
Discovering Implicit Knowledge with Unary Relations

Michael Glass
IBM Research AI
Knowledge Induction and Reasoning
mrglass@us.ibm.com

Alfio Gliozzo
IBM Research AI
Knowledge Induction and Reasoning
gliozzo@us.ibm.com

Abstract

State-of-the-art relation extraction approaches are only able to recognize relationships between mentions of entity arguments stated explicitly in the text and typically localized to the same sentence. However, the vast majority of relations are either implicit or not sententially localized. This is a major problem for Knowledge Base Population, severely limiting recall. In this paper we propose a new methodology to identify relations between two entities, consisting of detecting a very large number of unary relations, and using them to infer missing entities. We describe a deep learning architecture able to learn thousands of such relations very efficiently by using a common deep learning based representation. Our approach largely outperforms state of the art relation extraction technology on a newly introduced web scale knowledge base population benchmark, that we release to the research community.

1 Introduction

Knowledge Base Population (KBP) from text is the problem of extracting relations between entities with respect to a given schema, usually defined by a set of types and relations. The facts added to the KB are triples, consisting of two entities connected by a relation. Although providing explicit provenance for the triples is often a sub-goal in KBP, we focus on the case where correct triples are gathered from text without necessarily annotating any particular text with a relation. Humans are able to perform very well on the task of understanding relations in text. For example, if the target relation is presidentOf, anyone will be able to detect an occurrence of this relation between the entities TRUMP and UNITED STATES from both the sentences “Trump issued a presidential memorandum for the United States” and “The Houston Astros will visit President Donald Trump and the White House on Monday”. However, the first example expresses an explicit relation between the two entities, while the second states the same relation implicitly and requires some background knowledge and inference to identify it properly. In fact, the entity UNITED STATES is not even mentioned explicitly in the text, and it is up to the reader to recall that US presidents live in the White House, and therefore people visiting it are visiting the US president.

Very often, relations expressed in text are implicit. This reflects in the low recall of the current KBP relation extraction methods, that are mostly based on recognizing lexical-syntactic connections between two entities within the same sentence. The state-of-the-art systems are affected by very low performance, close to 16.6% F1, as shown in the latest TAC-KBP evaluation campaigns and in the open KBP evaluation benchmark. Existing approaches to dealing with implicit information such as textual entailment depend on unsolved problems like inducing entailment rules from text.

In this paper, we address the problem of identifying implicit relations in text using a radically different approach, consisting of reducing the problem of identifying binary relations into a much larger set of simpler unary relations. For example, to build a Knowledge Base (KB) about presidents in the G8 countries, the presidentOf relation can be expanded to presidentOf:UNITED STATES, presidentOf:GERMANY, presidentOf:JAPAN, and so on.

1https://kbpo.stanford.edu
on. For all these unary relations, we train a multi-class (and in other cases, multi-label) classifier from all the available training data. This classifier takes textual evidence where only one entity is identified (e.g., ANGELA MERKEL) and predicts a confidence score for each unary relation. In this way, ANGELA MERKEL will be assigned to the unary relation presidentOf GERMANY, which in turn generates the triple 

\[ \text{ANGELA MERKEL} : \text{presidentOf} : \text{GERMANY} \]

To implement the idea above, we explore the use of knowledge-level supervision, sometimes called distant supervision, to train a deep learning based approach. The training data in this approach is a knowledge base and an unannotated corpus. A pre-existing Entity Detection and Linking system first identifies and links mentions of entities in the corpus. For each entity, the system gathers its context set, the contexts (e.g. sentences or token windows) where it is mentioned. The context set forms the textual evidence for a multi-class, multi-label deep network. The final layer of the network is vector of unary relation predictions and the intermediate layers are shared. This architecture allows us to efficiently train thousands of unary relations, while reusing the feature representations in the intermediate layers across relations as a form of transfer learning. The predictions of this network represent the probability for the input entity to belong to each of its relations.

To demonstrate the effectiveness of our approach we developed a new KBP benchmark, consisting of extracting unseen DBpedia triples from the text of a web crawl, using a portion of DBpedia to train the model. As part of the contributions for this paper, we release the benchmark to the research community providing the software needed to generate it from Common Crawl and DBpedia as an open source project.

As a baseline, we adapt a state of the art deep learning based approach for relation extraction (Lin et al., 2016). Our experiments clearly show that using unary relations to generate new triples greatly complements traditional binary approaches. An analysis of the data shows that our approach is able to capture implicit information from textual mentions and to highlight the reasons why the assignments have been made.

The paper is structured as follows. In section 2 we describe the state of the art in distantly super-vised KBP methodologies, with a focus on knowledge induction applications. Section 3 introduces the use of Unary Relations for KBP and section 4 outlines the process for producing and training them. Section 5 describes a deep learning architecture able to recognize unary relations from textual evidence. In section 6 we describe the benchmark for evaluation. Section 7 provides an extensive evaluation of unary relations for map exploration of what the deep learning model has learned. Section 8 concludes the paper highlighting research directions for future work.

2 Related Work

Binary relation extraction using distant supervision has a long history (Searse et al., 2012, Mintz et al., 2009). Mentions of entities from the knowledge base are located in text. When two entities are mentioned in the same sentence that sentence becomes part of the evidence for the relation (if any) between those entities. The set of sentences mentioning an entity pair is used in a machine learning model to predict how the entities are related, if at all.

Deep learning has been applied to binary relation extraction. CNN-based (Zeng et al., 2014), LSTM-based (Xu et al., 2015), attention based (Wang et al., 2016) and compositional embedding based (Gюрmez et al., 2015) models have been trained successfully using a sentence as the unit of context. Recently, cross sentence approaches have been explored by building paths connecting the two identified arguments through related entities (Peng et al., 2017; Zeng et al., 2016). These approaches are limited by requiring both entities to be mentioned in a textual context. The context aggregation approaches of state-of-the-art neural models, max-pooling (Zeng et al., 2015) and attention (Lin et al., 2016), do not consider that different contexts may contribute to the prediction in different ways. Instead, the context pooling only determines the degree of a sentence’s contribution to the relation prediction.

TAC-KBP is a long running challenge for knowledge base population. Effective systems in these competitions combine many approaches such as rule-based relation extraction, directly supervised linear and neural network extractors, distantly supervised neural network models (Zhang et al., 2016) and tensor factorization approaches to relation prediction. Compositional Universal Schema is an approach based on combining the matrix factorization approach of universal schema (Riedel et al., 2013), with representations of textual relations produced by an LSTM (Chang et al., 2016). The rows of the universal schema matrix are entity pairs, and will only be supported by a textual relation if they occur in a sentence together.

Other approaches to relational knowledge extraction have distributed representations for words or entities and used a model to predict the relation between two terms based on their semantic vectors (Droux et al., 2016). This enables the discovery of relations between terms that do not co-occur in the same sentence. However, the distributed representation of the entities is developed from the corpus without any ability to focus on the relations of interest. One example of such work is LexNET, which developed a model using the distributional word vectors of two terms to predict lexical relations between them ($D_{th}$). The terms vectors are concatenated and used as input to a single hidden layer neural network. Unlike our approach to unary relations the term vectors are produced by a standard relation-independent model of the term’s contexts such as word2vec (Mikolov et al., 2013).

Unary relations can be considered to be similar to types. Work on ontology population has considered the general distribution of a term in text to predict its type (Cimiano and Völker, 2005). Like the method of $D_{th}$, this does not customize the representation of an entity to a set of target relations.

3 Unary vs Binary Relations

The basic idea presented in this paper is that in many cases relation extraction problems can be reduced to sets of simpler and inter-related unary relation extraction problems. This is possible by providing a specific value to one of the two arguments, transforming the relations into a set of categories. For example, the infix $\text{in}$ relation between persons and countries can be decomposed into 195 relations (one relation for each country), including $\text{in} \in \text{UNITED STATES}$, $\text{in} \in \text{CANADA}$, and so on. The argument that is combined with the binary relation to produce the unary relation is called the $\text{filler argument}$. The other argument is the $\text{focal argument}$. The $\text{KB extension}$ of a unary relation is the set of all filler arguments in the KB, and the $\text{corpus extension}$ is the subset of the KB extension that occurs in the corpus.

A requisite for a unary relation is that in the training KB there should exist many triples that share a relation and a particular entity as one argument, thus providing enough training for each unary classifier. Therefore, in the example above, we will not likely be able to generate predicates for all the 195 countries, because some of them either not occur at all in the training data or they will be very infrequent. However, even in cases where arguments tend to follow a long tail distribution, it makes sense to generate unary predicates for the most frequent ones.

Figure 1: Minimum Corpus Extension to Number of Unary Relations

Figure 1 shows the relationship between the threshold for the size of the corpus extension of a unary relation and the number of different unary relations that can be found in our dataset. The relation is approximately linear on a log-log scale. There are 26 unary relations with a corpus extension of at least 10,000. These relations include:

- hasLocation:UNITED STATES
- background GROUP OR BAND
- kingdom:ANIMAL
- language:ENGLISH LANGUAGE

Lowering the threshold to 100 we have 871 unary relations and we get close to 1M unary relations with more than 10 entities.

In a traditional binary KBP task a triple has a relevant context set if the two entities occur at least once together in the corpus - where the notion of ‘together’ is typically intra-sentential (within a single sentence). In KBP based on unary relations, a triple (FILLER refers FIXED) has a relevant context

---

https://github.com/IBM/cc-dbpe
The company competes with Holcim Philippines, the local unit of Swiss company LafargeHolcim, and Eagle Cement, a company backed by diversified local conglomerate San Miguel which is aggressively expanding into infrastructure.

... said Richmond, who is vice president of Dyna Management Services, a Bermuda-based insurance management company.

On the other hand, there are many triples that have no relevant context using binary extraction, but can be supported with unary extraction. JB Hi-Fi is a company located in Australia, (unary relation hasLocation: Australia). Although “JB Hi-Fi” never occurs together with “Australia” in our corpus, we can gather implicit textual evidence for this relation from its unary relation context sets. Furthermore, even cases where there is a relevant binary context set, the contexts may not provide enough or any textual support for the relation, while the unary context sets might.

Westfarmers, Harvey Norman, The Good Guys, and Harvey Norman were also trading higher.

JB Hi-Fi is in talks to buy The Good Guys

In equities news, protective glove and condom maker Ansell and JB Hi-Fi are slated to post half year results, while Bitcoin group is expected to list on ASX.

The key indicators are: “ASX”, which is an Australian stock exchange, and the other Australian businesses mentioned, such as Woolworths, Westfarmers, Harvey Norman, The Good Guys, Ansell and Bitcoin group. There is no strict logical entailment, indicating JB Hi-Fi is located in Australia, instead there is textual evidence that makes it probable.

5 Architecture for Unary Relations

Figure 3 illustrates the overall architecture. First an Entity Detection and Linking system identifies occurrences in text of entities that are or should be in the knowledge base. Second, the contexts (here we use a sentence as the unit of context) for each entity are then gathered into an entity context set. This context set provides all the sentences that contain a mention of a particular entity and is the textual evidence for what triples are true for the entity. Third, the context set is then fed into a deep neural network, given in Figure 4. The output of the network is a set of predicted triples that can be added to the knowledge base.

Figure 4 shows the architecture of the deep learning model for unary relation based KBP. From an entity context set, each sentence is projected into a vector space using a piecewise convolutional neural network (Zeng et al., 2015). The sentence vectors are then aggregated using a Network-in-Network layer (NN) (Lin et al., 2013).

The sentence-to-vector portion of the neural architecture begins by looking up the words in a word embedding table. The word embeddings are initialized with word2vec (Mikolov et al., 2013) and updated during training. The position of each word relative to the entity is also looked up in a position embedding table. Each word vector is concatenated with its position vector to produce each word representation vector. A piecewise max-pooled convolution (PCNN) is applied over...
the resulting sentence matrix, with the pieces defined by the position of the entity argument: before the entity, the entity, and after the entity. A fully connected layer then produces the sentence vector representation. This is a refinement of the Neural Relation Extraction (NRE) (Lin et al., 2016) approach to sentence-to-vector mapping. The presence of only a single argument simply reduces from two position encoding vectors to one. The fully connected layer over the PCNN is an addition.

The sentence vector aggregation portion of the neural architecture uses a Network-in-Network over the sentence vectors. Network-in-Network (NiN) (Lin et al., 2013) is an approach of 1x1 CNNs to image processing. The width-1 CNN we use for mention aggregation is an adaptation to a set of sentence vectors. The result is max-pooled and put through a fully connected layer to produce the score for each unary relation. Unlike a maximum aggregation used in many previous works (Riedel et al., 2010; Zeng et al., 2015) for binary relation extraction the evidence from many contexts can be combined to produce a prediction. Unlike attention-based pooling also used previously for binary relation extraction (Lin et al., 2016), the different contexts can contribute to different aspects, not just different degrees. For example, a prediction that a city is in France might depend on the conjunction of several facets of textual evidence linking the city to the French language, the Euro, and Norman history.

In contrast, the common maximum aggregation approach is to move the final prediction layer to the sentence-to-vector modules and then aggregate by max-pooling the sentence level predictions. This aggregation strategy means that only the sentence most strongly indicating the relation contributes to its prediction. We measured the impact of the Network-in-Network sentence vector aggregation approach on the validation set. Relative to Network-in-Network aggregation and using the same hyperparameters, a maximum aggregation strategy gets two percent lower precision at one thousand: 66.55% compared to 68.49%.

There are 790 unary relations with at least one thousand positives in our benchmark. To speed training, we divided these into eight sets of approximately 100 relations each and trained the models for them in parallel. Unary relations based on the same binary relation were grouped together to share useful learned representations. The resulting split also put similar numbers of positive examples in the training set for each model.

Training continued until no improvement was found on the validation set. This occurred at between five and nine epochs. All eight models were trained with the hyperparameters in Table 1. Dropout was applied on the penultimate layer, the max-pooled NiN. Based on validation set performance, we found that when larger numbers of relations are trained together the NiN filters and sentence vector dimension must be increased. Of all the hyperparameters, the training time is most sensitive to the

Table 1: Hyperparameters used

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>word embedding</td>
<td>50</td>
</tr>
<tr>
<td>position embedding</td>
<td>5</td>
</tr>
<tr>
<td>PCNN filters</td>
<td>1000</td>
</tr>
<tr>
<td>PCNN filter width</td>
<td>3</td>
</tr>
<tr>
<td>sentence vector</td>
<td>400</td>
</tr>
<tr>
<td>NiN filters</td>
<td>400</td>
</tr>
<tr>
<td>dropout rate</td>
<td>0.5</td>
</tr>
<tr>
<td>decay multiplier</td>
<td>0.003</td>
</tr>
<tr>
<td>batch size</td>
<td>16</td>
</tr>
<tr>
<td>optimizer</td>
<td>SGD</td>
</tr>
</tbody>
</table>

The precision / recall curves and focus on the relative area under the curves to evaluate the quality of different systems.

Figure 5 shows the distribution of triples with relevant unary context sets per relation type. The relations giving rise to the most triples are high level relations such as hasLocation, a super-relation comprised of the sub-relations: country, state, city, headquarter, hometown, birthPlace, deathPlace, and others. Interestingly there are 165 years with enough people born in them to produce unary relations. While these all will have at least 100 relevant context sets, typically the context sets do not have textual evidence for any birth year. Perhaps most importantly, there are a large number of diverse relations that are suitable for a unary KBP approach. This indicates the broad applicability of our method.

To test what improvement can be found by incorporating unary relations into KBP, we combine the output of a state-of-the-art binary relation extraction system with our unary relation extraction system. For binary relation extraction, we use a slightly altered version of the PCNN model from NRE (Lin et al., 2016), with the addition of a fully connected layer for each sentence representation before the max-pooled aggregation over relation predictions. We found this refinement to perform slightly better in NYT-FB (Riedel et al., 2010), a standard dataset for distantly supervised relation extraction. The binary and unary systems are trained from their relevant context sets to predict the triples in train. The validation set is used to tune hyperparameters and choose a stopping point for training. We combine the output of the two systems by, for each triple, taking the highest confidence from each system.

7 Evaluation

Figure 6 shows the precision-recall curves for unary only, binary only and the combined system. The unary and binary systems alone achieve similar performance. But they are effective at very different triples. This is shown in the large gains from combining these complementary approaches. For example, at 0.5 precision, the combined approach has a recall of more than double (15,750 vs 7,400) compared to binary alone, which represents over 100% relative improvement.

The recall is given as a triple count rather than...
a percentage. Traditional attempts to measure the recall of KBP systems use the set of all triples explicitly stated in text for the denominator of recall. This is unsuitable for evaluating our approach because the system is able to make probabilistic predictions based on implicit and partial textual evidence, thus producing correct triples outside the classic recall basis.

Cold Lake Provincial Park (Alberta, Canada) is mentioned in two sentences in the Common Crawl English text. The unary relational knowledge induction system predicts hasLocation.CANADA with the highest confidence (over 90%). Both sentences contribute to the decision. We see high weight from words including "cold", "provincial" and "French". A handful of countries have "provincial parks" including Argentina, Belgium, South Africa and Canada. Belgium and Canada have substantial French speaking populations and Canada has by far the coldest climate.

- located within 10 minutes of cold lake with quick access to OOV ridge ski hill , cold lake provincial park and french bay .
- welcome to cold lake provincial park on average 4.00 pages are viewed each , by the estimated 959 daily visitors .

Rock Kills Kid is a band mentioned twice in the corpus. From this context set, the relation background.GROUP.OR.BAND is predicted with high confidence. The fact that "Kid" occurs in the name of the entity seems to be important in identifying it as a musical group. The first sentence also draws focus to the band’s connection to rock and pop.

While the second sentence seems to recognize the band - song (year) pattern as well as the comparison to Duran Duran.

- the latest stylish pop synth band is rock kills kid
- rock kills kid - are you famous ? ( 2006 ) who ever thought duran duran would become so influential ?

The Japanese singer-songwriter Masaki Haruna, aka Klaha is mentioned twice in the corpus. From this context set, the relation background.SOLO_SINGER is predicted with high confidence. The first sentence clearly establishes the connection to music while the second indicates that Klaha is a solo artist. The conjunction of these two facets, accomplished through the context vector aggregation using NIN permits the conclusion of SOLO_SINGER.

- tvk music chat interview klaaha
- klaaha tvk music chat OOV red scarf interview the tv - k folks did after klaaha went solo :

8 Conclusions

In this paper we presented a new methodology to identify relations between entities in text. Our approach, focusing on unary relations, can greatly improve the recall in automatic construction and updating of knowledge bases by making use of implicit and partial textual markers. Our method is extremely effective and complement very nicely existing binary relation extraction methods for KBP.

This is just the first step in our wider research program on KBP, whose goal is to improve recall by identifying implicit information from texts. First of all, we plan to explore the use of more advanced forms of entity detection and linking, including propagating features from the EDL system forward for both unary and binary deep models. In addition we plan to exploit unary and binary relations as source of evidence to bootstrap a probabilistic reasoning approach, with the goal of leveraging constraints from the KB schema such as domain, range and taxonomies. We will also integrate the new triples gathered from textual evidence with new triples predicted from existing KB relationships by knowledge base completion.

References


Semantic Concept Discovery over Event Databases

Oktie Hassanzadeh, Shari Trewin, and Alfonso Giorno

IBM Research, Yorktown Heights, NY, USA
hassanzadeh@us.ibm.com

Abstract. In this paper, we study the problem of identifying certain types of entities (e.g., persons, organisations, topics) for a given analysis question with the goal of assisting a human analyst in writing a deep analysis report. We consider a case where we have a large event database describing events and their associated news articles along with meta-data describing various event attributes such as people and organisations involved and the topic of the event. We describe the use of semantic technologies in question understanding and deep analysis of the event database, and show a detailed evaluation of our proposed concept discovery techniques using reports from Human Rights Watch organisation and other sources. Our study finds that combining our neural network based semantic term embeddings over structured data with an index-based method can significantly outperform either method alone.

1 Introduction

Analysts are often tasked with preparing a comprehensive, accurate, and unbiased report on a given topic. The first step in preparing such a report is a daunting discovery task that requires researching through a massive amount of information. Information sources can have large volume, variety, varying velocity, and accuracy - the common characteristics of the so-called Big Data sources. Many times the analysis requires a deep understanding of various kinds of historical and ongoing events that are reported in the media. To enable better analysis of events, there exist several event databases containing structured representations of events extracted from news articles. Examples include GIBELT [17], ICWSM [4], and EventRegistry [5]. These event databases have been successfully used to perform various kinds of analysis tasks, e.g., forecasting societal events [22]. However, there has been little work on the discovery aspect of the analysis, that is, in a gap between the information requirements and the available data, and potentially a biased view of the available information.

In this paper, we present a framework for concept discovery over event databases using semantic technologies. Unlike existing concept discovery solutions that perform discovery over text documents and in isolation from the remaining data analysis tasks [18, 28], our goal is providing a unified solution that allows deep understanding of the same data that will be used to perform
other analysis tasks (e.g., hypothesis generation [27], scenario planning [29], or building models for forecasting [15, 22]). Figures 1 and 2 show different views of our system’s UI that is built using our concept discovery framework APIs. The analyst can enter a natural language question or a set of concepts, and retrieve collections of relevant concepts identified and ranked using different discovery algorithms described in Sect. 3. Using this system provides a new starting point for an analyst’s work. Instead of performing complex search queries and examining pages of results, the analyst reviews the related concepts, exploring what connects them to the analytical question. Unexpected concepts broaden the scope of their thinking, helping to overcome confirmation bias. A key aspect of our framework is the use of semantic technologies. In particular:

- A unified view over multiple event databases and a background RDF knowledge base is achieved through semantic link discovery and annotation.
- Natural language or keyword query understanding is performed through mapping of input terms to the concepts in the background knowledge base.
- Concept discovery and ranking is performed through neural network based semantic term embeddings.

In what follows, we first describe the overall framework and its various components. We then describe the algorithms used for concept discovery and ranking. In Sect. 4, we present the methodology and results of our evaluation using a ground truth built from a large corpus of reports written by human experts.

2 Concept Discovery Framework

Figure 3 shows the architecture of our system. The system takes in as input a set of event databases and RDF knowledge bases and provides as output a set of APIs that provide a unified retrieval mechanism over input data and knowledge bases, and an interface to a number of concept discovery algorithms. In what follows, we describe the input sources and each of the components in detail.

2.1 Event Data and Knowledge Sources

Event databases are structured records describing various kinds of societal, political, or economic events. While event extraction from text is a well-studied topic in the NLP literature [12,14] with a dedicated track at the annual Text Analysis Conference (TAC) [6], there are only a few publicly available large-scale event databases. The input of these event databases is a large corpus of news articles that are either gathered from various news sources (e.g., news agencies and other proprietary sources) or crawled from the Web. The output is structured records (i.e., relational data tables) describing various features of the identified events.
GDELT. The Global Data on Events, Location, and Tone (GDELT) project [17] claims to be "the largest, most comprehensive, and highest resolution open database of human society ever created". GDELT data contains three databases. GDELT Event database provides coded event data based on a popular scheme using the CAMEO (Conflict And Mediation Event Observations) coding framework [21] to code actors and actions. GDELT includes other features of the events such as the date of the event, information about the source articles, numerical scores reflecting the "tone" of the source articles and other similar features, and geographical coordinates. The second database provided by GDELT is the Global Knowledge Graph (GKG) and contains records describing the source articles of the events. Each record provides a comprehensive set of numerical features for the article, in addition to annotations with several dictionaries of persons, organizations, and "themes". The third database GDELT provides is the Mentions database which connects event records with GKG article records.

The most recent version of GDELT data is updated daily and includes historical data since February 2015. At the time of this writing, we have ingested 128,946,146 Event records, 157,082,264 GKG records, and 436,961,965 Mention records.

ICEWS. Integrated Conflict Early Warning System [11, 38] provides a coded event database similar to the GDELT Event database. ICEWS event records describe features of source and target actors including their name, "sector", and country, features of the action including date, source, a short text description, and geographical descriptions. A recent version of the data also includes CAMEO codes for actions. We have ingested the most recently available public data that has a coverage of historical events from 1995 to 2015, with 14,757,915 records.

EventRegistry. The EventRegistry [16] project takes a completely different approach than the coded event databases, and performs event extraction based on a clustering of news articles and event mentions. EventRegistry records contain a multilingual title and summary text, the number of articles reporting the event, the event date (when the event has happened or will happen and not the report date as in coded event databases), and a set of concepts along with the concept type (e.g., location, person, or "topic") and its Wikipedia URL. At the time of this writing, we have ingested 2,889,497 event records extracted from 33,540,678 news articles from the past two years, with 98,435,900 concept annotations, 42,006,679 similarity links, and 772,553 location annotations.

Knowledge Sources. In addition to event data, our system also ingests publicly available RDF knowledge bases to use as a source of reference knowledge. Our current knowledge sources include Wikidata [29], DBpedia [8], YAGO2 [23], and Freebase [6]. At the time of this writing, we have ingested over 6.3 billion RDF triples, containing over 888 million entities (unique URIs) and over 83 million English label statements.

2.2 Ingestion
As shown in Fig. 3, we have a common ingestion pipeline for both the event databases and knowledge sources that is capable of crawling remote sources, parsing structured relational, semi-structured (JSON), and RDF (NTriples) data, cleaning invalid records or statements and applying basic filters (e.g., removing non-English labels), and finally storing the data. Our platform is implemented on top of Apache Hadoop and Spark, enabling efficient data processing on a cluster on public or private cloud.

2.3 Curation
We adopt a pac-as-you-go integration approach [13, 19] and perform only a minimal curation by a lightweight mapping of known entities, linking them using a common URI when possible. To integrate the knowledge sources we use the existing Wikipedia URIs. We then index all the facts (RDF triples) in a key-value store (powered by Rink [4]) in addition to a document store (powered by SolrCloud [5]) that makes it possible to perform highly efficient fact-based or label-based knowledge search. We also create an auxiliary unified index of common entities using our mapping strategy that results in a collection of 16,109,676 entities with Wikipedia URIs, each linked with one or more of their Wikidata, DBpedia, YAGO, and Freebase URIs. All the event databases are indexed in a similar way in our key-value and document stores, with labels matched and linked with a Wikipedia URI when possible.

2.4 Semantic Embeddings Engine
Inspired by the idea of word embeddings in NLP [21], recent work has proposed the use of shallow neural networks to map values in structured tables to vectors (referred to as embeddings) [19]. This enables powerful semantic similarity queries even without a prior knowledge of the database schema and contents. We adopt a similar strategy and transform every value in the input event database into an embeddings vector using a variation of the continuous skip-gram model.
of the original wordtree [7, 30, 21]. The first step in this process is a virtual doc-
ment creation process in MapReduce, turning each row in the input database
into a context in a corpus of text. We then feed the text corpus into a wordtree
model construction modified to take into account the different characteristics of
structured data:

- The order of columns in structured databases is of little importance. While
distance between two words in a text document ranks them further in terms of
context, the first column in a database table is as relevant to the second column
as to the last column.

In text documents, typically a random-sized window of words is selected. The
length of each database record is fixed and so there is no need for a random
window size.

- Most importantly, while all words in a text corpus are treated in the same
way and do not have specific roles, values in different columns in structured
sources describe different (event) features and may need to be grouped and
queried differently. There is often a need to search over (or query using) the
terms from specific attributes (columns).

Once attribute values are mapped into low-dimensional vectors, aggregate vec-
tors can represent individual records (articles or events), and similarity queries
over the vectors can be used for concept discovery and analysis as described in
Sect. 3. These vectors represent the semantic context of every single value seen in
the input database, enabling a powerful and extremely efficient method of performing
similarity analysis over large amounts of data. As an example, the corpus size
(number of words in the "virtual documents") for GDELT KG is \(23,901,358,498\)
while the size of the vocabulary (number of unique words) in our embeddings is
3,828,213. Still, a key requirement is efficient similarity queries over the vectors
with milliseconds running time to enable real-time analysis queries through our
UI (Figs. 1 and 2) as some analysis queries require several similarity queries each
over millions of vectors. We achieve this using the efficient Annoy library [1] as
the core of our embeddings management system.

2.5 Event Knowledge Graph and Concept Discovery APIs

The final outcome of all the components is a set of APIs to perform knowledge
graph and concept discovery queries. In particular:

- **Lookup APIs.** These APIs provide access to the ingested and curated
event data and knowledge. For example, one can perform search over knowl-
dge base entity labels and subsequently retrieve human-readable facts as
JSON objects. Using this API the user can retrieve mention information
about each of the concepts shown under the “Global Context” box in Fig. 2a.
These APIs also enable queries across event databases, e.g., retrieves ICEWS,
GDELT, and EventRegistry events in a given time range that is annotated
with a particular context.

- **Natural Language and Keyword Query Understanding APIs.** These
APIs turn the user query into a set of knowledge base concepts and event
database terms. In Fig. 2a, the concepts shown under the “Global Context”
are extracted using the API that outputs knowledge base concepts, whereas
the terms shown under “DeepBlur Context” are terms found in GDELT KG data
used for the shown concept discovery results.

- **Concept Discovery and Ranking APIs.** These APIs take a set of concepts
or terms and return as output a ranked list of concepts of different types (e.g.,
Persons, Organizations, Themes). Details of the concept ranking algorithms
are described in the following section.

3 Concept Ranking Algorithms

In this section, we describe three classes of algorithms for concept discovery and
ranking. These algorithms identify and rank a set of most relevant concepts of
various types (e.g., persons or organizations) for a given set of concepts. An
example use of these rankings is shown in Fig. 2 where sorted lists of ranked
concepts relevant to the user’s analysis question are shown. The end goal is
providing the output either directly to an analyst or to other components of an
analysis system. We first describe the algorithms, followed by an evaluation of
their effectiveness in identifying relevant concepts.

3.1 Index-Based Method (co-occur)

The co-occur method relies on an efficient index to measure the level of co-
occurrence of concepts in a collection of events and uses this as a measure of
relevance. Using the index described in Sect. 2.3, we can search for (all or recent)
event records annotated with a given set of concepts. By counting the concept
annotations for every record in the output, a list of most frequently co-occurring
concepts of various types is returned along with the percentage of co-occurrence
of the annotations among all the retrieved event records. Figure 2b shows an
example of ranked “Topic”, “Key Player” (Person), and “Location” concepts
over EventRegistry event records. The concepts extracted from the input ques-
tion are “Caracas”, “Protest”, and “Venezuela”. Obviously, these concepts them-
selves are on top of the lists as they appear in 100% of the event records contain-
ing them. The topic concept “Government” appears in 87% of the events and
“Nicole Maduro” appears in 69% of the events, indicating that these concepts
are highly relevant to the input concepts in recent events. In

3.2 Deep Similarity Method Using Semantic Embeddings (context)

The context method relies on the term embeddings built over an event database
as described in Sect. 2.4. First, a vector is retrieved for each of the terms extracted
from the input question (where there exists a vector representation in the embed-
dings space), and an average vector is constructed by summing the values in each
dimension and normalizing the resulting vector. Using the embeddings management system, the most similar vectors of various kinds of terms are retrieved, ranked by their similarity to the average vector. Figure 3a shows an example of concept rankings with the same question. The API used in the following evaluation queries embeddings built over 157 million GDELT KGK records, with vectors of size 200 and cosine similarity as our choice of vector similarity measure. These rankings result in less-obvious and harder-to-explain but deeply relevant sets of concepts in the output.

3.3 Combination Methods

We implement two combination methods. In the co-occur.context method, we retrieve a set of 3k results of an index-based method, re-rank the output using the embeddings-based similarity of the terms in the output, then select the top k terms. In the context.co-occur method, we retrieve 3k results of the context retrieval and sort the output based on their position in the co-occur results before selecting the top k terms. In the following section in Table 1, we show an example of how these re-rankings improve the results.

4 Experiments

To evaluate the performance of the concept ranking algorithms, we use queries represented as sets of query terms that would be extracted from an analyst’s question. We sought to identify a “ground truth” of concepts related to each query, where a concept is a person or organization directly involved with the topic of the query.

As mentioned in Sect. 2, our framework enables real-time or near real-time response time for each of the algorithms and so we do not compare running times.

4.1 Evaluation Data

To our knowledge, there is no existing public data set that identifies key people and organizations for a set of analytical questions. To provide an objective basis for our evaluation, we used reports that summarize a political or social event or situation, making the assumption that these reports are a response to such a question, and will mention the most important key players. We did not use news articles because these are the source of GDELT events, so as not to bias the results. We identified three potential sources of reports:

- Declassified US Government intelligence reports. We were only able to identify one such report that relates to the time period covered by the GDELT KGK event database: “Assessing Russian Activities and Intentions in Recent US Elections”, released in January 2017.

- Wikipedia pages describing a newsworthy event or topic with relevance to social unrest. For example “Impeachment of Dilma Rousseff” or “Shortages in Venezuela”.

- Human Rights Watch reports. These are detailed descriptions of specific human rights situations around the world, for example “Philippine Police Killings in Duterte’s “War on Drugs””. 1,001 such reports are available in HTML.

Using these sources we developed test queries consisting of a small set of query terms and “ground truth” sets of people and organizations. The query terms can include a country, people, organizations, and themes (drawn from the GDELT KGK themes described earlier). The “ground truth” items are selected from the people and organizations mentioned in the report, to represent the ideal response of the system to the query. Three query sets were developed: Manual, Curated, and Auto.

Manual: A set of 12 hand-built queries derived from 6 Wikipedia pages. 5 Human Rights Watch Reports from 2014 or later; and 1 declassified intelligence report. All queries specified the country most strongly associated with the report, and 1-5 manually selected themes from GDELT KGK, in addition to any people or organizations mentioned in the report title. For example, the query terms for the Wikipedia page “Impeachment of Dilma Rousseff” consisted of the country “Brazil”, the person “Dilma Rousseff”, and the theme “IMPEACHMENT”.

The ground truth concepts were the people and organizations mentioned in each report as key players for the topic. We removed concepts that were found in the report but not in our embedding. Subsequently, each query had an average of 10 ground truth people and 7 ground truth organizations.

Curated: A set of 25 queries based on Wikipedia pages describing events from 2014-2018, where the query terms (country, people, organizations and themes) were selected manually, but the ground truth terms were automatically generated from the Wikipedia links within the page, and then curated to remove non-person and non-organization terms. Some people and organizations mentioned in the original report may be missed in this process.

Auto: A larger set of 179 queries derived from the Human Rights Watch reports, with fully automatic generation of both query terms and ground truth. To generate the query terms, the query builder used the document title, subtitle and text - a short paragraph of a few sentences describing the report. It used concept extraction software that combines output from OpenNLP [2] and OpenNLP [2] to identify noun phrases referring to named entities, and assigns types to them according to their linguistic context. We relied on these types to identify countries, people and organizations in the text. The ground truth people and organizations were generated by using the same concept extraction software, applied to the full text of the report. We removed people and organizations not
found in our embedding. Finally, we selected the 179 queries that had a country, at least one other query term (person, organization or theme), and had ground truth terms that were not already present in the query. Of these, the majority (162) were queries consisting of a country and the single organization ‘Human Rights Watch’. 26 contained a person, and 51 contained an organization other than Human Rights Watch. From these, we further selected only the usable queries with ground truth items that were not in the query (143 for people and 155 for organizations). These queries had, on average, 21 ground truth people and 32 ground truth organizations.

4.2 Example Results

Table 1 shows the set of ground truth people and the output of the algorithms for a query from the manual test set, based on the 2016 Human Rights Watch report “Venezuela’s Humanitarian Crisis. Severe Medical and Food Shortages, Inadequate and Repressive Government Response”.

Of the 14 most relevant people identified in the report, only 7 were present in the embedding (indicated in column 1 with (*)). Two of the others were not found in the KG data at all, and the remaining five were mentioned only 1–59 times - not enough to be included in the embedding. For organizations, 17 were mentioned in the report, and 9 of these were found in the embedding. The KG data does not often include common acronyms like BBC or FBI, although there are some exceptions. This creates challenges for automated testing since the reports often use an acronym to refer to an organization.

The most relevant people mentioned in the report, 14 in all, are listed in the first column of Table 1, while the remaining columns show the top 10 results for each algorithm, given the query for the country “Venezuela” and the KG theme “SELF-IDENTIFIED HUMANITARIAN CRISIS”. The seven items from the ground truth that are potentially findable in the index and in the embedding are indicated with (*) in column one, while the found items are highlighted in bold in the subsequent columns, including alternate spellings of the same person’s name.

The co-occur method finds only the Venezuelan leaders in the top 14. It also returns ten other world leaders, politicians and spokespersons. These people have either made statements about Venezuela’s humanitarian crisis, or Venezuela has made comments about their own country’s crisis (e.g. Bashar Assad). Although Donald Trump was not yet president of the United States during the period covered by the data, his opinions on foreign policy in Latin America were discussed in the news, and he made statements about the situation in Venezuela. Two journalists who write frequently about Venezuela are also suggested (Joshua Goodman, Gonzalo Solano).

In marked contrast, the context method’s results do not include any foreign leaders and politicians. Instead, there are seven Venezuelan politicians, along

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>co-occur</th>
<th>context</th>
<th>co-occur.context</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nicolas Maduro (*)</td>
<td>Nicolas Maduro</td>
<td>Deley Rodríguez</td>
<td>Deley Rodríguez</td>
</tr>
<tr>
<td>Hugo Chávez (*)</td>
<td>Béarç Obàma</td>
<td>Nicolás Maduro</td>
<td>Nicolás Maduro</td>
</tr>
<tr>
<td>Luis Almagro (*)</td>
<td>John Kerry</td>
<td>Nicolás Maduro</td>
<td>Béarç Obàma</td>
</tr>
<tr>
<td>Donald Trump</td>
<td>Joshua Goodman</td>
<td>Henry Capriles</td>
<td>Juan Manuel Santos</td>
</tr>
<tr>
<td>Carlos Córdoba</td>
<td>Guzmán Solano</td>
<td>Ignacio Chima</td>
<td>John Kerry</td>
</tr>
<tr>
<td>Flavio Funes</td>
<td>Vladimir Putin</td>
<td>Andrew Cardillo</td>
<td>Bashar Assad</td>
</tr>
<tr>
<td>Rafael Ureña</td>
<td>David Grainger</td>
<td>David Smilde</td>
<td>Donald Trump</td>
</tr>
<tr>
<td>Félix Cardona</td>
<td>John Kirby</td>
<td>Ernesto Villegas</td>
<td>Gonzalo Solano</td>
</tr>
<tr>
<td>Salva Kiir</td>
<td>Salva Kiir</td>
<td>Luis Almagro</td>
<td>Vladimir Putin</td>
</tr>
</tbody>
</table>

with four journalists and a human rights advocate and academic (David Smilde), and the secretary-general of the Organization of Latin American States (Luis Almagro). Some of these politicians are very closely associated with the humanitarian crisis in Venezuela, notably Vladimir Padrino, the Venezuelan Minister of Defense, who is responsible for food distribution, even though they were not mentioned by name in the report.

Combining these methods by ranking the first 90 co-occur results according to their context ranking moved five highly related candidates to the top of the list, including the new ground truth person: Luis Almagro. Similarly, the context.co-occur method (omitted from Table 1 for space constraints) moved four items to the top of the ranking, including the mispelling of Nicolás Maduro as Nicolás Madero. Both combination methods slightly increased the number of ground truth items found in the top 10 ranked results from 2 to 3 to 4 out of a possible maximum of 7.

4.3 Evaluation Method

To evaluate and compare the methods of identifying key players, we applied each of the four methods (co-occur, context, co-occur.context and context.co-occur) to the test query data sets (manual, curated and auto), for both people and organizations. All methods were limited to 30 returned candidates. For each query, we calculated four classic information retrieval evaluation measures: precision (ratio of correct concepts in the output), recall (ratio of
ground truth concepts returned in the output), F1 (harmonic mean of precision and recall), and average precision (average precision value at all the ranks where a correct concept is returned). Overall values for each test set are reported as the mean of the values for the individual queries in the set. Following the recommendation by Snchez et al. [5], we performed randomization test and two-tailed paired samples t-tests to test for statistical significance.

4.4 Evaluation Results

Manual. Table 2 shows the results for the manual data set. For person experiments, all measures showed better performance from the combination methods, with the context method performing the lowest. The co-occur, context method outperformed the co-occur method by 15%. However, pairwise comparisons of F1 scores between methods showed only the [context, context, co-occur] and (co-occur, co-occur, context) pairs to be statistically significantly different (p < 0.05). For the organization experiments, again the co-occur, context combination method performed best overall on all four measures, but only the [context, context, co-occur] pair was found to be statistically significant in terms of comparison by MAP or F1 scores. The lack of statistical significance is due to the high variance of the results for each query, and show in part the need for a larger data set for a proper comparison as our overall results described in Sect. 4.4 also confirm.

Table 2. Accuracy results over the manual data set.

<table>
<thead>
<tr>
<th>Person</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>MAP</td>
<td>0.333</td>
</tr>
<tr>
<td></td>
<td>0.251</td>
</tr>
<tr>
<td></td>
<td>0.179</td>
</tr>
<tr>
<td></td>
<td>0.184</td>
</tr>
<tr>
<td>F1</td>
<td>0.192</td>
</tr>
<tr>
<td></td>
<td>0.228</td>
</tr>
<tr>
<td></td>
<td>0.178</td>
</tr>
<tr>
<td></td>
<td>0.183</td>
</tr>
<tr>
<td>Pr</td>
<td>0.133</td>
</tr>
<tr>
<td></td>
<td>0.158</td>
</tr>
<tr>
<td></td>
<td>0.117</td>
</tr>
<tr>
<td></td>
<td>0.119</td>
</tr>
<tr>
<td>Ro</td>
<td>0.372</td>
</tr>
<tr>
<td></td>
<td>0.417</td>
</tr>
<tr>
<td></td>
<td>0.436</td>
</tr>
<tr>
<td></td>
<td>0.459</td>
</tr>
</tbody>
</table>

Curated. Table 3 shows the results for the curated data set. For the person experiments, the overall pattern was very similar to the manual data set, with the co-occur, context method showing the best performance across all measures, including an 18% improvement for F1 over the co-occur method. For F1, the differences between the [context, co-occur], (context, context, co-occur) and (co-occur, co-occur, context) pairs were statistically significant. For the organization experiments the co-occur and co-occur, context methods performed the best, and their F1 scores were not significantly different, while all other pairwise comparisons were, except for the two lowest performing methods: context and context, co-occur.

Table 3. Accuracy results over the curated data set.

<table>
<thead>
<tr>
<th>Person</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>MAP</td>
<td>0.323</td>
</tr>
<tr>
<td></td>
<td>0.251</td>
</tr>
<tr>
<td></td>
<td>0.179</td>
</tr>
<tr>
<td></td>
<td>0.184</td>
</tr>
<tr>
<td>F1</td>
<td>0.192</td>
</tr>
<tr>
<td></td>
<td>0.228</td>
</tr>
<tr>
<td></td>
<td>0.178</td>
</tr>
<tr>
<td></td>
<td>0.183</td>
</tr>
<tr>
<td>Pr</td>
<td>0.133</td>
</tr>
<tr>
<td></td>
<td>0.158</td>
</tr>
<tr>
<td></td>
<td>0.117</td>
</tr>
<tr>
<td></td>
<td>0.119</td>
</tr>
<tr>
<td>Ro</td>
<td>0.372</td>
</tr>
<tr>
<td></td>
<td>0.417</td>
</tr>
<tr>
<td></td>
<td>0.436</td>
</tr>
<tr>
<td></td>
<td>0.459</td>
</tr>
</tbody>
</table>

Anto. Table 1 shows the results for the auto data set. For these results, organizations followed a similar pattern to the two other datasets, and all pairwise comparisons were statistically significant for all metrics, with the only exception for MAP, where the two combination methods were not distinguishable. For person experiments, the results were lower, less than 0.1 for all metrics and methods, so that while the combination methods produced around 16% higher average scores, the differences were not statistically significant, with the exception of the (context, context, co-occur) and (co-occur, co-occur, context) pairs for F1 or MAP.

Table 4. Accuracy results over the auto data set.

<table>
<thead>
<tr>
<th>Person</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>MAP</td>
<td>0.441</td>
</tr>
<tr>
<td></td>
<td>0.050</td>
</tr>
<tr>
<td></td>
<td>0.132</td>
</tr>
<tr>
<td></td>
<td>0.117</td>
</tr>
<tr>
<td>F1</td>
<td>0.458</td>
</tr>
<tr>
<td></td>
<td>0.066</td>
</tr>
<tr>
<td></td>
<td>0.084</td>
</tr>
<tr>
<td></td>
<td>0.308</td>
</tr>
<tr>
<td>Pr</td>
<td>0.051</td>
</tr>
<tr>
<td></td>
<td>0.059</td>
</tr>
<tr>
<td></td>
<td>0.173</td>
</tr>
<tr>
<td></td>
<td>0.163</td>
</tr>
<tr>
<td>Ro</td>
<td>0.099</td>
</tr>
<tr>
<td></td>
<td>0.099</td>
</tr>
<tr>
<td></td>
<td>0.224</td>
</tr>
<tr>
<td></td>
<td>0.217</td>
</tr>
</tbody>
</table>

Comparing Results Across the Data Sets. We also explored whether the different data sets provided similar results. Figure 1 shows F1 values for people (left) and organizations (right) as boxplots. Each box indicates the interquartile range of the data, the center line indicates the median value, the whiskers above and below give the 95% confidence intervals, and circles indicate outliers. Significant differences are indicated above with red brackets. For people, the less curated sets of queries produced lower results, but the pattern of results is very similar across all three datasets. Recall that the auto queries did not contain any themes, and so they often did not capture the topic of a report well, giving the system a low chance of success. Results were twice as good for organizations as for people in the auto data set, probably reflecting the large number of queries that included an organization. Again, the pattern of results remained similar across the data sets. We observed similar trends for other accuracy measures.
4.5 Discussion

Overall, both the combination algorithms performed better than the individual co-occur and context algorithms. This suggests that combining methods did not necessarily improve the overall performance compared to using each method individually.

Not surprisingly, the algorithms produced the best results on the manual test set, followed by the curated set, and the lowest values for the automatically generated set, which has less well constructed queries that do not capture the topic of the report as well. Importantly, the similarities between data set results when comparing the concept discovery algorithms increases confidence in the evaluation, and more generally in the use of automated methods as a valid and scalable way to approach the evaluation of concept discovery algorithms, despite the noise and loss of accuracy compared to hand-curated data.

Our approach to evaluation has some limitations. Our source reports do not mention all of the people and organizations relevant to the topic by name. We do not translate mentions like “The Minister for the Interior” into a named person, and nor do we attempt to resolve references to groups like “Brazilian steel companies.” Our methods also draw from articles published after the publication of the report, when new concepts may be introduced. All people or organizations found by our methods but not named in the ground truth are treated as wrong answers, but some of these may be highly relevant to the topic. In the example shown in Table 1, the majority of the persons returned by the context method are in fact highly relevant despite the fact that the input report did not contain their names. This shows that potential use case for our system is complementing analysts in finding concepts that are not already covered in their report.

Also note that there is often a major difference between the number of candidates proposed (30) and the number of ground truth items provided (generally less than 30). Thus, our reported accuracy scores are very low and underestimate the overall quality of the responses.
A Systematic Classification of Knowledge, Reasoning, and Context within the ARC Dataset

Michael Boratko, Harshit Padigela, Divyendra Miklilineni, Pritish Yuvaraj, Rajarshi Das, Andrew McCallum
College of Information and Computer Sciences
University of Massachusetts, Amherst MA

Maria Chang, Achille Fokoue-Nkoutche, Pavan Kapanipathi, Nicholas Mattei, Ryan Musa, Kartik Talamadupula, Michael Witbrock
IBM Research, Yorktown Heights NY

Abstract

The recent work of Clark et al. (2018) introduces the AI2 Reasoning Challenge (ARC) and the associated ARC dataset that partitions open domain, complex science questions into an Easy Set and a Challenge Set. That paper includes an analysis of 100 questions with respect to the types of knowledge and reasoning required to answer them; however, it does not include clear definitions of these types, nor does it offer information about the quality of the labels. We propose a comprehensive set of definitions of knowledge and reasoning types necessary for answering the questions in the ARC dataset. Using ten annotators and a sophisticated annotation interface, we analyze the distribution of labels across the Challenge Set and statistics related to them. Additionally, we demonstrate that although naive information retrieval methods return sentences that are irrelevant to answering the query, sufficient supporting text is often present in the (ARC) corpus. Evaluating with human-selected relevant sentences improves the performance of a neural machine comprehension model by 42 points.

1 Introduction

The recent work of Clark et al. (2018) introduces the AI2 Reasoning Challenge (ARC)1 and the associated ARC dataset. This dataset contains science questions from standardized tests that are separated into an Easy Set and a Challenge Set. The Challenge Set is comprised of questions that are answered incorrectly by two solvers based on Pointwise Mutual Information (PMI) Information Retrieval (IR). In addition to this division, a survey of the various types of knowledge as well as the types of reasoning that are required to answer various questions in the ARC dataset was presented. This survey was based on an analysis of 100 questions chosen at random from the Challenge Set. However, very little detail is provided about the questions chosen, the annotations provided, or the methodology used. These questions surround the very core of the paper, since the main contribution is a dataset that contains complex questions. Additionally, while their manual analysis suggests that 95% of the questions can be answered using the ARC corpus, Clark et al. (2018) note that the IR system (Elasticsearch 2) serves as a severe bottleneck. Our annotation process supports this observation, but we also find that simple reformulations to the query can greatly increase the quality of the retrieved sentences.

Contributions: In this work, in order to overcome some of the limitations of Clark et al. (2018) described above, we present a detailed annotation process for the ARC dataset. Specifically, we (a) introduce a novel labeling interface that allows a distributed set of annotators to label the knowledge and reasoning types necessary for answering the questions in the ARC dataset. Using ten annotators and a sophisticated annotation interface, we analyze the distribution of labels across the Challenge Set and statistics related to them. Additionally, we demonstrate that although naive information retrieval methods return sentences that are irrelevant to answering the query, sufficient supporting text is often present in the (ARC) corpus. Evaluating with human-selected relevant sentences improves the performance of a neural machine comprehension model by 42 points.

1https://www.elastic.co/products/elasticsearch
ment to improve existing QA systems. Our annotators were also asked to mark whether individual retrieved sentences were relevant to answering a given question. Our labeling interface logs the retrieved passages quoted by each annotator, as well as their relevance annotations. To quantitatively demonstrate the effectiveness of the relevant sentences, we (d) evaluate a subset of questions and the relevant retrieval results with a pre-trained DgQA model (Chen et al., 2017), and find that the performance of the system increases by 42 points.

2 Related Work

To explore reading comprehension as a research problem, Hirschman et al. (1999) manually created a dataset of 3rd and 6th grade reading comprehension questions with short answers. The techniques that were explored for this dataset included pattern matching, rules, and logistic regression. More such datasets have been created that include natural language questions: for instance, MCTest (Richardson et al., 2013). MCTest is crowdsourced and comprises of 660 elementary level children’s fictional stories, which are the source of questions and multiple choice answers. Questions and answers were constructed with a restrictive vocabulary that a 7-year-old could understand. Half of the questions constructed necessitated the answer to be derived from two sentences, with the motivation being to encourage research in multi-hop (one-hop) reasoning. Recent techniques such as Wang et al. (2015) and Yin et al. (2016) have performed well on this dataset. Currently, SQuAD (Rajpurkar et al., 2016) is one of the most popular datasets for reading comprehension: it uses Wikipedia passages as its source, and question-answer pairs are created using crowd-sourcing. While it is stated that SQuAD requires logical reasoning, the complexity of reasoning required is far less than that for the AI2 Science Questions dataset (Clark and Etzioni, 2016, Kenthavii et al., 2017). NewQA (Trischler et al., 2016) is another dataset that was created using crowd-sourcing; it utilizes passages from 10,000 news articles to create questions.

Most of the datasets mentioned above are closed domain, where the answer exists in a given snippet of text. On the other hand, in the open domain setting, the question-answer datasets are constructed to encompass the whole pipeline for question-answering, starting with the retrieval of relevant documents. SearchQA (Dunn et al., 2017) is an effort to create such a dataset; it contains 140K question-answer (QA) pairs. While the motivation was to create an open domain dataset, SearchQA provides text that contains “evidence” (as noted search results) and hence falls short of being a complete open-domain QA dataset. Triviqa (Joshi et al., 2017) is another reading comprehension dataset that contains 650K QA pairs with evidence.

Datasets created from standardized science tests offer some of the few existing examples of questions that require exploration of complex reasoning techniques to find solutions. A number of science-question focused datasets have been released over the past few years. The AI2 Science Question dataset was introduced by Clark (2015) along with the Aristo Framework, which we build off of. This dataset contains over 1,000 multiple choice questions from state and federal science questions for elementary and middle school students. A survey of the knowledge base requirements for accomplishing this task was performed by Clark et al. (2013), and concluded that advanced inference methods were necessary for many of the questions, as they could not be answered by simple fact-based retrieval. The SciQA Dataset (Wells et al., 2017) contains 13,679 crowd-sourced multiple choice science questions. To construct this dataset, workers were shown a passage and asked to construct a question along with correct and incorrect answer options. The dataset contains both the source passage as well as the question and answer options.

3 ARC Dataset Annotations

In previous work (Clark et al., 2018), the standardized test questions under consideration are split into various categories based on the kinds of knowledge and reasoning that are needed to answer those questions. The idea of classifying questions by these two types is central to the notion of standardized testing, which endeavors to test students on various kinds of knowledge, as well as various problem types and solution techniques. In accordance with this, Clark et al. (2018) provide preliminary definitions for knowledge and reasoning categories that can be employed by a QA system to solve a given question. These categories allow for the classification of questions, which makes it easier to partition them into sets to measure performance and improve solution strategies. In this work, we present an interface (c.f. Section 4) and annotation rules that seek to turn this classification of questions into a systematic process. In this section, we first discuss the classification types and associated annotation rules.

3.1 Knowledge Types

In most question-answering (QA) scenarios, the knowledge that is present with the system (or the agent) determines whether a given question can be answered. The full list of the revised knowledge labels (types)—along with the instructions given to annotators and respective exemplars from the ARC question set—is given in Table 1. The labels were given the following instructions at the beginning of the annotation process:

You are to answer the question, “In a perfect world given an ideal knowledge source, what types of knowledge would you as a human need to answer this question?” You are allowed to re-label the types for this type which will be recorded as an ordered list. You are to assign labels in the order of importance to answering the questions at hand.

The wording of the paragraph above is quite debatable. First, we make the non-trivial point that the kind of knowledge that is available determines the reasoning type to be employed, and eventually whether the given question can be answered or not. For example, the question:

**Question:** What is the first step of the process in the formation of sedimentary rocks?

**Options:**
(A) Erosion
(B) Deposition
(C) Compaction
(D) Crystallization

**Knowledge Label Instructions Example Question**

**Definition**
A question should be labeled definition if it requires you to know the definition of a term or ask only that definition to answer the question. What is a greenhouse effect?

**Purpose**
A question should be labeled purpose if answering it requires understanding the function of one or more terms in the question. What is the main function of the circulatory system?

**To answer this question, you are allowed to seek additional external resources**

**Type**
A question should be labeled type if answering it requires one or more terms in the question.

**Physical Model**
Any question that refers to a spatial/kinematic/physical model of the physical world in order to be answered.

<table>
<thead>
<tr>
<th>Knowledge Type</th>
<th>Instructions</th>
<th>Example Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>A question should be labeled definition if it requires you to know the definition of a term or ask only that definition to answer the question.</td>
<td>What is a greenhouse effect?</td>
</tr>
<tr>
<td>Purpose</td>
<td>A question should be labeled purpose if answering it requires understanding the function of one or more terms in the question.</td>
<td>What is the main function of the circulatory system?</td>
</tr>
<tr>
<td>Type</td>
<td>A question should be labeled type if answering it requires one or more terms in the question.</td>
<td>What is the first step of the process in the formation of sedimentary rocks?</td>
</tr>
<tr>
<td>Physical Model</td>
<td>Any question that refers to a spatial/kinematic/physical model of the physical world in order to be answered.</td>
<td>What is the first step of the process in the formation of sedimentary rocks?</td>
</tr>
</tbody>
</table>

**Knowledge Label Instructions Example Question**

**Definition**
A question should be labeled definition if it requires you to know the definition of a term or ask only that definition to answer the question. What is a greenhouse effect?

**Purpose**
A question should be labeled purpose if answering it requires understanding the function of one or more terms in the question. What is the main function of the circulatory system?

**To answer this question, you are allowed to seek additional external resources**

**Type**
A question should be labeled type if answering it requires one or more terms in the question.

**Physical Model**
Any question that refers to a spatial/kinematic/physical model of the physical world in order to be answered.

**Knowledge Label Instructions Example Question**

**Definition**
A question should be labeled definition if it requires you to know the definition of a term or ask only that definition to answer the question. What is a greenhouse effect?

**Purpose**
A question should be labeled purpose if answering it requires understanding the function of one or more terms in the question. What is the main function of the circulatory system?

**To answer this question, you are allowed to seek additional external resources**

**Type**
A question should be labeled type if answering it requires one or more terms in the question.

**Physical Model**
Any question that refers to a spatial/kinematic/physical model of the physical world in order to be answered.

**Knowledge Label Instructions Example Question**

**Definition**
A question should be labeled definition if it requires you to know the definition of a term or ask only that definition to answer the question. What is a greenhouse effect?

**Purpose**
A question should be labeled purpose if answering it requires understanding the function of one or more terms in the question. What is the main function of the circulatory system?

**To answer this question, you are allowed to seek additional external resources**

**Type**
A question should be labeled type if answering it requires one or more terms in the question.

**Physical Model**
Any question that refers to a spatial/kinematic/physical model of the physical world in order to be answered.

**Knowledge Label Instructions Example Question**

**Definition**
A question should be labeled definition if it requires you to know the definition of a term or ask only that definition to answer the question. What is a greenhouse effect?

**Purpose**
A question should be labeled purpose if answering it requires understanding the function of one or more terms in the question. What is the main function of the circulatory system?

**To answer this question, you are allowed to seek additional external resources**

**Type**
A question should be labeled type if answering it requires one or more terms in the question.

**Physical Model**
Any question that refers to a spatial/kinematic/physical model of the physical world in order to be answered.
Table 2: Reasoning type definitions and examples given to the annotators.

<table>
<thead>
<tr>
<th>Reasoning Label Instructions</th>
<th>Example Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct Answer Selection</td>
<td>A question should be labeled algebraic if answering it requires at least two distinct pieces of evidence. Examples are: (A) algebraic, (B) linguistic, (C) mathematical formula along with linguistic matching should be labeled algebraic, linguistic.</td>
</tr>
<tr>
<td>Linguistic Reasoning</td>
<td>If the Sun were larger, what would most likely also have to be true for Earth to remain liquid? (A) Earth would have to be farther from the Sun (B) Earth would have to be closer to the Sun (C) Earth would have to be larger.</td>
</tr>
<tr>
<td>Hypothetical / Counterfactual</td>
<td>The right hand side of the interface deals with the labeling process itself. There are two boxes for annotating knowledge and reasoning types respectively. The labels are populated from Table 1 and Table 2. The annotator can also provide optional instruction-based definitions for each class, as opposed to the single exemplars provided previously. We believe that greatly simplifies the annotation task for new annotators, since they no longer need to perform a preliminary manual analysis of the QA set in order to understand the distinctions between the classes. Second, we completely eliminated the Structure type – this is a very specific type of knowledge, and we believe it is not represented in any significant percentage in the current ARC QA set. Third, we rename some of the labels to bring them more in line with the specific properties of the knowledge that they are describing – for example, spatial / kinematic is renamed to Physical Model in our table.</td>
</tr>
</tbody>
</table>
also allows annotators to input their own custom queries. We found that reformulating the initial query significantly improved the quality of the retrieved context (results). While not the main focus of this work, we encouraged the annotators to mark the contexts (results) that they thought were relevant to answering the question at hand. For example, in Figure 1, the annotator came up with a novel query – ‘metals are solid at room temperatures’ – and also marked the relevant sentences which are needed to answer this question. Note that sometimes we need to reason over multiple sentences to arrive at the answer. For example, the question in Figure 1 can be answered by combining the first and third sentences in the ‘Relevant Results’ tab.

To quantitatively measure the efficacy of the annotated context, we evaluated 47 questions and their respective human-annotated relevant sentences with a pretrained DrQA model (Chen et al., 2017). We compared this to a baseline which only returned the sentences retrieved by using the text of the question plus given options as input queries. Since DrQA returns a span from the input sentences, we picked the multiple choice option that maximally overlapped with the returned answer span. Our baseline results are 7 correct out of 47 questions. With the annotated context, the performance increased to 27 correctly answered questions - a 42% increase in accuracy. Encouraged by these results, we posit that the community should focus a lot of attention on improving the retrieval portions of the various QA systems available; we think that annotated context will certainly help in training a better ranker.

6 Results
Each of the team members were given access to the labeling interface (which includes the question, answers, query search results and more information). One roadblock for training a sophisticated retriever is the lack of available training data which annotates the relevance of a retrieved context with respect to the question. We believe our annotated retrieval data can be used to train a better ranker/retriever.

The underlying retriever in our interface is a simple Elasticsearch, similar to the one used by Clark et al. (2018). The interface is populated by default with the top ranked sentences that are retrieved with the given question as the input query. However, we noticed that results thus retrieved were often irrelevant to answering the question. To address this, our labeling interface also uses a small piece of text (e.g. paragraph) (Wang and Ji wrap, 2016; Xiong et al., 2016; Seo et al., 2016; Lee et al., 2016, inter alia). However, most work in open-domain settings (Chen et al., 2017; Clark and Gardner, 2017; Wang et al., 2018) only uses a simple retriever (such as TF-IDF based). As a result, there is a notable decrease in the performance of the QA system. One roadblock for training a sophisticated retriever is the lack of available training data which annotates the relevance of a retrieved context with respect to the question. We believe our annotated retrieval data can be used to train a better ranker/retriever.

The underlying retriever in our interface is a simple Elasticsearch, similar to the one used by Clark et al. (2018). The interface is populated by default with the top ranked sentences that are retrieved with the given question as the input query. However, we noticed that results thus retrieved were often irrelevant to answering the question. To address this, our labeling interface also allows annotators to input their own custom queries. We found that reformulating the initial query significantly improved the quality of the retrieved context (results). While not the main focus of this work, we encouraged the annotators to mark the contexts (results) that they thought were relevant to answering the question at hand. For example, in Figure 1, the annotator came up with a novel query – ‘metals are solid at room temperatures’ – and also marked the relevant sentences which are needed to answer this question. Note that sometimes we need to reason over multiple sentences to arrive at the answer. For example, the question in Figure 1 can be answered by combining the first and third sentences in the ‘Relevant Results’ tab.

To quantitatively measure the efficacy of the annotated context, we evaluated 47 questions and their respective human-annotated relevant sentences with a pretrained DrQA model (Chen et al., 2017). We compared this to a baseline which only returned the sentences retrieved by using the text of the question plus given options as input queries. Since DrQA returns a span from the input sentences, we picked the multiple choice option that maximally overlapped with the returned answer span. Our baseline results are 7 correct out of 47 questions. With the annotated context, the performance increased to 27 correctly answered questions - a 42% increase in accuracy. Encouraged by these results, we posit that the community should focus a lot of attention on improving the retrieval portions of the various QA systems available; we think that annotated context will certainly help in training a better ranker.

6 Results
Each of the team members were given access to the labeling interface (which includes the question, answers, query search results and more information). One roadblock for training a sophisticated retriever is the lack of available training data which annotates the relevance of a retrieved context with respect to the question. We believe our annotated retrieval data can be used to train a better ranker/retriever.

The underlying retriever in our interface is a simple Elasticsearch, similar to the one used by Clark et al. (2018). The interface is populated by default with the top ranked sentences that are retrieved with the given question as the input query. However, we noticed that results thus retrieved were often irrelevant to answering the question. To address this, our labeling interface also allows annotators to input their own custom queries. We found that reformulating the initial query significantly improved the quality of the retrieved context (results). While not the main focus of this work, we encouraged the annotators to mark the contexts (results) that they thought were relevant to answering the question at hand. For example, in Figure 1, the annotator came up with a novel query – ‘metals are solid at room temperatures’ – and also marked the relevant sentences which are needed to answer this question. Note that sometimes we need to reason over multiple sentences to arrive at the answer. For example, the question in Figure 1 can be answered by combining the first and third sentences in the ‘Relevant Results’ tab.

To quantitatively measure the efficacy of the annotated context, we evaluated 47 questions and their respective human-annotated relevant sentences with a pretrained DrQA model (Chen et al., 2017). We compared this to a baseline which only returned the sentences retrieved by using the text of the question plus given options as input queries. Since DrQA returns a span from the input sentences, we picked the multiple choice option that maximally overlapped with the returned answer span. Our baseline results are 7 correct out of 47 questions. With the annotated context, the performance increased to 27 correctly answered questions - a 42% increase in accuracy. Encouraged by these results, we posit that the community should focus a lot of attention on improving the retrieval portions of the various QA systems available; we think that annotated context will certainly help in training a better ranker.

6 Results
Each of the team members were given access to the labeling interface (which includes the question, answers, query search results and more information). One roadblock for training a sophisticated retriever is the lack of available training data which annotates the relevance of a retrieved context with respect to the question. We believe our annotated retrieval data can be used to train a better ranker/retriever.

The underlying retriever in our interface is a simple Elasticsearch, similar to the one used by Clark et al. (2018). The interface is populated by default with the top ranked sentences that are retrieved with the given question as the input query. However, we noticed that results thus retrieved were often irrelevant to answering the question. To address this, our labeling interface also allows annotators to input their own custom queries. We found that reformulating the initial query significantly improved the quality of the retrieved context (results). While not the main focus of this work, we encouraged the annotators to mark the contexts (results) that they thought were relevant to answering the question at hand. For example, in Figure 1, the annotator came up with a novel query – ‘metals are solid at room temperatures’ – and also marked the relevant sentences which are needed to answer this question. Note that sometimes we need to reason over multiple sentences to arrive at the answer. For example, the question in Figure 1 can be answered by combining the first and third sentences in the ‘Relevant Results’ tab.

To quantitatively measure the efficacy of the annotated context, we evaluated 47 questions and their respective human-annotated relevant sentences with a pretrained DrQA model (Chen et al., 2017). We compared this to a baseline which only returned the sentences retrieved by using the text of the question plus given options as input queries. Since DrQA returns a span from the input sentences, we picked the multiple choice option that maximally overlapped with the returned answer span. Our baseline results are 7 correct out of 47 questions. With the annotated context, the performance increased to 27 correctly answered questions - a 42% increase in accuracy. Encouraged by these results, we posit that the community should focus a lot of attention on improving the retrieval portions of the various QA systems available; we think that annotated context will certainly help in training a better ranker.
In addition to Fleiss’ $\kappa$ we also use the Kemeny voting rule (Kemeny, 1959) to measure the consensus by the annotators. The Kemeny voting rule minimizes the Kendall Tau distance between the output ordering and the ordering of all annotators. One theory of voting (aggregation) is that there is a true or correct ordering and all voters provide a noisy observation of the ground truth. This method of thinking is largely credited to Condorcet (de Caritat, 1785; Young, 1988) and there is recent work in characterizing other voting rules as maximum likelihood estimators (MLEs) (Cottinez et al., 2009). The Kemeny voting rule is the MLE of the Condorcet Noise Model, in which pairwise inferences of the preference order happen uniformly at random (Young, 1988). Hence, if we assume all annotators make pairwise errors uniformly at random then Kemeny is the MLE of label orders they report.

**Reasoning Labels.** The reasoning labels tell a very different story from the knowledge labels. The agreement was $\kappa = -0.683$, which indicates that raters did not agree above chance on their labels. Strong evidence for this comes from the fact that only 27/102 questions had a consensus label. This may be due to the fact that we allow multiple labels, and the annotators simply disagree on the order of the labels. However, the score of the consensus ranking for each question is 6.57, which indicates that on average the ordering of the labels is quite far apart.

Considering the histogram in Figure 3, we see that the $qn$ logic, linguistic, and explanation are the most frequent label types; this may indicate that getting better at understanding the questions themselves could lead to a big boost for reasoners. For Figure 4, we have merged the first and second label (if present) for all annotators. Now, the set of all possible labels is all singletons as well as all pairs of labels. Comparing this histogram to the one in Figure 3, we see that while linguistic and explanation remain somewhat unchanged, the $qn$ logic label becomes very spread out across the types. This is more support for our hypothesis that annotators may be disagreeing on the ordering of the labels, rather than the content itself.

### Table 4: Pairwise inter-rater agreement along with the mean and Fleiss’ $\kappa$ for survey responses.

<table>
<thead>
<tr>
<th>Knowledge Type</th>
<th>Label</th>
<th>Apache</th>
<th>Majority</th>
<th>Consensus</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic facts</td>
<td>logic</td>
<td>125</td>
<td>89</td>
<td>64</td>
</tr>
<tr>
<td>linguistic</td>
<td>expletive</td>
<td>14</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>cause</td>
<td>27</td>
<td>13</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>experiment</td>
<td>15</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>purpose</td>
<td>20</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>physical</td>
<td>18</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

Fleiss’ $\kappa$ = 0.342

### Table 5: Accuracy on our subset of ARC Challenge Set questions, partitioned based on the first label from the Kemeny ordering of the reasoning and knowledge type annotations, respectively: 1/3 partial credit is given when the correct answer is in the set of $k$ selected answers. The number of questions assigned each primary label is indicated by (#).

<table>
<thead>
<tr>
<th>Knowledge Type</th>
<th>Label</th>
<th>Text Search</th>
<th>word2vec</th>
<th>SemanticILP</th>
<th>DecompAttn</th>
<th>DGEM</th>
<th>BiDAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic facts</td>
<td>logic</td>
<td>1.7</td>
<td>2.1</td>
<td>1.3</td>
<td>2.0</td>
<td>1.7</td>
<td>2.8</td>
</tr>
<tr>
<td></td>
<td>expletive</td>
<td>1.5</td>
<td>1.5</td>
<td>1.3</td>
<td>1.7</td>
<td>1.8</td>
<td>2.2</td>
</tr>
<tr>
<td></td>
<td>cause</td>
<td>1.5</td>
<td>1.4</td>
<td>1.1</td>
<td>1.6</td>
<td>1.5</td>
<td>1.8</td>
</tr>
<tr>
<td></td>
<td>experiment</td>
<td>1.4</td>
<td>1.1</td>
<td>1.0</td>
<td>1.2</td>
<td>1.5</td>
<td>1.6</td>
</tr>
<tr>
<td></td>
<td>purpose</td>
<td>1.4</td>
<td>1.3</td>
<td>1.0</td>
<td>1.4</td>
<td>1.5</td>
<td>1.6</td>
</tr>
<tr>
<td></td>
<td>physical</td>
<td>1.1</td>
<td>1.0</td>
<td>1.0</td>
<td>1.2</td>
<td>1.5</td>
<td>1.6</td>
</tr>
</tbody>
</table>

### Acknowledgments

We would like to thank Salim Roukos for helpful suggestions on the annotation process, and Daniel Khashabi for assistance with SemanticILP. This work is funded in part by International Business Machines Corporation Cognitive Horizons Network agreement number W166853 and The Center for Data Science.
References


Peter Clark and Oren Etzioni. 2016. My computer is an honor student but how intelligent is it? standardized tests as a measure of ai. AI Magazine, 37(1):5–12.


Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding

Kexin Yi∗
Harvard University

Jiajun Wu∗
MIT CSAIL

Chuang Gan
MIT-IBM Watson AI Lab

Antonio Torralba
MIT CSAIL

Pushmeet Kohli
DeepMind

Joshua B. Tenenbaum
MIT CSAIL

Abstract

We marry two powerful ideas: deep representation learning for visual recognition and language understanding, and symbolic program execution for reasoning. Our neural-symbolic visual question answering (NS-VQA) system first recovers a structural scene representation from the image and a program trace from the question. It then executes the program on the scene representation to obtain an answer. Incorporating symbolic structure as prior knowledge offers three unique advantages. First, executing programs on a symbolic space is more robust to long program traces; our model can solve complex reasoning tasks better, achieving an accuracy of 99.8% on the CLEVR dataset. Second, the model is more data- and memory-efficient: it performs well after learning on a small number of training data; it can also encode an image into a compact representation, requiring less storage than existing methods for offline question answering. Third, symbolic program execution offers full transparency to the reasoning process; we are thus able to interpret and diagnose each execution step.

1 Introduction

Looking at the images and questions in Figure 1, we instantly recognize objects and their attributes, parse complicated questions, and leverage such knowledge to reason and answer the questions. We can also clearly explain how we reason to obtain the answer. Now imagine that you are standing in front of the scene, eyes closed, only able to build your scene representation through touch. Not surprisingly, reasoning without vision remains effortless. For humans, reasoning is fully interpretable, and not necessarily interwoven with visual perception.

The advances in deep representation learning and the development of large-scale datasets [Malinowski and Fritz, 2014, Antol et al., 2015] have inspired a number of pioneering approaches in visual question-answering (VQA), most trained in an end-to-end fashion [Yang et al., 2016]. Though innovative, pure neural net–based approaches often perform less well on challenging reasoning tasks. In particular, a recent study [Johnson et al., 2017a] designed a new VQA dataset, CLEVR, in which each image comes with intricate, compositional questions generated by programs, and showed that state-of-the-art VQA models did not perform well.

Later, Johnson et al. [2017b] demonstrated that machines can learn to reason by wiring in prior knowledge of human language as programs. Specifically, their model integrates a program generator that infers the underlying program from a question, and a learned, attention-based executor that runs the program on the input image. Such a combination achieves very good performance on the CLEVR dataset.

∗ indicates equal contributions

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.
dataset, and generalizes reasonably well to CLEVR-Humans, a dataset that contains the same images as CLEVR but now paired with human-generated questions. However, their model still suffers from two limitations: first, training the program generator requires many annotated examples; second, the behaviors of the attention-based neural executor are hard to explain. In contrast, we humans can reason on CLEVR and CLEVR-Humans even with a few labeled instances, and we can also clearly explain how we do it.

In this paper, we move one step further along the spectrum of learning vs. modeling, proposing a neural-symbolic approach for visual question answering (NS-VQA) that fully decouples vision and language understanding from reasoning on the dataset, exploiting powerful tools for parsing—inferring structural, object-based scene representation from images, and generating programs from questions. We then incorporate a symbolic program executor that, complementary to the neural parser, runs the program on the scene representation to obtain an answer.

The combination of deep recognition modules and a symbolic program executor offers three unique advantages. First, the use of symbolic representation offers robustness to long, complex program traces. It also reduces the need of training data. On the CLEVR dataset, our method is trained on questions with 270 program annotations plus 4K images, and is able to achieve a near-perfect accuracy of 99.8%.

Second, both our reasoning module and visual scene representation are light-weighted, requiring minimal computational and memory cost. In particular, our compact structural image representation requires much less storage during reasoning, reducing the memory cost by 99% compared with other state-of-the-art algorithms.

Third, the use of symbolic scene representation and program traces forces the model to accurately recover underlying programs from questions. Together with the fully transparent and interpretable nature of symbolic representations, the reasoning process can be analyzed and diagnosed by-step-by-step.

2 Related Work

Structural scene representation. Our work is closely related to research on learning an interpretable, disentangled representation with a neural network [Kalantari et al., 2015, Yang et al., 2015, Wu et al., 2017]. For example, Kalantari et al. [2015] proposed convolutional inverse graphics networks that learn to infer the pose and lighting of a face. Yang et al. [2015] explored learning disentangled representations of pose and content from chair images. They have also been working on learning disentangled representation without direct supervision [Higgins et al., 2018, Siddharth et al., 2018, Velas et al., 2018], some with sequential generative models [Eslami et al., 2016, Bo et al., 2015]. In a broader view, our model also relates to the field of “vision as inverse graphics” [Yuille and Kersten, 2000]. Our NS-VQA model builds upon the structural scene representation [Wu et al., 2017] and explores how it can be used for visual reasoning.

Program induction from language. Recent papers have explored using program search and neural networks to recover programs from a domain-specific language [Balog et al., 2017, Neelakantan et al., 2016, Parikh et al., 2017]. For sentences, semantic parsing methods map them to logical forms via a knowledge base or a program [Hirant et al., 2013, Liang et al., 2013, Vinyals et al., 2015, Gou et al., 2017]. In particular, Andreas et al. [2016] attempted to use the latent structure in language to help question answering and reasoning. Roth et al. [2017] studied the use of formal programs in modeling human questions, and Goldman et al. [2017] used abstract examples to build weakly-supervised semantic parsers.

Visual question answering. Visual question answering (VQA) [Malinowski and Fritz, 2014, Antol et al., 2015] is a versatile and challenging test bed for AI systems. Compared with the well-studied text-based question answering, VQA emerges by its requirement on both semantic and visual understanding. There have been numerous papers on VQA, among which some explicitly used structural knowledge to help reasoning [Wang et al., 2017]. Current leading approaches are based on neural attention [Yang et al., 2016, Lu et al., 2016], which draw inspiration from human perception and learn to attend the visual components that serve as informative evidence to the question. Nonetheless, Jabri et al. [2016] recently proposed a remarkably simple yet effective classification baseline. Their system directly extracts visual and text features from whole images and questions, concatenates them, and trains multi-class classifiers to select answers. This paper, among others [Goyal et al., 2017], reveals potential caveats in the proposed VQA systems—models are overfitting dataset biases.

Visual reasoning. Johnson et al. [2017a] built a new VQA dataset, named CLEVR, carefully controlling the potential bias and benchmarking how well models reason. Their subsequent model achieved good results on CLEVR by combining a recurrent program generator and an attentive execution engine [Johnson et al., 2017b]. There have been other end-to-end neural models that has achieved nice performance on the dataset, exploiting various attention structures and accuracies on object relations [Hudson and Manning, 2018, Santoro et al., 2017, Hu et al., 2017, Perez et al., 2018, Zhu et al., 2017]. More recently, several papers have proposed to directly incorporate the syntactic and logical structures of the reasoning task to the attentive module network’s architecture for reasoning. These structures include the underlying functional programs [Mascharka et al., 2018, Suzer et al., 2018] and dependency trees [Cao et al., 2018] of the input question. However, training of the models relies heavily on these extra signals.

From a broader perspective, Misra et al. [2017] explored learning to reason by asking questions and Bisk et al. [2017] studied spatial reasoning in a 3D blocks world. Recently, Aditya et al. [2018] incorporated probabilistic soft logic into a neural attention module and obtained some interpretability of the model, and Gan et al. [2017] learned to associate image segments with questions. Our model moves along this direction further by modeling the entire scene into an object-based, structural representation, and integrating it with a fully transparent and interpretable symbolic program executor.

3 Approach

Our NS-VQA model has three components: a scene parser (de-renderer), a question parser (program generator), and a program executor. Given an image-question pair, the scene parser de-renders the image to obtain a structural scene representation (Figure 2-b), the question parser generates a hierarchical program from the question (Figure 2-d), and the executor runs the program on the structural representation to obtain an answer (Figure 2-b).

Our scene parser recovers a structural and disentangled representation of the scene in the image (Figure 2-a), based on which we can perform fully interpretable symbolic reasoning. The parser takes a two-step, segment-based approach for de-renders: it first generates a number of segment proposals (Figure 2-b), and for each segment, classifies the object and its attributes. The final, structural scene representation is disentangled, compact, and rich (Figure 2-c).

The question parser maps an input question in natural language (Figure 2-d) to a latent program (Figure 2-e). The program has a hierarchy of functional modules, each fulfilling an independent operation on the scene representation. Using a hierarchical program as our reasoning backbone naturally supplies compositionality and generalization power.

The program executor takes the output sequence from the question parser, applies these functional modules on the abstract scene representation of the input image, and generates the final answer (Figure 2-b). The executable program performs purely symbolic operations on its input throughout.

Figure 1: Human reasoning is interpretable and disentangled: we first draw abstract knowledge of the scene via visual perception and then perform logic reasoning on it. This enables compositional, accurate, and generalizable reasoning in rich visual contexts.
The encoder is a bidirectional LSTM [Hochreiter and Schmidhuber, 1997] that takes as input a sequence of tokens from the input program and outputs an encoded vector \( e_i \) at each time step. The decoder word embedding \( \Phi_i \) is the joint trained encoder word embedding, \( \Phi_i = \Phi \left( \Phi_p \left( y_i \right) \right) \), where \( \Phi_p \) is the self-attention layer to obtain a context vector \( \alpha_t \) as weighted sum of the encoded states via \( \alpha_t = \text{softmax}(W_{\text{att}} \cdot e_t) \).

Here, \( \Phi_p \) is the decoder word embedding. For simplicity, we set the dimensions of vectors \( q_i, e_i, \alpha_t \) to be the same and let the attention weight matrix \( W_{\text{att}} \) to be an identity matrix. Finally, the context vector, together with the decoder output, is passed to a fully connected layer with softmax activation to obtain the distribution for the predicted token \( y \) by \( \text{softmax}(W_p \cdot e_t) \).

Both the encoder and decoder have two hidden layers with a 256-dim hidden vector. We set the dimensions of both the encoder and decoder to be an identity matrix. Finally, the context vector, together with the decoder output, is passed to a fully connected layer with softmax activation to obtain the distribution for the predicted token \( y \) by \( \text{softmax}(W_p \cdot e_t) \).

The encoder is a bidirectional LSTM, and the decoder is a unidirectional LSTM [Hochreiter and Schmidhuber, 1997] that takes as input \( y_i \) at each time step and outputs a sequence of tokens from the input program. The decoder is a unidirectional LSTM, and the encoder is a bidirectional LSTM [Hochreiter and Schmidhuber, 1997].

**Table 1:** (Not shown in the image) Our model (NS-VQA) outperforms current state-of-the-art methods on CLEVR and achieves near-perfect question answering accuracy. The question-program pairs used for pretraining our model are uniformly drawn from the 90-question families of the dataset: 90, 180, 270 programs correspond to 1, 2, 3 samples from each family respectively. (*) trains on all program annotations (700K).

**Section 3.2 Training Paradigm**

**Scene Parsing.** Our implementation of the object proposal network (Mask R-CNN) is based on "Detectron" [Girshick et al., 2018]. We use ResNet-50 FPN [Lin et al., 2017] as the backbone and train the model for 30,000 iterations with eight images per batch. Please refer to He et al. [2017] and Girshick et al. [2018] for more details. Our feature extraction network outputs the values of continuous attributes. We train the network on the proposed object segments computed from the training data using the mean square error as loss function for 30,000 iterations with learning rate 0.002 and batch size 50. Both networks of our scene parser are trained on 4,000 generated CLEVR images.

**Reasoning.** We adopt the following two-step procedure to train the question parser to learn the mapping from a question to a program. First, we select a small number of ground truth question-program pairs from the training set. We then pair each question with its deterministic program parser, and use REINFORCE [Williams, 1992] to fine-tune the parser on a larger set of question-answer pairs, using only the correctness of the execution result as the reward signal. During supervised pretraining, we train with learning rate \( \eta \) for 20,000 iterations. For reinforcement learning, we set the learning rate to be \( \eta \times 10^{-5} \) and run at most 2M iterations with early stopping. The reward is maximized over a constant baseline with a decay weight \( \lambda \) to reduce variance. Batch size is fixed to 64 for both training stages. All our models are implemented in PyTorch.

**Section 4 Evaluations**

We demonstrate the following advantages of our disentangled structural scene representation and symbolic execution engine. First, our model can learn from a small number of training data and outperform the current state-of-the-art methods while precisely recovering the latent programs (Sections 4.1). Second, our model generalizes well to other question styles (Sections 4.3), attribute combinations (Sections 4.4), and visual context (Section 4.5).
4.1 Data-Efficient, Interpretable Reasoning

Setup. We evaluate our NS-VQA on CLEVR [Johnson et al., 2017b]. The dataset includes synthetic images of 3D primitives with multiple attributes—shape, color, material, size, and 3D coordinates. Each image has a set of questions, each of which associates with a program (a set of symbolic modules) generated by machines based on 90 logic templates.

Our structural scene representation for a CLEVR image characterizes the objects in it, each labeled with its shape, size, color, material, and 3D coordinates (see Figure 9). We evaluate our method’s performance on the validation set under various supervision signals for training, including the numbers of ground-truth programs used for pretraining and question-answer pairs for REINFORCE. Results are compared with other state-of-the-art methods including the IEP baseline [Johnson et al., 2017b]. We do not assess the correctness of the answer obtained by our model, but also well how it recovers the underlying program. An interpretable model should be able to output the correct program in addition to the correct answer.

Results. Quantitative results on the CLEVR dataset are summarized in Table 1. Our NS-VQA achieves near-perfect accuracy and outperforms other methods on all five question types. We first pretrain the question parser on 270 annotated programs sampled across the 90 question templates (3 questions per template), a number below the weekly supervised limit suggested by Johnson et al. [2017b] (9k), and then run REINFORCE on all the question-answer pairs. Repeated experiments starting from different sets of programs show a standard deviation of less than 0.1 percent on the results for 270 pretraining programs (and beyond). The variance is larger when we train our model with 15,000 programs (90 and 180). The reported numbers are the mean of three runs.

We further investigate the data-efficiency of our method with respect to both the number of programs used for pretraining and the overall question-answer pairs used in REINFORCE. Figure 10a shows the result when we vary the number of pretraining programs. NS-VQA outperforms the IEP baseline under various conditions, even with a weaker supervision during REINFORCE (2k and 9k question-answer pairs in REINFORCE). The number of question-answer pairs can be further reduced by pretraining the model on a larger set of annotated programs. For example, our model achieves the same near-perfect accuracy of 99.9% with 9k question-answer pairs with annotated programs for both pretraining and REINFORCE.

Figure 10b compares how well our NS-VQA recovers the underlying programs compared to the IEP model. IEP starts to capture the true programs when trained with over 1k programs, and only recovers half of the programs with 9k programs. Qualitative examples in Figure 10c demonstrate that IEP tends to take a long wrong program that leads to the correct answer. In contrast, our model achieves 90% program accuracy with 500 annotations, and performs almost perfectly on both question answering and program recovery with 9k programs.

Figure 10c shows the QA accuracy vs. the number of questions and answers used for training, where our NS-VQA has the highest performance under all conditions. Among the baseline methods we compare with, MAC [Hudson and Manning, 2018] obtains high accuracy with zero program annotations; in comparison, our method needs to be pretrained on 270 program annotations, but requires fewer question-answer pairs to reach similar performance. Our model also requires minimal memory for offline question answering: the structural representation of each image only occupies less than 100 bytes; in comparison, attention-based methods like IEP require storing either the original image or its feature maps, taking at least 20k bytes per image.

4.2 Generalizing to Unseen Attribute Combinations

Recent neural reasoning models have achieved impressive performance on the original CLEVR QA task [Johnson et al., 2017b, Mascharak et al., 2018, Perez et al., 2018], but they generalize less well across biased dataset splits. This is revealed on the CLEVR-CoGenT dataset [Johnson et al., 2017a], a benchmark designed specifically for testing models’ generalization to novel attribute combinations.

Setup. The CLEVR-CoGenT dataset is derived from CLEVR and separated into two biased splits: split A only contains cubes that are either gray, blue, brown or yellow, and cylinders that are red, green, purple or cyan; split B has the opposite color shape pairs for cubes and cylinders. Both split A contains spheres of any color. Split A has 70k images and 70k questions for training and both splits have 15k images and 150k questions for evaluation and testing. The desired behavior of a generalizable model is to perform equally well on both splits while only trained on split A.

Result. Table 2a shows the generalization results with a few interesting findings. The vanilla NS-VQA trained purely on split A and fine-tuned purely on split B (1000 images) does not generalize as well as the state-of-the-art. We observe that this is because of the bias in the attribute recognition network of the scene parser, which learns to classify object shape based on color. NS-VQA works well after we fine-tune it on data from both splits (400 A, 1000 B). Here, we only fine-tune the attribute recognition network with annotated images from split B, but no questions or programs; thanks to the disentangled pipeline and symbolic scene representation, our question parser and executor are not overfitting to particular splits. To validate this, we train a separate shape recognition network that takes gray-scale but not color images as input (NS-VQA-Gray). The augmented model works well on both splits without seeing any data from split B. Further, with an image parser trained on the original condition (i.e., the same as in CLEVR), our question parser and executor also generalize well across splits (NS-VQA-AQ0).
<table>
<thead>
<tr>
<th>Methods</th>
<th>Fine-tuned # Programs</th>
<th>NS-VQA # Programs</th>
<th>IEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNNA-LESTM+SA</td>
<td>A: 68.2, B: 68.7</td>
<td>100</td>
<td>38.7</td>
</tr>
<tr>
<td>IEP (3K programs)</td>
<td>B: 75.7, 75.8</td>
<td>200</td>
<td>40.1</td>
</tr>
<tr>
<td>CNNA-LESTM+ILFM</td>
<td>A: 81.1, B: 90.9</td>
<td>500</td>
<td>49.2</td>
</tr>
<tr>
<td>Thirdeye</td>
<td>B: 96.9, 96.3</td>
<td>1K</td>
<td>63.4</td>
</tr>
<tr>
<td>NS-VQA (ours)</td>
<td>99.8, 63.9</td>
<td>18K</td>
<td>67.8</td>
</tr>
<tr>
<td>NS-VQA (ours) + Gray</td>
<td>99.8, 65.9</td>
<td></td>
<td>66.6</td>
</tr>
<tr>
<td>NS-VQA (ours) + Grayscale</td>
<td>99.8, 98.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NS-VQA (ours) + Activity</td>
<td>99.8, 99.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a) Generalizing results on CLEVR-CogQA7.

Table 2: Generalizing to unseen attribute compositions and question styles. (a) Our image parser is trained on 4,000 synthetic images from split A and fine-tuned on 1,000 images from split B. The question parser is only trained on split A starting from 500 programs. Baseline methods are fine-tuned on 3K images plus 30K questions from split B. NS-VQA + Gray adopts a gray channel in the image parser for shape recognition and NS-VQA + Grayscale uses an image parser trained from the original images from CLEVR. Please see text for more details. (b) Our model outperforms IEP on CLEVR-Humans under various training conditions.

4.3 Generalizing to Questions from Humans

Our model also enables efficient generalization toward more realistic question styles over the same logic domain. We evaluate this on the CLEVR-Humans dataset, which includes human-generated questions on CLEVR images (see Johnson et al. [2017b] for details). The questions follow a real-life human conversation style without a regular structural expression.

Setup. We adopt a training paradigm for CLEVR-Humans similar to the original CLEVR dataset: we first pretrain the model with a limited number of programs from CLEVR, and then fine-tune it on CLEVR-Humans with REINFORCE. We initialize the encoder word embedding by the GloVe word vectors (Pennington et al., 2014) and keep it fixed during pretraining. The REINFORCE stage lasts for at most 1M iterations; early stop is applied.

Results. The results on CLEVR-Humans are summarized in Table 3b. Our NS-VQA outperforms IEP on CLEVR-Humans by a considerable margin under small amount of annotated programs. This shows our structural scene representation and symbolic program executor helps to exploit the strong exploitation power of REINFORCE, and also demonstrates the model’s generalizability across different question styles.

4.4 Extending to New Scene Context

Structural scene representation and symbolic programs can also be extended to other visual and contextual scenarios. Here we show results on reasoning tasks from the Minecraft world.

Setup. We now consider a new dataset where objects and scenes are taken from Minecraft and therefore have drastically different scene context and visual appearance. We use the dataset generation tool provided by Wu et al. (2017) to render 10,000 Minecraft scenes, building upon the Malmo interface (Johnson et al., 2018). Each image consists of 3 to 6 objects, and each object is sampled from a set of 12 entities. We use the same configuration details as suggested by Wu et al. (2017). Our structural representation has the following fields for each object: category (12-dim), position in the 2D plane (2-dim, [x, y]), and the direction the object faces (front, back, left, right). Each object is thus encoded as a 18-dim vector.

We generate diverse questions and programs associated with each Minecraft image based on the objects’ categorical and spatial attributes (position, direction). Each question is composed as a hierarchy of three families of basic questions: first, querying object attributes (class, location, directions); second, counting the number of objects satisfying certain constraints; third, verifying if an object has a certain property. Our dataset differs from CLEVR primarily in two ways: Minecraft hosts a larger set of 3D objects with richer image content and visual appearance; our questions and programs involve hierarchical attributes. For example, a “wolf” and a “pig” are both “animals”, and an “animal” and a “tree” are both “creatures”. We use the first 9,000 images with 88,109 questions for training and the remaining 1,000 images with 9,761 questions for testing. We follow the same recipe as described in Section 3.2 for training on Minecraft.

Results. Quantitative results are summarized in Table 5b. The overall behavior is similar to that on the CLEVR dataset, except that reasoning on Minecraft generally requires weaker initial program signals. Figure 5 shows the results on three test images: our NS-VQA finds the correct answer and recovers the correct program under the new scene context. Also, most of our model’s wrong answers on this dataset are due to errors in perceiving heavily occluded objects, while the question parser still preserves its power to parse input questions.

5 Discussion

We have presented a novel, neural-symbolic approach for VQA that disentangles reasoning from visual perception and language understanding. Our model uses deep learning for inverse graphics and inverse language modeling—recognizing and characterizing objects in the scene; it then uses a symbolic program executor to reason and answer questions.

We see our research suggesting a possible direction to unify two powerful ideas: deep representation learning and symbolic program execution. Our model connects to, but also differs from the recent pure deep learning approaches for visual reasoning. Wiring in symbolic representation as prior knowledge increases performance, reduces the need for annotated data and for memory significantly, and makes reasoning fully interpretable.

A common criticism about symbolic approaches is on generalization. Especially, building structured representation for general scenes can be challenging due to the widely known semantic gap. This is less a problem for approaches like ours, however, as they leverage deep networks for scene recognition. Beyond supervised learning, some recent papers have made inspiring attempts to explore how concepts naturally emerge during unsupervised learning [Bilgic et al., 2017]. We see integrating our model with these approaches a promising future direction.

Acknowledgements

We thank Rayuan Mao, Karthik Narasimhan, and Jon Gauthier for helpful discussions and suggestions. We also thank Drew A. Hudson for sharing experimental results for comparison. This work is in part supported by ONR MURI N00014-16-1-007, the Center for Brain, Minds, and Machines (CBMM), HMI Research, and Facebook.
Section 2
Learning, Knowledge and Reasoning

References


Chuan Guo, Yuandong Tian, Haoxiang Li, Chen Sun, and Boqing Gong. Vqa: Linking segmentations to questions and answers for supervised attention in vqa and question-focused semantic segmentation. In CVPR, 2017.


Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Inferring and executing programs for visual reasoning. In ICCV, 2017b.


Optimal Action
Reinforcement learning and planning are closely related: both areas of AI deal with the process of guiding an agent, situated in a dynamic environment, to achieve a set of predefined goals. However, there are also key differences. In reinforcement learning, the objective is to maximize the reward through experience and through interaction with the environment, while in planning, the objective is to find sequences of actions that lead from an initial state to a given goal. IBM has a long research history in both of these fields, since the days of TD-Gammon in 1992 and Deep Blue beating Garry Kasparov at chess in 1996. To date, this is still a very active research area within IBM Research.

We are currently seeing a resurgence of reinforcement learning in the AI community due to recent achievements in AI for game-playing (e.g. Go), but this technique will likely have applications in many other contexts. We, and other members of the AI community, are actively exploring additional applications.

This chapter highlights our papers on reinforcement learning and planning, including: different approaches of learning a hierarchy of actions or planning on multiple time scales [13] [14], a framework to assist software developers in modeling and solving planning problems within their applications [15], and a demonstration of how planning algorithms can be used to solve inference problems [16].

"That which we persist in doing becomes easier – not that the nature of the task has changed, but our ability to do has increased."

— Ralph Waldo Emerson

---

**Selected Publications**

13. Eigenoption Discovery Through The Deep Successor Representation
   p. 198

14. Learning Abstract Options
   p. 222

15. Semi Black-Box: Rapid Development of Planning Based Solutions
   p. 248

16. From Stochastic Planning to Marginal MAP
   p. 258
EIGENOPTION DISCOVERY THROUGH THE DEEP SUCCESSOR REPRESENTATION

Marlos C. Machado1, Clemens Rosenbaum2, Xiaoxiao Guo3
Miao Liu4, Gerald Tesauro4, Murray Campbell3
1 University of Alberta, Edmonton, AB, Canada
2 University of Massachusetts, Amherst, MA, USA
3 IBM Research, Yorktown Heights, NY, USA

ABSTRACT
Options in reinforcement learning allow agents to hierarchically decompose a task into subtasks, having the potential to speed up learning and planning. However, autonomously learning effective sets of options is still a major challenge in the field. In this paper we focus on the recently introduced idea of using representation learning methods to guide the option discovery process. Specifically, we look at eigenoptions, options obtained from representations that encode diffusive information flow in the environment. We extend the existing algorithms for eigenoption discovery to settings with stochastic transitions and in which handcrafted features are not available. We propose an algorithm that discovers eigenoptions while learning non-linear state representations from raw pixels. It exploits recent successes in the deep reinforcement learning literature and the equivalence between proto-value functions and the successor representation. We use traditional tabular domains to provide intuition about our approach and Atari 2600 games to demonstrate its potential.

1 INTRODUCTION
Sequential decision making usually involves planning, acting, and learning about temporally extended courses of actions over different time scales. In the reinforcement learning framework, options are a well-known formalization of the notion of actions extended in time; and they have been shown to speed up learning and planning when appropriately defined (e.g., Brunskill & Li, 2014; Guo et al., 2017; Solway et al., 2014). In spite of that, autonomously identifying good options is still an open problem. This problem is known as the problem of option discovery. Option discovery has received ample attention over many years, with varied solutions being proposed (e.g., Bacon et al., 2017; Šimsek & Barto, 2004; Daniel et al., 2016; Florensa et al., 2017; Konidaris & Barto, 2009; Mankowitz et al., 2016; McGovern & Barto, 2001). Recently, Machado et al. (2017) and Vezhnevets et al. (2017) proposed the idea of learning options that traverse directions of a latent representation of the environment. In this paper we further explore this idea.

More specifically, we focus on the concept of eigenoptions (Machado et al., 2017), options learned using a model of diffusive information flow in the environment. They have been shown to improve agents' performance by reducing the expected number of time steps a uniform random policy needs in order to traverse the state space. Eigenoptions are defined in terms of proto-value functions (PVFs; Mahadevan, 2005), basis functions learned from the environment’s underlying state-transition graph. PVFs and eigenoptions have been defined and thoroughly evaluated in the tabular case. Currently, eigenoptions can be used in environments where it is infeasible to enumerate states only when a linear representation of these states is known beforehand.

In this paper we extend the notion of eigenoptions to stochastic environments with non-enumerated states, which are commonly approximated by feature representations. Despite methods that learn representations generally being more flexible, more scalable, and often leading to better performance, current algorithms for eigenoption discovery cannot be combined with representation learn-
ing. We introduce an algorithm that is capable of discovering eigenoptions while learning representations. The learned representations implicitly approximate the model of diffusive information flow (hereafter abbreviated as the DIF model) in the environment. We do so by exploiting the equivalence between PVFs and the successor representation (SR; Dayan, 1993). Notably, by using the SR we also start to deal with stochastic transitions naturally, a limitation of previous algorithms.

We evaluate our algorithm in a tabular domain as well as on Atari 2600 games. We use the tabular domain to provide intuition about our algorithm and to compare it to the algorithms in the literature. Our evaluation in Atari 2600 games provides promising evidence of the applicability of our algorithm in a setting in which a representation of the agent’s observation is learned from raw pixels.

2 BACKGROUND

In this section we discuss the reinforcement learning setting, the options framework, and the set of options known as eigenoptions. We also discuss the successor representation, which is the main concept used in the proposed algorithm.

2.1 REINFORCEMENT LEARNING AND OPTIONS

We consider the reinforcement learning (RL) problem in which a learning agent interacts with an unknown environment in order to maximize a reward signal. RL is often formalized as a Markov decision process (MDP), described as a 5-tuple \((S, A, P, r, \gamma)\). At time \(t\) the agent is in state \(s_t \in S\) where it takes action \(a_t \in A\) that leads to the next state \(s_{t+1} \in S\) according to the transition probability kernel \(p(s_{t+1} | s_t, a_t)\). The agent also observes a reward \(R_{t+1}\) generated by the function \(r : S \times A \rightarrow \mathbb{R}\). The agent’s goal is to learn a policy \(\pi : S \times A \rightarrow [0, 1]\) that maximizes the expected discounted return \(G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}[s_t]\), where \(\gamma \in [0, 1]\) is the discount factor.

In this paper we are interested in the class of algorithms that determine the agent’s policy by being greedy with respect to estimates of value functions, either w.r.t. the state value \(v_\pi(s)\), or w.r.t. the state-action value function \(Q_\pi(s, a)\). Formally, \(v_\pi(s) = \mathbb{E}_\pi[G_t | s_t = s]\), \(Q_\pi(s, a) = \mathbb{E}_\pi[G_t | s_t = s, a_t = a]\). Notice that in large problems these estimates have to be approximated because it is infeasible to learn a value for each state-action pair. This is generally done by parameterizing \(Q_\pi(s, a)\) with a set of weights \(\theta\) such that \(Q(s, \theta) = q_\pi(s, \theta)\). Currently, neural networks are the most successful parametrization approach in the field (e.g., Mnih et al., 2015; Tesauro, 1995). One of the better known instantiations of this idea is the algorithm called Deep Q-network (DQN, Mnih et al., 2015), which uses a neural network to estimate state-action value functions from raw pixels.

Options (Sutton et al., 1999) are our main topic of study. They are temporally extended actions that allow us to represent courses of actions. An option \(\omega \in \Omega\) in the tabular case is a 3-tuple \((\mathcal{L}, \pi, \mathcal{T})\) where \(\mathcal{L} \subseteq S\) denotes the option’s initiation set, \(\pi : \mathcal{L} \times A \rightarrow [0, 1]\) denotes the option’s policy, and \(\mathcal{T} \subseteq S\) denotes the option’s termination set. Consider the tabular model in which a meta-policy \(\mu : S \rightarrow \Omega\) dictates the agent’s behavior (notice \(A \subseteq \Omega\)). After the agent decides to follow option \(\omega\) from a state in \(\mathcal{L}\), actions are selected according to \(\pi\) until the agent reaches a state in \(\mathcal{T}\). We are interested in learning \(\mathcal{L}, \pi, \mu\), and \(\mathcal{T}\) from scratch.

2.2 PROTO-VALUE FUNCTIONS AND EIGENOPTIONS

Eigenoptions are options that maximize eigenpurposes \(r^\pi\), intrinsic reward functions obtained from the DIF model (Machado et al., 2017). Formally,

\[
r^\pi(s', s) = \mathbb{E}_\pi[\phi(s') - \phi(s)],
\]

where \(\phi()\) denotes a feature representation of a given state (e.g., one-hot encoding in the tabular case) and \(r^\pi\) denotes an eigenvector encoding the DIF model at a specific time-step. Each intrinsic reward function, defined by the eigenvector being used, incentivizes the agent to traverse a different latent dimension of the state space.

In the tabular case, the algorithms capable of learning eigenoptions encode the DIF model through the combinatorial graph Laplacian \(L = D^{-1/2}(D - W)D^{-1/2}\), where \(W\) is the graph’s weight matrix and \(D\) is the diagonal matrix whose entries are the row sums of \(W\). The weight matrix is a square matrix where the \(i\)-th entry represents the connection between states \(i\) and \(j\). Notice that this approach does not naturally deal with stochastic or unidirectional transitions because \(W\) is generally defined as a symmetric adjacency matrix. Importantly, the eigenvectors of \(L\) are also known as proto-value functions (PVFs; Mahadevan, 2005; Mahadevan & Maggioni, 2007).

In settings in which states cannot be enumerated, the DIF model is represented through a matrix of transitions \(T\), where \(r_i\) denotes the linear interpolation known beforehand (if \(T\) is diagonal, \(r_i\) can be different from \(r_i\) if transitions are observed more than once). Machado et al. (2017) justifies this sampling strategy with the fact that, in the tabular case, if every transition is sampled once, the eigenvectors of matrix \(T\) converge to \(PVFs\). Because transitions are added once, regardless of their frequency, this algorithm is not well suited to stochastic environments. In this paper we introduce an algorithm that naturally deals with stochasticity and that does not require \(r_i\) to be known beforehand. Our algorithm learns the environment’s DIF model while learning a representation of the environment from raw pixels.

2.3 THE SUCCESSOR REPRESENTATION

The successor representation (SR; Dayan, 1993) determines state generalization by how similar its successor states are. It is defined to be the expected future occupancy of state \(s'\) given the agent’s policy is \(\pi\) and its starting state is \(s\). It can be seen as defining state similarity in terms of time. See Figure 1 for an example. The Euclidean distance between state \(A\) and state \(C\) is smaller than the Euclidean distance between state \(A\) and state \(B\). However, if one considers the gray tiles to be walls, an agent in state \(A\) can reach state \(B\) much quicker than state \(C\). The SR captures this distinction, ensuring that states that are similar to state \(B\) is than \(A\) is to state \(B\).

Let \(s_{\omega}(\cdot)\) denote the indicator function, the SR, \(\Psi(s, s')\), is formally defined, for \(\gamma < 1\), as

\[
\Psi(s, s') = \mathbb{E}_\pi \left( \gamma \sum_{\omega \in \Omega} \langle I_{s' \in \mathcal{T}(\omega)} \rangle \right).
\]

This expectation can be estimated from samples with temporal-difference error (Sutton, 1988):

\[
\hat{\Psi}(s, s') = \Psi(s, s') + \eta \left( I_{s' \in \mathcal{T}(\omega)} + \gamma \hat{\Psi}(s, s') - \Psi(s, s') \right).
\]

where \(\eta\) is the step-size. In the limit, the SR converges to \(\Psi(s, s')\). This lets us decompose the value function into the product between the SR and the immediate reward (Dayan, 1993):

\[
v_\pi(s) = \sum_{s' \in S} \Psi(s, s') \Phi(s').
\]

The SR is directly related to several other ideas in the field. It can be seen as the dual approach to dynamic programming and to value-function based methods in reinforcement learning (Wang et al., 2007). Moreover, the eigenvectors generated from its eigendecomposition are equivalent to proto-value functions (Stachenfeld et al., 2014; 2017) and to slow feature analysis (Sprekeler, 2011).
### 3 Eigenoption Discovery

In order to discover eigenoptions, we first need to obtain the eigenpurposes through the eigenvectors encoding the DIF model in the environment. This is currently done through PVFs, which the agent obtains by either explicitly building the environment’s adjacency matrix or by enumerating all of the environment’s transitions (c.f. Section 2.2). Such an approach is fairly effective in deterministic settings in which states can be enumerated and uniquely identified, i.e., the tabular case. However, there is no obvious extension of this approach to stochastic settings. It may be hard for the agent to explicitly model the environment dynamics in a weight matrix. The existing algorithm, to enumerate the environment’s transitions, may have a large cost. These issues become worse when states cannot be enumerated, i.e., the function approximation case. The existing algorithm that is applicable to the function approximation setting requires a fixed representation as input, not being able to learn a representation while estimating the DIF model.

In this paper we introduce an algorithm that addresses the aforementioned issues by estimating the DIF model through the SR. Also, we introduce a new neural network that is capable of approximating the SR from raw pixels by learning a latent representation of game screens. The learned SR is then used to discover eigenoptions, replacing the need for knowing the combinatorial Laplacian. In this section we discuss the proposed algorithm in the tabular case, the equivalence between PVFs and the SR, and the algorithm capable of estimating the SR, and eigenoptions, from raw pixels.

#### 3.1 The Tabular Case

The general structure of the algorithms capable of discovering eigenoptions is fairly straightforward, as shown in Alg. 1. The agent learns (or is given) a representation that captures the DIF model (e.g., the combinatorial Laplacian). It then uses the eigenvectors of this representation to define eigenoptions (EXTRACT-EIGENOPTIONS), the intrinsic reward functions described by Equation 1 that will learn how to maximize the option’s policy is the one that maximizes this new reward function, while a state $s$ is defined to be terminal with respect to the eigenvector $e_i$ if $q^\pi(\{s, a\}) \leq 0$ for all $a \in A$. The initiation set of an option $e_i$ is defined to be $S_i \subseteq T$.

In the tabular case, our proposed algorithm is also fairly simple. Instead of assuming the matrix $\hat{\Psi}$ is given in the form of the graph Laplacian, or trying to estimate the graph Laplacian from samples by stacking the row vectors corresponding to the different observed transitions, we estimate the DIF model through the successor representation (c.f. Alg. 2). This idea is supported by the fact that, for our purposes, the eigenvectors of the normalized Laplacian and the eigenvectors of the SR are equivalent. Below we formalize this concept and discuss its implications. We show that the eigenvectors of the normalized Laplacian are equal to the eigenvectors of the SR scaled by $\gamma^{-1} D^{1/2}$.

#### 3.2 Relationship between PVFs and the SR

As aforementioned, PVFs (the eigenvectors of the normalized Laplacian) are equal to the eigenvectors of the successor representation scaled by $\gamma^{-1} D^{1/2}$. To the best of our knowledge, this equivalence was first explicitly discussed by Stachenfeld et al. (2014). We provide below a more formal statement of such an equivalence, for the eigenvalues and the eigenvectors of both approaches. We use the proof to further discuss the extent of this interchangeability.

**Theorem.** Stachenfeld et al. (2014): Let $0 < \gamma < 1$ i.e. $\Phi = (1-\gamma T)^{-1}$ denotes the matrix encoding the SR, and let $L = D^{1/2} [(W - W^T) D^{-1/2}]$ denote the matrix corresponding to the normalized Laplacian, both obtained under a uniform random policy. The $i$-th eigenvalue ($\lambda_{iSR}$) of the SR and the $j$-th eigenvalue ($\lambda_{PVF,i}$) of the normalized Laplacian are related as follows:

$$
\lambda_{PVF,i} = (1 - (1 - \lambda_{iSR}) \gamma^{-1})^{-1}
$$

The $i$-th eigenvalue ($\lambda_{iPVF}$) of the SR and the $j$-th eigenvalue ($\lambda_{PVF,i}$) of the normalized Laplacian, where $i + j = n + 1$, with $n$ being the total number of rows (and columns) of matrix $L$, are related as follows:

$$
e_{PVF,j} = ((1 - \lambda_{iPVF})^{-1}) \lambda_{PVF,j}
$$

**Proof.** Let $\lambda_i, \lambda_j$ denote the $i$-th eigenvalue and eigenvector of the SR, respectively. Using the fact that the SR is known to converge, in the limit, to $(1-\gamma T)^{-1}$ (through the Neumann series), we have:

$$
(1 - (1 - \gamma T)^{-1}) = \lambda_i
$$

$$
(I - \gamma T) \lambda_j = \lambda_j
$$

$$
(1 - (1 - \lambda_i) \gamma) \lambda_j = (1 - (1 - \lambda_i) \gamma) \lambda_j
$$

and

$$
(1 - (1 - \lambda_i) \gamma) \lambda_j = \lambda_j
$$

Finally, we have

$$
D^{-1/2} [(W - W^T) D^{-1/2}] \lambda_i = \lambda_i
$$

and

$$
D^{-1/2} [(W - W^T) D^{-1/2}] \lambda_j = \lambda_j
$$

Importantly, when using PVFs we are first interested in the eigenvectors with the corresponding smallest eigenvalues, as they are the “smoothest” ones. However, when using the SR we are interested in the eigenvectors with the largest eigenvalues. The change of variables in Eq. 3 highlights this fact i.e., $\lambda_i = [1 - (1 - \lambda_i) \gamma^{-1}]^{-1}$. The indices $i$ are sorted in the reverse order of the indices $i$. This distinction can be very important when trying to estimate the relevant eigenvectors. Finding the largest eigenvalues/eigenvectors is statistically more robust to noise in estimation and does not depend on the lowest spectrum of the matrix. Moreover, notice that the scaling by $D^{1/2}$ does not change the direction of the eigenvectors when the size of the action set is constant across all states. This is often the case in the RL problems being studied.

#### 3.3 The Function Approximation Case: The SR through Deep Neural Networks

The tabular case is interesting to study because it provides intuition about the problem and it is easier to analyze, both empirically and theoretically. However, the tabular case is only realizable

---

1 Notice the matrix $\Phi$ is not guaranteed to be symmetric. In that case one can define the eigenpurposes to be $\Phi$’s right eigenvectors, as we do in Section 3.3.
Optimal Action

205

Eigenoption Discovery Through The Deep Successor Representation

ψ

the fact that we have two neural networks. We use

where

extension of the SR to this setting. We use Barreto et al.’s (2017) definition of successor features,

The SR with non-enumerated states: Originally, the SR was not defined in the function approxima-

tion setting, where states are described in terms of feature vectors. Successor features are the natural

The architecture we use is depicted in Fig 2. The reconstruction module uses the learned latent represen-
tation as input \( i.e. \), the output of the representation learning module.

The proposed neural network receives raw pixels as input and learns to estimate the successor fea-
tures of a lower-dimension representation learned by the neural network. The loss function \( L_{SR} \)

we use to learn the successor features is:

\[
L_{SR}(s, a, s') = \mathbb{E} \left( [\phi^T(s') + \gamma \phi^T(s') - \phi(\pi(s))] \right),
\]

where \( \phi(s) \) denotes the feature vector encoding the learned representation of state \( s \) and \( \psi(\cdot) \) denotes the estimated successor features. In practice, \( \phi(\cdot) \) is the output of the representation learning module and \( \psi(\cdot) \) is the output of the SR estimator, as shown in Fig. 2. The loss function above also highlights the fact that we have two neural networks. We use \(-\) to represent a target network (Mnih et al., 2015), which is updated at a slower rate for stability purposes.

We cannot directly estimate the successor features from raw pixels using only \( L_{SR} \) because zero is one of its fixed points. This is the reason we added Oh et al.’s (2015) reconstruction module in the proposed network. It behaves as an auxiliary task (Jaderberg et al., 2017) that predicts the next state to be observed given the current state and action. By predicting the next state we increase the likelihood the agent will learn a representation that takes into consideration the pixels that are under its control, which has been shown to be a good bias in RL problems (Bellemare et al., 2012). Such an auxiliary task is defined through the network’s reconstruction error \( L_{RE} \):

\[
L_{RE}(s, a, s') = \| \phi(s, a) - s' \|^2,
\]

where \( \phi(\cdot) \) denotes the output of the reconstruction module, as shown in Fig. 2. The final loss being optimized is \( L(s, a, s') = L_{RE}(s, a, s') + L_{SR}(s, a, s') \).

Finally, to ensure that the SR will not interfere with the learned features, we zero the gradients coming from the SR estimator (represented with the symbol \( \psi \)) in Fig. 2. We trained our model with RMSProp and we followed the same protocol Oh et al. (2015) used to initialize the network.

4. EXPERIMENTS

We evaluate the discovered eigenoptions quantitatively and qualitatively in this section. We use the traditional rooms domain to evaluate the impact, on the eigenvectors and on the discovered options, of approximating the DIF model through the SR. We then use Atari 2600 games to demonstrate how the proposed network does discover purposeful options from raw pixels.

4.1 Tabular Case

Our first experiment evaluates the impact of estimating the SR from samples instead of assuming the DIF model was given in the form of the normalized Laplacian. We use the rooms domain (Fig. 3a; Sutton et al., 1999) to evaluate our method. Fig. 4b depicts the first eigenvector obtained from the SR while Fig. 4c depicts the corresponding eigenoption. We followed the uniform random policy for 1,000 episodes to learn the SR. Episodes were 100 time steps long. We used a step-size of 0.1, and we set \( \gamma = 0.9 \). The estimated eigenvector is fairly close to the true one and, as expected, the obtained eigenvector is fairly similar to the PVFs that are obtained for this domain. In the Appendix we provide the plots for the true SR and the PVF, as well as plots for different eigenvectors, comparing them to those obtained from \((I - \gamma T)^{-1} \).

Eigenoptions are known for improving the agent’s ability to explore the environment. We use the metric diffusion time to validate whether such an ability is preserved with our method. The diffusion time can be seen as a proxy for how hard it is for an agent to reach the goal state when following a uniform random policy. It is defined as the expected number of decisions (action selection steps) an agent needs to take, when following the uniform random policy, to navigate between two randomly chosen states. We compared the agent’s diffusion time when using eigenoptions obtained with PVFs to the diffusion time when using eigenvectors obtained with estimates of the SR. As we can see in Fig 3d, the eigenoptions obtained with the SR do help the agent to explore the environment.
gap between the diffusion time when using PVFs and when using the SR is likely due to different ways of dealing with corners. The SR implicitly models self-loops in the state-space, while the agent takes an action and it observes it did not move.

We also evaluated how the estimates of the SR evolve as more episodes are used during learning, and its impact in the diffusion time (Fig 3d). In the Appendix we present more results, showing that the local structure of the graph is generally preserved. Naturally, more episodes allow us to learn more accurate estimates of the SR as a more global facet of the environment is seen, since the agent has more chances to further explore the state space. However, it seems that even the SR learned from few episodes allow us to discover useful eigenpurposes, as depicted in Fig 3d. The eigenpurposes obtained from the SR learned using only 100 episodes are already capable of reducing the agent’s diffusion time considerably. Finally, it is important to stress that the discovered options do more than randomly selecting subgoal states. “Random options” only reduce the agent’s diffusion time when hundreds of them are added to the agent’s action set (Machado et al., 2017).

Finally, we evaluated the use of the discovered eigenpurposes to maximize reward. In our experiments the agent learned, off-policy, the greedy policy over primitive actions (target policy) while following the uniform random policy over actions and eigenpurposes (behavior policy). We used Q-learning (Watkins & Dayan, 1992) in our experiments – parameters $\alpha = 0.1$, $\gamma = 0.9$. As before, episodes were 100 time steps long. Figure 4 summarizes the obtained results comparing the performance of our approach to regular Q-learning over primitive actions. The eigenpurposes were extracted from estimates of the SR obtained after 100 episodes. The reported results are the average over 24 independent runs when learning the SR, with each one of these runs encoding 100 runs evaluating Q-Learning. The options were added following the sorting provided by the eigenvalues. For example, if $p$ options denotes an agent with the action set used in the behavior policy being composed of the four primitive actions and the four eigenpurposes generated by the top 2 eigenvalues (both directions are used). Notice that these results do not try to take the sample efficiency of our approach into consideration, they are only meant to showcase how eigenpurposes, once discovered, can speed up learning. The sample complexity of learning options is generally justified in lifelong learning settings where they are re-used over multiple tasks (e.g. Brunskill & Li, 2014).

This is beyond the scope of this paper. The obtained results clearly show that eigenpurposes are not only capable of reducing the diffusion time in the environment but also improving the agent’s control performance. But they do so by increasing the likelihood that the agent will cover a larger part of the state-space, given the same amount of time. Moreover, as before, it seems that a very accurate estimate of the successor representation is not necessary for the eigenpurposes to be useful. Similar results can be obtained for important subgoals of the start and goal states, and when the estimates of the SR are more accurate. These results can be seen in the Appendix.

4.2 Atari 2600

This second set of experiments evaluates the eigenpurposes discovered when the SR is obtained from raw pixels. We obtained the SR through the neural network described in Section 3. We used four Atari 2600 games from the Arcade Learning Environment (Bellemare et al., 2013) as testbed: BANK HEIST, FREEWAY, MONTZUMA’S REVENGE, and MS. PAC-MAN.

We followed the protocol described in the previous section to create eigenpurposes. We trained the network in Fig. 2 to estimate the SR under the uniform random policy. Since the network does not impact the policy being followed, we built a dataset of 500,000 samples for each game and used this dataset to optimize the network weights. We passed through the shuffled dataset 10 times, using RMSProp with a step size of $10^{-3}$. Once we were done with the training, we let the agent follow a uniform random policy for 50,000 steps while we stored the SR output by the network for each observed state as a row of matrix $T$. We define $\phi$, in the eigenpurposes we maximize (c.f. Eq. 1), to be the right eigenvectors of the matrix $T$, while $\phi'$ is extracted at each time step from the network in Fig. 2. Due to computational constraints, we approximated the final eigenpurposes. We did so by using the ALE’s internal emulator to do a one-step lookahead and act greedily with respect to each eigenpurposes (in practice, this is equivalent to learning with $\gamma = 0$). This is not ideal because the options we obtain are quite limited, since they do not deal with delayed rewards. However, even in such limiting setting we were able to obtain promising results. We discuss below.

Following Machado et al. (2017), we evaluate the discovered eigenpurposes qualitatively. We execute all options following the procedure described above (greedy one-step lookahead) while tracking the agent’s position on the screen. Figure 5 summarizes the behavior of some of the meaningful options discovered. The trajectories generated by different options are represented by different colors and the color’s intensity at a given location represents how often the agent was at that location. Eigenpurposes were introduced as options that generate purposeful behavioral patterns that helps agents explore the environment. We can clearly see that the discovered eigenpurposes are indeed purposeful. They aim to reach a specific location and stay there. If this was not the case, the agent’s trajectory would be much more visible. Instead, what we actually observe is that the mass of visitation is concentrated on one location on the screen, dominating (color intensity) all the others. The location the agent is spending most of its time on in fact can be seen as the option’s terminal state. Constantly being in a state suggests the agent has arrived to a myopic local maximum for that eigenpurposes. In three out of four games (BANK HEIST, MONTZUMA’S REVENGE, MS. PAC-MAN) our algorithm discovers options that clearly push the agent to corners and to other relevant parts of the state space, corroborating the intuition that eigenpurposes also improve exploration. In MONTZUMA’S REVENGE, the terminal state of the highlighted options even correspond to what are considered good subgoals for the game (Kulkarni et al., 2016a). It is likely that additional subgoals, such as the key, were found not due to our myopic greedy approach. This approach may also explain why our algorithm was ineffective in FREEWAY. Avoiding cars may be impossible without longer-term planning. A plot depicting the two meaningful options discovered in this game is in the Appendix. Importantly, the fact that myopic policies are able to navigate to specific locations and stay there also suggests that, as in the tabular case, the proposed approach gives rise to dense intrinsic rewards that are very informative. This is another important constraint between randomly assigned subgoals and our approach. Randomly assigned subgoals do not give rise to such dense rewards. Thus, one can argue that our approach does not only generate useful options but it also gives rise to dense eigenpurposes, making it easier to build the policies associated with them.
It is important to stress that our algorithm was able to discover eigenoptions, from raw pixels, similar to those obtained by algorithms that use the RAM state of the game as a feature representation. The RAM state of the game often uses specific bytes to encode important information of the game, such as the position of the player’s avatar in the game. Our algorithm had to implicitly learn what were the meaningful parts of the screen. Also, different from previous algorithms, our approach is not constrained by the dimensionality of the state representation nor to binary features. Based on this discussion, we consider our results to be very promising, even though we only depict options that have effect on the initial state of the games. We believe that in a more general setting (e.g., using DQN to learn policies) our algorithm has the potential to discover even better options.

5 Related Work

Our work was directly inspired by Kulkarni et al. (2016b), the first to propose approximating the SR using a neural network. We use their loss function in a novel architecture. Because we are not directly using the SR for control, we define the SR in terms of states, instead of state-action pairs. Different from Kulkarni et al. (2016b), our network does not learn a reward model and it does not use an autorencoder to learn a representation of the world. It tries to predict the next state the agent will observe. The prediction module we used was introduced by Oh et al. (2015). Because it predicts the next state, it implicitly learns representations that take into consideration the parts of the screen that are under the agent’s control. The ability to recognize such features is known as contingency awareness, and it is known to have the potential to improve agents’ performance (Bellemare et al., 2012). Kulkarni et al. (2016b) did suggest the deep SR could be used to find bottleneck states, which are commonly used as subgoals for options, but such an idea was not further explored. Importantly, Jong et al. (2008) and Machado et al. (2017) have shown that options that look for bottleneck states can be quite harmful in the learning process.

The idea of explicitly building hierarchies based on the learned latent representation of the state space is due to Machado et al. (2017) and Vezhnevets et al. (2017). Machado et al. (2017) proposed the concept of eigenoptions, but limited to the linear function approximation case. Vezhnevets et al. (2017) do not explicitly build options with initiation and termination sets. Instead, they learn a hierarchy through an end-to-end learning system that does not allow us to easily retrieve options from it. Finally, Kompella et al. (2017) has proposed the use of slow feature analysis (SFA), Wiskott & Sejnowski, 2002) to discover options. Sprechler (2011) has shown that, given a specific choice of adjacency function, PVFs (and consequently the SR) are equivalent to SFA. However, their work is limited to linear function approximation. Our method also differs in how we define the initiation and termination sets. The options they discover look for bottleneck states, which is not our case.

6 Conclusion

In this paper we introduced a new algorithm for eigenoption discovery in RL. Our algorithm uses the successor representation (SR) to estimate the model of diffusive information flow in the environment, leveraging the equivalence between proto-value functions (PVFs) and the SR. This approach circumvents several limitations from previous work: (i) it builds increasingly accurate estimates using a constant-cost update-rule; (ii) it naturally deals with stochastic MDPs; (iii) it does not depend on the assumption that the transition matrix is symmetric; and (iv) it does not depend on handcrafted feature representations. The first three items were achieved by simply using the SR instead of the PVFs, while the latter was achieved by using a neural network to estimate the SR.

The proposed framework opens up multiple possibilities for investigation in the future. It would be interesting to evaluate the compositionality of eigenoptions, or how transferable they are between similar environments, such as the different modes of Atari 2600 games (Machado et al., 2018). Finally, now that the fundamental algorithms have been introduced, it would be interesting to investigate whether one can use eigenoptions to accumulate rewards instead of using them for exploration.

Acknowledgments

The authors would like to thank Craig Sherstan and Martha White for feedback on an earlier draft, Kamyar Azizzadenesheli, Marc G. Bellemare and Michael Bowling for useful discussions, and the anonymous reviewers for their feedback and suggestions.

References


APPENDIX: SUPPLEMENTARY MATERIAL

This supplementary material contains details omitted from the main text due to space constraints. The list of contents is below:

- A more detailed proof of the theorem in the paper;
- Empirical results evaluating how the number of episodes used to learn the successor representation impacts the obtained eigenvectors and their corresponding eigenoptions;
- Evaluation of the reconstruction module (auxiliary task) that learns the latent representation that is used to estimate the successor representation.

A MORE DETAILED PROOF OF THE THEOREM IN THE MAIN PAPER

Theorem. Stachenfeld et al. (2014): Let $0 < q < 1$ and $\mathbb{F} = (1 - \gamma T)^{-1}$ denote the matrix encoding the SR, and let $L = D^{-1/2}(D - W)D^{-1/2}$ denote the matrix corresponding to the normalized Laplacian, both obtained under a uniform random policy. The $i$-th eigenvalue ($\lambda_{SR,i}$) of the SR and the $j$-th eigenvalue ($\lambda_{PVF,j}$) of the normalized Laplacian are related as follows:

$$\lambda_{SR,i} = (1 - 1/\lambda_{PVF,i})^{-1}$$

The $i$-th eigenvector ($e_{SR,i}$) of the SR and the $j$-th eigenvector ($e_{PVF,j}$) of the normalized Laplacian, such that $i + j = n + 1$, with $n$ being the total number of rows (and columns) of matrix $T$, are related as follows:

$$e_{SR,i} = (\gamma^{-1} D^{1/2}) e_{PVF,j}$$

Proof. This proof is more detailed than the one presented in the main paper. Let $\lambda_i, e_i$ denote the $i$-th eigenvalue and eigenvector of the SR. Using the fact that the SR is known to converge, in the limit, to $(1 - \gamma T)^{-1}$ (through the Neumann series), we have:

$$L = D^{-1/2}(D - W)D^{-1/2}$$

where

$$L = \sum_{k=1}^{n} (I - \gamma T)^{k-1} W (I - \gamma T)^{-1}$$

This expression is obtained by expanding the Neumann series:

$$L = \lim_{k \to \infty} \sum_{k=0}^{n} (I - \gamma T)^{k-1} W (I - \gamma T)^{-1} = \sum_{k=1}^{n} (I - \gamma T)^{k-1} W (I - \gamma T)^{-1}$$

Then, the $i$-th eigenvalue of the SR is given by

$$\lambda_{SR,i} = \frac{\sum_{j=1}^{n} (I - \gamma T)^{j-1} W (I - \gamma T)^{-1} e_i}{\sum_{j=1}^{n} (I - \gamma T)^{j-1} W (I - \gamma T)^{-1}}$$

and the $j$-th eigenvalue of the normalized Laplacian is given by

$$\lambda_{PVF,j} = \frac{\sum_{i=1}^{n} (I - \gamma T)^{i-1} W (I - \gamma T)^{-1} e_j}{\sum_{i=1}^{n} (I - \gamma T)^{i-1} W (I - \gamma T)^{-1}}$$

Finally, the eigenvectors are related as

$$e_{SR,i} = (\gamma^{-1} D^{1/2}) e_{PVF,j}$$

This completes the proof.
THE IMPACT THE NUMBER OF EPISODES HAS IN LEARNING THE SR AND THE EIGENOPTIONS

In Section 4.1 we briefly discussed the impact of estimating the successor representation from samples instead of assuming the agent has access to the normalized Laplacian. It makes much more sense to use the successor representation as the DIF model in the environment if we can estimate it quickly. The diffusion time was the main evidence we used in Section 4.1 to support our claim that early estimates of the successor representation are useful for eigenoption discovery. In order to be concise we did not actually plot the eigenvectors of the estimates of the successor representation at different moments, nor explicitly compared them to proto-value functions or to the eigenvectors of the matrix $(I - \gamma T)^{-1}$. We do so in this section.

Figures 7–10 depict the first four eigenvectors of the successor representation in the Rooms domain, after being learned for different number of episodes (episodes were 100 time steps long, $\gamma = 0.1$, $\gamma = 0.9$). We also depict the corresponding eigenvectors of the $(I - \gamma T)^{-1}$ matrix1, and of the normalized Laplacian (Machado et al., 2017). Because the eigenvectors orientation (sign) is often arbitrary in an eigendecomposition, we matched their orientation to ease visualization.

Overall, after 500 episodes we already have an almost perfect estimate of the first eigenvectors in the environment, while 100 episodes seem to not be enough to accurately learn the DIF model in all rooms. However, learning the successor representation for 100 episodes seems to be enough to generate eigenoptions that reduce the agent’s diffusion time, as shown in Figure 3d. We can better discuss this behavior by looking at Figures 11–14, which depict the options generated by the obtained eigenvectors.

With the exception of the options generated after learning the successor representation for 100 episodes, all the eigenoptions obtained from estimates of the successor representation already move the agent towards the “correct” room(s). Naturally, they do not always hit the corners, but the general structure of the policies can be clearly seen. We also observe that the eigenoptions obtained from proto-value functions are shifted one tile from the corners. As discussed in the main paper, this is a consequence of how Machado et al.’s (2017) dealt with corners. They did not model self-loops in the MDP, despite the fact that the agent can be in the same state for two consecutive steps. The successor representation captures this naturally. Finally, we use Figure 11a to speculate why the options learned after 100 episodes are capable of reducing the agent’s diffusion time. The first eigenoption learned by the agent moves it to the parts of the state space it has never been to, this may be the reason that the combination of these options is so effective. It also suggests that incremental methods for option discovery and exploration are a promising path for future work.

USING EIGENOPTIONS TO ACCUMULATE REWARD IN THE ENVIRONMENT

In Section 4.1 we also evaluated the agent’s ability to accumulate reward after the eigenoptions have been learned. We further analyze this topic here. As in Section 4.1, the agent learned, off-policy, the greedy policy over primitive actions (target policy) while following the uniform random policy over actions and eigenoptions (behavior policy). We used Q-learning (Watkins & Dayan, 1992) in our experiments—parameters $\lambda = 0$, $\alpha = 0.1$, $\gamma = 0.9$. Episodes were 100 time steps long. Figures 16–19 summarize the obtained results comparing the performance of our approach to regular Q-learning over primitive actions in four different environments (c.f. Figure 15). We evaluate the agent’s performance when using eigenoptions extracted from estimates of the SR obtained after 100, 500, and 1000 episodes, as well eigenoptions obtained from the true SR, i.e., $(I - \gamma T)^{-1}$. The reported results are the average over 24 independent runs when learning the SR, with each one of these runs encoding 100 runs evaluating Q-Learning. The options were added following the sorting provided by the eigenvalues. For example, 4 options denotes an agent with the action set used in the behavior policy being composed of the four primitive actions and the four eigenoptions generated by the top 4 eigenvalues (both directions are being used).

We can see that eigenoptions are not only capable of reducing the diffusion time in the environment but also improving the agent’s control performance. They do so by increasing the likelihood that the agent will cover a larger part of the state space given the same amount of time. Interestingly, few eigenoptions seem to be enough for the agent. Moreover, although rough estimates of the SR seem to be enough to improve the agent’s performance (e.g., estimates obtained after only 100 episodes),

Recall $(I - \gamma T)^{-1}$ is the matrix to which the successor representation converges to in the limit.
Figure 7: Evolution of the **first eigenvector** being estimated by the SR and baselines.

Figure 8: Evolution of the **second eigenvector** being estimated by the SR and baselines.

Figure 9: Evolution of the **third eigenvector** being estimated by the SR and baselines.

Figure 10: Evolution of the **fourth eigenvector** being estimated by the SR and baselines.

Figure 11: Evolution of the **first eigenoption** being estimated by the SR and baselines.

Figure 12: Evolution of the **second eigenoption** being estimated by the SR and baselines.

Figure 13: Evolution of the **third eigenoption** being estimated by the SR and baselines.

Figure 14: Evolution of the **fourth eigenoption** being estimated by the SR and baselines.
Figure 15: Different environments (varying start and goal locations) used when evaluating the agent’s ability to accumulate reward with and without eigenoptions.

Figure 16: Plot depicting the agent’s performance when following options obtained through estimates of the SR (100, 500, and 1,000 episodes), as well as through the true SR, in environment 1.

Figure 17: Plot depicting the agent’s performance when following options obtained through estimates of the SR (100, 500, and 1,000 episodes), as well as through the true SR, in environment 2.

Figure 18: Plot depicting the agent’s performance when following options obtained through estimates of the SR (100, 500, and 1,000 episodes), as well as through the true SR, in environment 3.

Figure 19: Plot depicting the agent’s performance when following options obtained through estimates of the SR (100, 500, and 1,000 episodes), as well as through the true SR, in environment 4.

Figure 20: Final 1-step predictions in the game BANK HEIST. We use the task of predicting the next game screen as an auxiliary task when estimating the successor representation.
Figure 21: Final 1-step predictions in the game FREEWAY. We use the task of predicting the next game screen as an auxiliary task when estimating the successor representation.

Figure 22: Final 1-step predictions in the game MONTEZUMA’S REVENGE. We use the task of predicting the next game screen as an auxiliary task when estimating the successor representation.
Figure 23: Final 1-step predictions in the game Ms. PACMAN. We use the task of predicting the next game screen as an auxiliary task when estimating the successor representation.
Abstract

Building systems that autonomously create temporal abstractions from data is a key challenge in scaling learning and planning in reinforcement learning. One popular approach for addressing this challenge is the options framework [29]. However, only recently in [1] was a policy gradient theorem derived for online learning of general purpose options in an end to end fashion. In this work, we extend previous work on this topic that only focuses on learning a two-level hierarchy including options and primitive actions to enable learning simultaneously at multiple resolutions in time. We achieve this by considering an arbitrarily deep hierarchy of options where high level temporally extended options are composed of lower level options with finer resolutions in time. We extend results from [1] and derive policy gradient theorems for a deep hierarchy of options. Our proposed hierarchical option-critic architecture is capable of learning internal policies, termination conditions, and hierarchical compositions over options without the need for any intrinsic rewards or subgoals. Our empirical results in both discrete and continuous environments demonstrate the efficiency of our framework.

1 Introduction

In reinforcement learning (RL), options [29, 21] provide a general framework for defining temporally abstract courses of action for learning and planning. Discovering these temporal abstractions autonomously has been the subject of extensive research over the past two decades [16, 28, 17, 27, 26] with approaches that can be used in continuous state and/or action spaces only recently becoming feasible [9, 20, 15, 14, 10, 31, 3]. Most existing work has focused on finding subgoals (i.e. useful states for the agent to reach) and then learning policies to achieve them. However, these approaches do not scale well because of their combinational nature. Recent work on option-critic learning blurs the line between option discovery and option learning by providing policy gradient theorems for optimizing a two-level hierarchy of options and primitive actions [1]. These approaches have achieved success when applied to Q-learning on Atari games, but also in continuous action spaces [7] and with asynchronous parallelization [6]. In this paper, we extend option-critic to a novel hierarchical option-critic framework, by presenting generalized policy gradient theorems that can be applied to an arbitrarily deep hierarchy of options.

Figure 1: State trajectories over a three-level hierarchy of options. Open circles represent SMDP decision points while filled circles are primitive steps within an option. The low level options are temporally extended over primitive actions, and high level options are even further extended.
Work on learning with temporal abstraction is motivated by two key potential benefits over learning with compact action or skill space: long term credit assignment and exploration. Learning about abstractions or action level or even with low levels of abstraction slows down learning, because agents must learn longer sequences of actions to achieve the desired behavior. This frustrates the process of learning in environments with sparse rewards. By contrast, agents that learn a high level decomposition of sub-tasks are able to explore the environment more effectively by exploring in the abstract action space rather than the primitive action space. While the recently proposed deliberation cost \cite{6} can be used as a margin that effectively controls how temporally extended learned options are, the standard two-level version of the option-critic framework is still ill-equipped to learn complex tasks that require sub-task decomposition at multiple quite different temporal resolutions of abstraction. In Figure 1 we depict how we overcome this obstacle to learn a deep hierarchy of options. The standard two-level option hierarchy constructs a Semi-Markov Decision Process (SMDP), where new options are chosen when temporally extended sequences of primitive actions are terminated. In our framework, we consider not just options and primitive actions, but an arbitrary depth hierarchy of lower level and higher level options as well. Higher level options represent a further temporally extended SMDP than the low level options below, as they only have an opportunity to terminate during points where all lower level options also terminate.

We will start by reviewing related research and by describing the seminar we build upon in this paper that first derived policy gradient theorems for learning with primitive actions \cite{30} and options \cite{1}. We will then describe the core ideas of our approach, presenting hierarchical intra-option policy and termination gradient theorems. We leverage this new type of policy gradient learning to construct a hierarchical option-critic architecture, which generalizes the option-critic architecture to an arbitrarily deep hierarchy of options. Finally, we demonstrate the empirical benefits of this architecture over standard option-critic when applied to RL benchmarks. To the best of our knowledge, this is the first general purpose end-to-end approach for learning a deep hierarchy of options beyond two-levels in RL settings, scaling to very large domains at comparable efficiency.

2 Related Work

Our work is related to recent literature on learning to compose skills in RL. As an example, Sahni et al. \cite{23} leverages a logic for combining pre-learned skills by learning an embedding to represent the combination of a skill and state. Unfortunately, their system relies on a pre-specified sub-task decomposition into skills. In \cite{25}, the authors propose to ground all goals in a natural language description space. Created descriptions can then be high level and express a sequence of goals. While these are interesting directions for further exploration, we will focus on a more general setting without provided natural language goal descriptions or sub-task decomposition information.

Our work is also related to methods that learn to decompose the problem over long time horizons. A prominent paradigm for this is Fraud Reinforcement Learning \cite{4}, which learns by managing both task and worker models. Theoretically, this can be extended to a deeper hierarchy of managers and their managers as done in the original work for a hand-designed decomposition of the state space. More recently, Veitch et al. \cite{32} showed the ability to successfully train a neural model end to end with deep neural networks for the Atari games. However, this has only been achieved for a very low hierarchy (i.e. one manager and one worker). We can think of Fraud Reinforcement learning to decompose the problem with respect to the state space, while the options framework learns a temporal decomposition of the problem. Recent work \cite{11} also breaks down the problem over a temporal hierarchy, but like \cite{32} is based on learning a latent goal representation that modulates the policy behavior as opposed to options. Conceptually, options stress choosing among skill abstractions and featural stresses the achievement of certain kinds of states. Humans tend to use both of these kinds of reasoning when appropriate and we conjecture that a hybrid approach will likely win out in the end. Unfortunately, in the space available we feel that we cannot come to definitive conclusions on the precise nature of the differences and potential synergies of these approaches.

The concept of learning a hierarchy of options is not new. It is an obviously desirable extension of options envisioned in the original papers. However, actually learning a deep hierarchy of options ended to have resulted surprisingly little attention to date. Compositional planning where options select other options was first considered in \cite{26}. The authors provided a generalization of value iteration to option models for multiple subgoals, leveraging explicit subgoals for options. Recently, Fox et al. \cite{5} successfully trained a hierarchy of options end to end for imitation learning. Their approach leverages an EM based algorithm for recursive discovery of additional levels of the option hierarchy. Unfortunately, their approach is only applicable to the imitation learning setting and not for general use. In this work, we are first to propose theorems along with a practical algorithm and architecture to train arbitrarily deep hierarchies of options end to end using policy gradients, maximizing the expected return.

3 Problem Setting and Notation

A Markov Decision Process (MDP) consists of a set of states \( \mathcal{S} \), a set of actions \( \mathcal{A} \), a transition function \( \mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S} \) and a reward function \( r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R} \). We follow \cite{1} and develop our ideas assuming discrete state and action sets, while our results extend to continuous spaces using usual measure-theoretic assumptions as we demonstrate in our experiments. A policy is a probability distribution over actions conditioned on states, \( \pi : \mathcal{S} \rightarrow \mathcal{A} \). The value function of a policy \( \pi \) is defined as the expected return \( V_{\pi}(s) = \mathbb{E}_{\mathcal{T}} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right] \) with an action-value function of \( Q_{\pi}(s,a) = \mathbb{E}_{\mathcal{T}} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right] \) where \( \gamma \in [0,1) \) is the discount factor.

Policy gradient methods \cite{30,8} address the problem of finding a good policy by performing stochastic gradient descent to optimize a performance objective over a given family of parameterized stochastic policies, \( \pi \). The policy gradient theorem \cite{30} provides an expression for the gradient of the discounted reward objective with respect to \( \pi \). The objective is defined with respect to a designated start state (or distribution) \( \rho : \mathcal{S} \rightarrow \mathbb{R} \) and defines the update rule for the termination policy for the initial condition \( o \in \mathcal{O} \).

The options framework \cite{29,21} formalizes the idea of temporally extended actions. A Markovian option \( o \in \mathcal{O} \) is a tuple \( (I_o, \pi_o, \delta_o) \) in which \( I_o \subseteq \mathcal{S} \) is an initiation set, \( \pi_o \) is an intra-option policy, and \( \delta_o : \mathcal{S} \rightarrow \mathcal{O} \) is a termination function. Like most option discovery algorithms, we assume that all options are available everywhere. MDPs with options become SMDPs \cite{22} with a corresponding optimal value function over options \( V_o(s) \) and option-value function \( Q_o(s,a) \) \cite{29,21}.

The option-critic architecture \cite{1} leverages a call-and-return option execution model, in which an agent picks option \( o \) according to its option policy \( \pi_o \), then follows the intra-option policy \( \pi_o \) until termination (as dictated by \( \delta_o \)). For each option, we can define the intra-option policy of option \( o \) by \( \Theta \) and a termination function of \( o \) parameterized by \( \phi \). Like policy gradient methods, the option-critic architecture optimizes directly the discounted return expected over all the trajectories starting at a designated state \( s_0 \) and option \( o \) according to \( Q_{\Theta,\phi}(s,a) \). The option-value function is defined as

\[
Q_{\Theta,\phi}(s,a) = \sum_{o \in \mathcal{O}} \pi_o(s,a) Q_o(s,a,o),
\]

where \( Q_o(s,a) = r(s,a) + \gamma \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | \delta_o(s,a)] \) is the value of executing an action in the context of a state-option pair: \( Q_o(s,a) = r(s,a) + \gamma \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | \delta_o(s,a)] \). The (r,s) pair leads to an augmented state space \[23\]. The option-critic architecture instead leverages the function \( U : \mathcal{S} \times \mathcal{O} \rightarrow \mathbb{R} \) which is called the option-value function upon arrival \( Q_{\Theta,\phi} \). The value of executing \( o \) upon entering state \( s \) is given by

\[
U(s,o) = \left\{ \begin{array}{ll}
Q_o(s,a) & \text{if } o \in \mathcal{O} \\
\rho(s) & \text{otherwise}
\end{array} \right..
\]

\( \rho \) and \( U \) both depend on \( \Theta \) and \( \phi \) but are omitted from the notation for clarity. The intra-option policy gradient theorem results from taking the derivative of the expected discounted return with respect to the intra-option policy parameters \( \Theta \) and defines the update rule for the intra-option policy:

\[
\nabla_{\Theta} Q_{\Theta,\phi}(s,a) = \sum_{o \in \mathcal{O}} \pi_o(s,a,o) \nabla_{\Theta} \left[ Q_o(s,a,o) \right] - \nabla_{\Theta} Q_o(s,a,o)
\]

where \( \pi_o(s,a,o) \) is a discounted weighting of intra-option state pairs along trajectories starting from \( (s_0,o) \), \( \pi_o(s,a,o) \) is a discounted weighting of intra-option state pairs along trajectories starting from \( (s_0,o) \), and \( \pi_o(s,a,o) \) is a discounted weighting of intra-option state pairs along trajectories starting from \( (s_0,o) \). The termination gradient theorem results from taking the derivative of the expected discounted return with respect to the termination policy parameters \( \phi \) and defines the update rule for the termination policy for the initial condition \( (s_0,o) \):

\[

\nabla_{\phi} Q_{\Theta,\phi}(s,a) = \sum_{o \in \mathcal{O}} \pi_o(s,a,o) \nabla_{\phi} \left[ U(s,o) \right] - \nabla_{\phi} U(s,o)
\]

where \( \nabla_{\phi} U(s,o) = \sum_{o \in \mathcal{O}} \pi_o(s,a,o) \nabla_{\phi} \left[ Q_o(s,a) \right] - \nabla_{\phi} Q_o(s,a) \).
4 Learning Options with Arbitrary Levels of Abstraction

Notation: As it makes our equations much clearer and more condensed we adopt the notation \( x^{i_1 : i_k} = x^{i_1}, \ldots, x^{i_k} \). This implies that \( x^{i} \) denotes a list of variables in the range of \( i \) through \( i + j \).

The hierarchical options framework: that we introduce in this work considers an agent that learns using an \( N \) level hierarchy of policies, termination functions, and value functions. Our goal is to extend the ideas of the option-critic architecture in such a way that our framework simplifies to policy gradient based learning when \( N = 1 \) and option-critic learning when \( N = 2 \). At each hierarchical level above the lowest primitive action level policy, we consider an available set of options \( \Omega^{N-1} \) that is a subset of the total set of available options \( \Omega \). This way we keep our view of the possible available options at each level very broad. On one extreme, each hierarchical level may get its own unique set of options and on the other extreme each hierarchical level may share the same set of options. We present a diagram of our proposed architecture in Figure 2.

![Figure 2: A diagram describing our proposed hierarchical option-critic architecture. Dotted lines represent processes within the agent while solid lines represent processes within the environment. Option selection is top down through the hierarchy and option termination is bottom up (represented with red dotted lines).](image)

We denote \( \pi^N(o^N) \) as the policy over the most abstract options in the hierarchy \( o^N \in \Omega^N \) given the state \( s \). For example, \( \pi^N = \pi_0 \) from our discussion of the option-critic architecture. Once \( o^N \) is chosen with policy \( \pi^N \), then we go to policy \( \pi_0^N(o_0^N, s^1) \), which is the next highest level policy, to select \( o^1 \in \Omega^1 \) conditioning it on both the current state \( s \) and the selected highest level option \( o^N \). This process continues on in the same fashion stepping down to policies at lower levels of abstraction conditioned on the augmented state space considering all selected higher level options until we reach policy \( \pi^1_o(o^1, s^1) \). \( \pi^1 \) is the lowest level policy and it finally selects over the primitive action space conditioned on all the selected options.

Each level of the option hierarchy has a complementary termination function \( z^1, \ldots, z^{N-1} \), \( \beta^N \), and \( \beta^1, \ldots, \beta^{N-1} \) that governs the termination pattern of the selected option on that level. We adopt a bottom up termination strategy where high level options only have an opportunity to terminate when all of the lower level options have terminated first. For example, \( o^{N-2} \) cannot terminate until \( o^{N-1} \) terminates at which point we can assess \( \beta^{N-1}(o^{N-1}, s^{N-2}) \) to see whether \( o^{N-2} \) terminates. If it did terminate, this would allow \( o^{N-1} \), the opportunity to assess if it should terminate and so on. This condition ensures that higher level policy options will be more temporally extended than their lower level option building blocks, which is a key motivation of this work.

The final key component of our system is the value function over the augmented state space. To enable comprehensive reasoning about the policies at each level of the option hierarchy, we need to maintain value functions that consider the state and every possible combination of active options and actions \( \mathcal{V}(s, o^1, \ldots, o^N) \). These value functions collectively serve as the critic in our analogy to the actor-critic and option-critic training paradigms.

4.1 Generalizing the Option Value Function to \( N \) Hierarchical Levels

Like policy gradient methods and the option-critic architecture, the hierarchical options framework optimizes directly for the discounted return expected over all the trajectories starting at a designated state \( s_0 \) and active options \( o_0^{N-1} \):

\[
p(\mathcal{L}^{N-1}, s_0^{N-1}, o_0^{N-1}) = \mathcal{E}(\mathcal{L}^{N-1}) = \mathcal{E}(z^{N-1}, s^{N-1}) + \sum_{i=1}^{N-1} \mathcal{E}(z^i, s^i) q_{\pi}(s^i, o^i)
\]

This return depends on the policies and termination functions at each level of abstraction. We now consider the option value function for understanding reasoning about an option \( o^i \) at level \( 1 \leq i \leq N \) based on the augmented state space \( s^{i+1} \). This way we keep our view of the possible available options at each level very broad. On one extreme, each hierarchical level may get its own unique set of options and on the other extreme each hierarchical level may share the same set of options. We present a diagram of our proposed architecture in Figure 2.

We denote \( \pi^N(o^N) \) as the policy over the most abstract options in the hierarchy \( o^N \in \Omega^N \) given the state \( s \). For example, \( \pi^N = \pi_0 \) from our discussion of the option-critic architecture. Once \( o^N \) is chosen with policy \( \pi^N \), then we go to policy \( \pi_0^N(o_0^N, s^1) \), which is the next highest level policy, to select \( o^1 \in \Omega^1 \) conditioning it on both the current state \( s \) and the selected highest level option \( o^N \). This process continues on in the same fashion stepping down to policies at lower levels of abstraction conditioned on the augmented state space considering all selected higher level options until we reach policy \( \pi^1_o(o^1, s^1) \). \( \pi^1 \) is the lowest level policy and it finally selects over the primitive action space conditioned on all the selected options.

Each level of the option hierarchy has a complementary termination function \( z^1, \ldots, z^{N-1} \), \( \beta^N \), and \( \beta^1, \ldots, \beta^{N-1} \) that governs the termination pattern of the selected option on that level. We adopt a bottom up termination strategy where high level options only have an opportunity to terminate when all of the lower level options have terminated first. For example, \( o^{N-2} \) cannot terminate until \( o^{N-1} \) terminates at which point we can assess \( \beta^{N-1}(o^{N-1}, s^{N-2}) \) to see whether \( o^{N-2} \) terminates. If it did terminate, this would allow \( o^{N-1} \), the opportunity to assess if it should terminate and so on. This condition ensures that higher level policy options will be more temporally extended than their lower level option building blocks, which is a key motivation of this work.

The final key component of our system is the value function over the augmented state space. To enable comprehensive reasoning about the policies at each level of the option hierarchy, we need to maintain value functions that consider the state and every possible combination of active options and actions \( \mathcal{V}(s, o^1, \ldots, o^N) \). These value functions collectively serve as the critic in our analogy to the actor-critic and option-critic training paradigms.

4.2 Generalizing the Intra-option Policy Gradient Theorem

On a high level we can think of actor-critic architectures, generalizing to the option-critic architecture as well, as pairing a critic with each actor network so that the critic has additional information about the value of the actor’s actions that can be used to improve the actor’s learning. However, this is derived by taking gradients with respect to the parameters of the policy, while optimizing for the expected discounted return. The discounted return is approximated by a critic (i.e. value function) with the same augmented state-space as the policy being optimized for. As examples, in actor-critic learning \( \pi(s) \) is optimized by taking the derivative of its parameters with respect to \( V_{\theta}(s) \) and in option-critic learning \( \pi(o, s) \) is optimized by taking the derivative of its parameters with respect to \( Q_{o\theta}(s, o) \). The intra-option policy gradient theorem is an important contribution as it outlines how to optimize for a policy that is also associated with a termination function. As the policy over options in that work never terminates, it does not need a special training methodology and the option-critic architecture allows the practitioner to pick their own method of learning the policy over options while using \( Q \) Learning as an example in their experiments. We do the same in our paper for

\footnote{Note that when no options terminate, as in the first term in equation 10, the lowest level option does not terminate and thus no higher level options have the opportunity to terminate.}
the policy over the highest level options $\pi^3$ that also never terminates. For all other policies $\pi^{N-3}$ we perform a generalization of actor-critic learning by providing an associated critic at each level and guiding gradients using the appropriate critic.

We now seek to generalize the intra-option policy gradients theorem by deriving the correct gradient to follow for a policy at an arbitrary level of abstraction $\pi^l$ by taking the gradient with respect to its parameters $\theta^l$ of the value function with the same augmented state space $Q_{\theta^l}(s,a^{l-1})$. Substituting from equation 8 we find that

$$\nabla_{\theta^l} Q_{\theta^l}(s,a^{l-1}) = \sum_{a^l} \sum_{s'} \pi^l(s'|s,a^{l-1}) Q_{\theta^l}(s',a^l).$$

(11)

**Theorem 1** (Hierarchical Intra-option Policy Gradient Theorem). Given an $N$ level hierarchical set of Markov options with stochastic intra-option policies differentiable in their parameters $\theta^l$ governing each policy $\pi^l$, the gradient of the expected discounted return with respect to $\theta^l$ and initial conditions $(s_0,a^{0-1})$ is:

$$\nabla_{\theta^l} Q_{\theta^l}(s,a^{l-1}) = \sum_{a^l} \sum_{s'} \pi^l(s'|s,a^{l-1}) \sum_{i,l'} p_i'(s'|a^l) \nabla_{\theta^l} U_{\theta^l}(s,a^l).$$

(12)

Hence, the key quantity is the gradient of $U$ with respect to $\theta^l$ and initial conditions $(s_0,a^{0-1})$. This is a natural consequence of the call-and-return execution, in which the quality of termination functions can be only evaluated upon entering the next state.

**Theorem 2** (Hierarchical Termination Gradient Theorem). Given an $N$ level hierarchical set of Markov options with stochastic termination functions differentiable in their parameters $\theta^l$ governing each policy $\pi^l$, the gradient of the expected discounted return with respect to $\theta^l$ and initial conditions $(s_0,a^{0-1})$ is:

$$\nabla_{\theta^l} Q_{\theta^l}(s,a^{l-1}) = \sum_{a^l} \sum_{s'} \pi^l(s'|s,a^{l-1}) \sum_{i,l'} p_i'(s'|a^l) \nabla_{\theta^l} U_{\theta^l}(s,a^l).$$

where $\mu_a(s,a^{l-1}|s_0,a^{0-1})$ is a discounted weighting of augmented state tuples along trajectories starting from $(s_0,a^{0-1})$.

We would now like to empirically validate the efficacy of our proposed hierarchical option-critic model. We achieve this by exploring popular benchmarks in the tabular and non-linear function approximation settings. In each case we implement an agent that is restricted to primitive actions (i.e. $N = 1$), an agent that leverages the option-critic architecture (i.e. $N = 2$), and an agent with the hierarchical option-critic architecture at $N = 3$. We will demonstrate that complex RL problems may be more easily learned using beyond two levels of abstraction and that the hierarchical option-critic architecture can successfully facilitate this level of learning using data from scratch.

For our tabular architectures, we followed protocol from [1] and chose to parameterize the intra-option policies with softmax distributions and the terminations with sigmoid functions. The policy over options was learned using intra-option Q-learning. We also implemented primitive actor-critic using a softmax policy. For the non-linear function approximation setting, we trained our agents using A3C [19]. Our primitive action agents conduct A3C training using a convolutional network when there is image input followed by an LSTM to contextualize the state. This way we ensure that benefits seen from options are orthogonal to those seen from these common neural network building blocks. We follow [6] to extend A3C to the Option-Critic and Hierarchical Option-Critic architectures.

We include detailed algorithm descriptions for our experiments in the tabular and non-linear function approximation settings in Appendix B. We also conducted hyperparameter optimization that is summarized along with detail on experimental protocol in Appendix B. In all of our experiments, we made sure that the two-level option-critic architecture had access to more total options than the three level alternative and that the three level architecture did not include the optimization of any additional hyperparameters. This ensures that empirical gains are the result of increasingly abstract options.

### 5.1 Tabular Learning Challenge Problems

**Exploring four rooms**: We first consider a navigation task in the four-rooms domain [29]. Our goal is to evaluate the ability of a set of options learned fully autonomously to learn an efficient exploration policy within the environment. The initial state and the goal state are drawn uniformly from all open non-wall cells every episode. This setting makes the task highly non-stationary, since the goal changes for every episode. Additionally, primitive movements can fail with probability $\frac{1}{2}$, in which case the agent transitions randomly to one of the empty adjacent cells. The reward is $+1$ at the goal and $0$ otherwise. In Figure 3 we can clearly see that reasoning with higher levels of abstraction is critical to achieving a good exploration policy and that reasoning with three levels of abstraction results in better sample efficient learning than reasoning with two levels of abstraction. For this experiment we explore four levels of abstraction as well, but unfortunately there seem to be diminishing returns at least for this tabular setting.

**Discrete stochastic decision process**: Next, we consider a hierarchical RL challenge problem as explained in [10] with a stochastic decision process where the reward depends on the history of visited states in addition to the current state. There are 6 possible states and the agent always starts at $s_2$. The agent moves left deterministically when it chooses left action, but the action right only succeeds half of the time, resulting in a left move otherwise. The terminal state is, $s_1$ and the agent receives the reward of 1 when it first visits $s_4$ and then $s_2$. The reward for going to $s_1$ without visiting $s_4$ is 0.91. In Figure 4 we again find, reasoning with higher levels of abstraction critical to performing well at this task with reasonable sample efficiency. Both option-critic learning and hierarchical option-critic learning converge to a high quality solution surpassing performance obtained in [10]. However, it seems that reasoning with three levels of abstractions allows the agent to arrive at this solution quicker than it can with just two levels of abstraction.

### 5.2 Deep Function Approximation Problems
Multistory Building Navigation: For an intuitive look at higher level reasoning, we consider the four room problem in a partially observed setting with an 11x17 grid at each level of a seven level building. The agent has a receptive field size of 3 in both directions, so observations for the agent are 9-dimension feature vectors with 0 in empty spots, 1 where there is a wall, 0.25 if there are stairs, or 0.5 if there is a goal location. The stairwells in the north east corner of the floor lead upstairs to the south west corner of the next floor up. Stairs in the south west corner of the floor lead down to the north east corner of the floor below. Agents start in a random location in the basement (which has no south west stairwell) and must navigate to the roof (which has no north east stairwell) to find the goal in a random location. The reward is +1.0 for finding the goal and -0.1 for hitting a wall. This task could seemingly benefit from abstraction given the benefit of planning using a composition of sub-policies to get to the stairs at each intermediate level. In Figure 5 we can see a qualitative difference between the policies learned with three levels of abstraction which has high variance, but fairly often finds the goal location and those learned with less abstraction. The option-critic and actor-critic model are hovering around zero reward, which qualitatively equates to just learning a policy that does not run into walls.

Learning many Atari games with one model: We finally consider application of the hierarchical option-critic architecture to the Atari games [2]. Evaluation protocols for the Atari games are famously inconsistent [13], so to ensure for fair comparisons we implement apples to apples versions of our baseline architectures deployed with the same code-base and environment settings. We put our models to the test and consider a very challenging setting [24] where a single agent attempts to learn many Atari games at the same time. Our agents attempt to learn 21 Atari games simultaneously2, matching the largest previous multi-task setting on Atari [24]. Our tasks are handed picked to fall into three categories of related games each with 7 games represented. The first category is games that include maze style navigation (e.g. MsPacman), the second category is mostly fully observable shooter games (e.g. SpaceInvaders), and the final category is partially observable shooter games (e.g. BattleZone). We train each agent by always sampling the game with the least training frames after each episode, ensuring the games are sampled very evenly throughout training. We also clip rewards to allow for equal learning rates across tasks [18]. We train each game for 10 million frames (i.e. 210 million total frames) and report statistics evaluating the clipped reward achieved by each agent when evaluating the policy without learning for another 3 million frames on each game across 5 separate training runs. As our main performance metric, we report the summary of how each multi-task agent maximizes the reward signal it receives in Table 1. While all models struggle in this difficult setting, a hierarchy of options is better able to exploit commonalities across games despite using fewer parameters and policies.

Analysis of Learned Options: An advantage of the multi-task setting is it allows for a degree of quantitative interpretability regarding when and how options are used. We report characteristics of the agent with median performance during the evaluation period. The option-critic architecture with 16 options uses 5 options the bulk of the time (31%, 23%, 11%, 9% and 8% respectively) with the rest of the time largely split among another 6 options. The average number of time steps between switching options has a pretty narrow range across games falling between 3.4 (Solaris) and 5.5 (MsPacman). In contrast, hierarchical option-critic with three options at each branch of the hierarchy learns to switch options at a rich range of temporal resolutions. The most popular high level option (used 43% of the time) applies to a mix of game types with pretty evenly used low level options lasting on average between 1.5 and 3.1 steps depending on the game and high level options ranging from 6.5 to 6.9 steps on average. Interestingly, we find the other high level options are used pretty much evenly with one specializing in shooter games and one in maze games. The shooter game option is distributed pretty evenly at the lower level while the maze game option uses two of its lower level options. One of its lower level options (used 85% of the time) lasts 7.7 to 9.1 steps on average with the higher level option almost always switching once it does. The other low level option is used almost exclusively for BankHeist. In the appendix we provide additional statistics on how options specialize across games.

6 Conclusion
In this work we propose the first policy gradient theorems to optimize an arbitrarily deep hierarchy of options to maximize the expected discounted return. Moreover, we have proposed a particular hierarchical option-critic architecture that is the first general purpose reinforcement learning architecture to successfully learn options from data with more than two abstraction levels. We have conducted extensive empirical evaluation in the tabular and deep non-linear function approximation settings. In all cases we found that, for significantly complex problems, reasoning with more than two levels of abstraction can be beneficial for learning. While the performance of the hierarchical option-critic architecture is impressive, we envision our proposed policy gradient theorems eventually transcending it in overall impact. Although the architectures we explore in this paper have a fixed structure and fixed depth of abstraction for simplicity, the underlying theorems can also guide learning for much more dynamic architectures that we hope to explore in future work.
Acknowledgements

The authors thank Murray Campbell, Xiaoxiao Guo, Ignacio Cases, and Tim Klinger for fruitful discussions that helped shape this work.

References


A Derivation of Generalized Policy Gradient and Termination Gradient Theorems

A.1 The Derivation of U

To help explain the meaning and derivation of equation (10), we separate the expression into four primary terms. The first term is applicable for $N \geq 1$ and represents the expected return from cases where no options terminate. The second term is applicable for $N \geq 2$ and represents the expected return from cases where every option terminates. The third and fourth terms are applicable for $N \geq 3$ and represent the expected return from cases where some options terminate. We will first discuss how to estimate the return when there are no terminated options. In this case we simply use our estimate of the value of the current state following the current options if there are any. As we are computing the expectation, we also multiply this term by its likelihood of happening which
We must establish the Markov chain along which we can measure performance for options with $N$ steps with the existence of a unique stationary distribution over the augmented state space. (13) is homogeneous. Additionally, when options are available at every state, the process is ergodic of learning algorithms in [1] to an option hierarchy with $\pi$. Next we turn our attention to estimating the return when all options are terminated. This can be approximated using our estimate of the return given the optimal value for an $\pi$. When $N = 2$, equation (10) simplifies to equation (3): This expression is precisely the option value function upon arrival at the option-critic framework derived in [1].

The final quantity we will estimate bridges the gap to cases where only some options terminate. When all options are terminated, we can analyze the likelihood that at each level the lower level options terminate while the current does not. In such a case, we multiply this likelihood by the value one level more abstract than the current option hierarchy level. For convenience in our derivation, we split our notation for this quantity into two separate terms accounting explicitly for the case when only lower level options terminate.

A.2 Generalized Markov Chain and Augmented Process

We must establish the Markov chain along which we can measure performance for options with $N$ levels of abstraction. The natural approach is to consider the chain defined in the augmented state space because state and active option based tuples now play the role of regular states in a usual Markov chain. If options $\phi_{s+1}^{N-1}$ have been initiated or are executing at time $t$ in state $s$, then the probability of transitioning to $(s+1, o_{s+1}^{N-1})$ in one step is:

$$p^{t}(s+1, o_{s+1}^{N-1}; s, o_{s}^{N-1}) = \frac{p[s+1|s, o_{s}^{N-1}]}{1 - \beta_{t}^{N-1}(s, o_{s}^{N-1})} +$$

$$= \sum_{j=1}^{N} (1 - \beta_{t}^{N-1}(s, o_{s}^{N-1})) \prod_{k=1}^{j-1} \beta_{t}^{k}(s, o_{s}^{k}) \prod_{k=j+1}^{N} \pi[s+1|s, o_{s+1}^{N-1}]$$

(13)

Like the Markov chain derived for the option critic architecture [1], the process given by equation (13) is homogeneous. Additionally, when options are available at every state, the process is ergodic with the existence of a unique stationary distribution over the augmented state space tuples.

We continue by presenting an extension of results about augmented processes used for derivation of learning algorithms in [1] to an option hierarchy with $N$ levels of abstraction. If options $\phi_{s}^{N-1}$ have been initiated or are executing at time $t$, then the discounted probability of transitioning to $(s+1, o_{s+1}^{N-1})$ where $\ell < N$ is:

$$p^{t}(s+1, o_{s+1}^{N-1}; s, o_{s}^{N-1}) = \frac{p[s+1|s, o_{s}^{N-1}]}{1 - \beta_{t}^{N-1}(s, o_{s}^{N-1})} +$$

$$= \sum_{j=1}^{N} (1 - \beta_{t}^{N-1}(s, o_{s}^{N-1})) \prod_{k=1}^{j-1} \beta_{t}^{k}(s, o_{s}^{k}) \prod_{k=j+1}^{N} \pi[s+1|s, o_{s+1}^{N-1}]$$

(14)

As such, when we condition the process from $(s, o_{s}^{N-1})$, the discounted probability of transitioning to $(s+1, o_{s+1}^{N-1})$ is:

$$p^{t}(s+1, o_{s+1}^{N-1}; s, o_{s}^{N-1}) = \frac{p[s+1|s, o_{s}^{N-1}]}{1 - \beta_{t}^{N-1}(s, o_{s}^{N-1})} +$$

$$= \sum_{j=1}^{N} (1 - \beta_{t}^{N-1}(s, o_{s}^{N-1})) \prod_{k=1}^{j-1} \beta_{t}^{k}(s, o_{s}^{k}) \prod_{k=j+1}^{N} \pi[s+1|s, o_{s+1}^{N-1}]$$

(15)

This definition will be very useful later for our derivation of the hierarchical intra-option policy gradient. However, for the derivation of the hierarchical termination gradient theorem we should
reformulate the discounted probability of transitioning to \((s_{t+1}, a_{t+1}^{(l)})\) from the view of the termination policy at abstraction level \(l\) explicitly separating out terms that depend on \(\theta'\):

\[
P_l([s_{t+1}, a_{t+1}^{(l)}]; \theta) = \prod_{j=\ell+1}^N \pi_j(o_j|s, a_{1:j-1}^{(l)})
\]

where \(P_l([s_{t+1}, a_{t+1}^{(l)}]; \theta)\) is the probability of transitioning to a state based on the augmented state space \([s, a^{(l)}]\):

\[
P([s, a^{(l)}]) = \sum_{k=1}^N \sum_{i=1}^p p_{i|k}(s_{t+1}, a_{t+1}^{(l)}|s_{t}, a_{1:t-1}^{(l)}) P_l([s_{t+1}, a_{t+1}^{(l)}]; \theta)
\]

More generally, the \(k\)-step discounted probabilities can be expressed recursively as follows:

\[
P_l^{(k)}([s_{t+1}, a_{t+1}^{(l)}]) = \prod_{j=\ell+1}^N \pi_j(o_j|s_{t}, a_{1:j-1}^{(l)})
\]

Or rather conditioning on \(t-1\) as in equation (15):

\[
\prod_{j=\ell+1}^N \pi_j(o_j|s_{t}, a_{1:j-1}^{(l)})
\]

We can simplify our expression:

\[
\frac{\partial U'(s', a', s^{(l-1)}, \theta')}{\partial \theta'} = -\sum_{a'} \frac{\partial \pi'(s', a', s^{(l-1)})}{\partial \theta'} \frac{\partial U(s', a', s^{(l-1)})}{\partial \theta'}
\]

where \(\pi'(s', a', s^{(l-1)})\) is the probability of transitioning to a state based on the augmented state space \([s, a^{(l)}]\):

\[
P'(s'|s, a^{(l)}) = \sum_{a'} \sum_{k=1}^N p_{i|k}(s_{t+1}, a_{t+1}^{(l)}|s_{t}, a_{1:t-1}^{(l)})
\]

We continue by computing the gradient with respect to \(U\) again assuming that \(\theta'\) only appears in the intra-option policy at level \(l\) and not in any policy at another level or in the termination function:

\[
\frac{\partial U'(s', a', s^{(l-1)})}{\partial \theta'} = -\sum_{a'} \frac{\partial \pi'(s', a', s^{(l-1)})}{\partial \theta'} \frac{\partial U(s', a', s^{(l-1)})}{\partial \theta'}
\]

where \(\pi'(s', a', s^{(l-1)})\) is the probability of transitioning to a state based on the augmented state space \([s, a^{(l)}]\):

\[
P'(s'|s, a^{(l)}) = \sum_{a'} p_{i}(s'') \prod_{j=\ell+1}^N \pi_j(o_j|s, a_{1:j-1}^{(l)})
\]

A.3 Proof of the Hierarchical Intra-Option Policy Gradient Theorem

Taking the gradient of the value function with an augmented state space:

\[
\frac{\partial Q(s, a^{(l-1)})}{\partial \theta'} = \frac{\partial}{\partial \theta'} \sum_{s'} \pi'(s'|s, a^{(l-1)}) Q(s', a')
\]

Then substituting in equation 9 with the assumption that \(\theta'\) only appears in the intra-option policy at level \(l\) and not in any policy at another level or in the termination function:

\[
\frac{\partial Q(s, a^{(l-1)})}{\partial \theta'} = \sum_{s'} \frac{\partial \pi'(s'|s, a^{(l-1)})}{\partial \theta'} Q(s', a') + \pi'(s'|s, a^{(l-1)}) \frac{\partial Q(s, a^{(l-1)})}{\partial \theta'}
\]

and

\[
\frac{\partial Q(s, a^{(l-1)})}{\partial \theta'} = \pi'(s'|s, a^{(l-1)}) \frac{\partial U(s', a', s^{(l-1)})}{\partial \theta'}
\]

We can simplify our expression:

\[
\frac{\partial U'(s', a', s^{(l-1)}, \theta')}{\partial \theta'} = -\sum_{a'} \sum_{k=1}^N \prod_{j=\ell+1}^N \pi_j(o_j|s_{t}, a_{1:j-1}^{(l)})
\]

We can simplify our expression:

\[
\frac{\partial U'(s', a', s^{(l-1)}, \theta')}{\partial \theta'} = -\sum_{a'} \sum_{k=1}^N \prod_{j=\ell+1}^N \pi_j(o_j|s_{t}, a_{1:j-1}^{(l)})
\]

We can simplify our expression:

\[
\frac{\partial U'(s', a', s^{(l-1)}, \theta')}{\partial \theta'} = -\sum_{a'} \sum_{k=1}^N \prod_{j=\ell+1}^N \pi_j(o_j|s_{t}, a_{1:j-1}^{(l)})
\]
We proceed by substituting (24) into (20):

\[
\frac{\partial Q_{\Omega}(s,o_1:\ell-1)}{\partial \theta^\ell} = \sum_{o_\ell} \left( \frac{\partial \pi^\ell}{\partial \theta^\ell} \right) \frac{\partial U(s,o_1:\ell)}{\partial s} + \sum_{o_\ell} (1 - \beta_{\Omega}^{-1}(s',o_{\ell+1})) \prod_{k=1}^{\ell} \beta_k(o'_k,a'_k) \frac{\partial Q_{\Omega}(s',o_1:\ell-1)}{\partial \theta^\ell} + \\
\sum_{\ell \geq 1} \left( 1 - \beta_{\Omega}^{-1}(s',o_{\ell+1}) \right) \prod_{k=1}^{\ell} \beta_k(o'_k,a'_k) \frac{\partial Q_{\Omega}(s',o_1:\ell-1)}{\partial \theta^\ell} \tag{25}
\]

This yields a recursion, which can be further simplified to:

\[
\frac{\partial Q_{\Omega}(s,a_{\ell-1}^{\ell})}{\partial \theta^\ell} = \sum_{o_\ell} \frac{\partial \pi^\ell}{\partial \theta^\ell} \frac{\partial U(s,a_{\ell-1}^{\ell})}{\partial s} + \sum_{o_\ell} (1 - \beta_{\Omega}^{-1}(s',o_{\ell+1})) \prod_{k=1}^{\ell} \beta_k(o'_k,a'_k) \frac{\partial Q_{\Omega}(s',a_{\ell-1}^{\ell})}{\partial \theta^\ell} \tag{26}
\]

Using the previous remarks about an augmented process and substituting in equation 15, this expression can be transformed into:

\[
\frac{\partial Q_{\Omega}(s,a_{\ell-1}^{\ell})}{\partial \theta^\ell} = \sum_{s_1} \sum_{a_1} \beta_1(s_1,a_1|s_0,a_0) \sum_{o_1} \frac{\partial \pi^1}{\partial \theta^1} \frac{\partial U(s_1,a_1|s_0,a_0)}{\partial s_1} = \sum_{s_1} \sum_{a_1} \beta_1(s_1,a_1|s_0,a_0) \frac{\partial Q_{\Omega}(s_1,a_1|s_0,a_0)}{\partial \theta^1} \tag{27}
\]

The gradient of the expected discounted return with respect to \( \theta^1 \) is then:

\[
\frac{\partial Q_{\Omega}(s_0,a_0^{X-1})}{\partial \theta^0} = \sum_{s_1} \sum_{a_1} \beta_1(s_1,a_1|s_0,a_0) \frac{\partial \pi^1}{\partial \theta^1} \frac{\partial U(s_1,a_1|s_0,a_0)}{\partial s_1} = \sum_{s_1} \sum_{a_1} \beta_1(s_1,a_1|s_0,a_0) \frac{\partial Q_{\Omega}(s_1,a_1|s_0,a_0)}{\partial \theta^1} = \sum_{s_1} \sum_{a_1} \beta_1(s_1,a_1|s_0,a_0) \frac{\partial Q_{\Omega}(s_1,a_1|s_0,a_0)}{\partial \theta^1} \tag{28}
\]

A.4 Proof of the Hierarchical Termination Gradient Theorem

The expected sum of discounted rewards starting from \((s_1,a_0^{X-1})\) is given by:

\[
U(s_1,a_0^{X-1}) = E \sum_{s_0} \beta_1(s_0,a_0^{X-1}) \tag{29}
\]

We start by reformulating \( U \) from equation 10 at level of abstraction \( \ell \) rather than \( \ell - 1 \) as follows:

\[
u(s',a') = \left( 1 - \beta_{\Omega}^{-1}(s',o_{\ell+1}) \right) \prod_{k=1}^{\ell} \beta_k(o'_k,a'_k) \tag{30}
\]

As we will be interested in analyzing this expression with respect to \( \theta^\ell \), we separate the term where only lower level options terminate in two separate terms. In the special case where \( \ell + 1 \) terminates and \( \ell \) does not, we still utilize \( \theta^\ell \) even though it did not terminate:

\[
u(s,a) = \left( 1 - \beta_{\Omega}^{-1}(s',o_{\ell+1}) \right) Q_{\Omega}(s',a') + \nu(s) \prod_{k=1}^{\ell} \beta_k(o'_k,a'_k) \tag{31}
\]

16

17
The original expression of $U$ was more useful for the gradient with respect to $\theta'$, which does not depend on this case. The gradient of $U$ with respect to $\theta'$ is then:

$$
\frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} = \nabla_U(s(i)) \frac{\partial \beta'_k(\varepsilon', \omega', \lambda')}{\partial \theta'} + \sum_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda') \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} + \sum_{k=1}^{n_{\text{all}}} \beta'_k(\varepsilon', \omega', \lambda') \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'}
$$

The terms are:

1. **All options terminate**:
   $$
   \nabla_U(s(i)) \frac{\partial \beta'_k(\varepsilon', \omega', \lambda')}{\partial \theta'}
   $$

2. **Only lower level options terminate**:
   $$
   \sum_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda') \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'}
   $$

3. **Some relevant higher level options terminate**:
   $$
   \sum_{k=1}^{n_{\text{all}}} \beta'_k(\varepsilon', \omega', \lambda') \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'}
   $$

We define the probability weighted advantage of not terminating $A_2$ as:

$$
A_2(\varepsilon', \omega', \lambda') = Q(\varepsilon', \omega', \lambda') - \nabla_U(s(i)) \prod_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda') - \sum_{k=1}^{n_{\text{all}}} \beta'_k(\varepsilon', \omega', \lambda') Q(\varepsilon', \omega', \lambda') \prod_{j=k+1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda')
$$

We proceed to substitute equation 34 into equation 33:

$$
\frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} = -\prod_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda') \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} A_2(\varepsilon', \omega', \lambda') + (1 - \beta'_1(\varepsilon', \omega', \lambda')) \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} + \prod_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda')
$$

Next we integrate out our last three terms so that they are in terms of a common derivative:

$$
\frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} = -\prod_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda') \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} A_2(\varepsilon', \omega', \lambda') + (1 - \beta'_1(\varepsilon', \omega', \lambda')) \frac{\partial U(\varepsilon', \omega', \lambda')}{\partial \theta'} + \prod_{j=1}^{n_{\text{all}}} \beta'_j(\varepsilon', \omega', \lambda')
$$

18

19
We can then simplify the expression:

\[
\frac{\partial U(\ell', o^\ell)}{\partial \phi^\ell} = -\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell', o^\ell) \frac{\partial \beta'_k(\ell', o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell', o^\ell) \frac{\partial \beta'_k(\ell', o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell', o^{\ell-1})) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1} = \sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

We now note that substituting equation 21 into equation 12 yields:

\[
\frac{\partial Q_\ell(o^\ell)}{\partial \phi^\ell} = \mathbf{p}(o^\ell) \frac{\partial U(\ell', o^\ell)}{\partial \phi^\ell}
\]

Substituting this expression into equation 37 we find that:

\[
\frac{\partial U(\ell', o^\ell)}{\partial \phi^\ell} = -\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell', o^\ell) \frac{\partial \beta'_k(\ell', o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell', o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

Using the structure of the augmented process and substituting in equation 16:

\[
\frac{\partial U(\ell', o^\ell)}{\partial \phi^\ell} = -\sum_{i=0}^{n_i} \beta'_i(o^\ell) \frac{\partial \beta'_i(o^\ell)}{\partial \phi^\ell} A_{\ell^i}(o^\ell) +
\]

\[
\sum_{i=0}^{n_i} \beta'_i(o^\ell) \frac{\partial \beta'_i(o^\ell)}{\partial \phi^\ell} A_{\ell^i}(o^\ell) +
\]

\[
\sum_{i=0}^{n_i} \beta'_i(o^\ell) \frac{\partial \beta'_i(o^\ell)}{\partial \phi^\ell} A_{\ell^i}(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

we finally obtain:

\[
\frac{\partial U(\ell', o^\ell)}{\partial \phi^\ell} = -\sum_{i=0}^{n_i} \beta'_i(o^\ell) \frac{\partial \beta'_i(o^\ell)}{\partial \phi^\ell} A_{\ell^i}(o^\ell) +
\]

\[
\sum_{i=0}^{n_i} \beta'_i(o^\ell) \frac{\partial \beta'_i(o^\ell)}{\partial \phi^\ell} A_{\ell^i}(o^\ell) +
\]

\[
\sum_{i=0}^{n_i} \beta'_i(o^\ell) \frac{\partial \beta'_i(o^\ell)}{\partial \phi^\ell} A_{\ell^i}(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
\sum_{j=0}^{N-1} \sum_{k=1}^{q_j} \beta'_k(\ell, o^\ell) \frac{\partial \beta'_k(\ell, o^\ell)}{\partial \phi^\ell} A_\ell(o^\ell) +
\]

\[
(1 - \beta_{p_0}^{-1}(\ell, o^{\ell-1}) \mathbf{1}_{\ell=1} + \sum_{i=1}^{n_i} \mathbf{1}_{\ell^i = 1}
\]
Algorithm 1: Hierarchical Option-Critic with Tabular Intra-option Q-Learning

procedure LEARN EPISODE (N, α , γ, π, β , k)
    s ← s0
    repeat
        choose action a according to π(a|s)
        take action a in s, observe r and s′
        calculate the expected discounted return
        if s is non-terminal then
            r ← r + γQ(s′,a|αθ)
        else
            r ← 0
        δ ← γQ(s,a|αθ) + αδ
        Q(s,a|αθ) ← Q(s,a|αθ) + δ
        update the termination policies
        for j = 1,...,N do
            Qj(s,a|αθ) ← Qj(s,a|αθ) + δj
        update the critic policies
        for j = 1,...,N do
            θj ← θj + αΩj(πj|N)
    until episode ends or t ≥ T
    return o1:k

Algorithm 2: Asynchronous Advantage Hierarchical Option-Critic

procedure LEARN EPISODE (N, α , γ, π, β , k)
    s ← s0
    repeat
        choose action a according to π(a|s)
        take action a in s, observe s′ and r
        –calculate the expected discounted return
        if s is non-terminal then
            r ← r + γQ(s′,a|αθ)
        else
            r ← 0
        δ ← γQ(s,a|αθ) + αδ
        Q(s,a|αθ) ← Q(s,a|αθ) + δ
        update the termination policies
        for j = 1,...,N do
            Qj(s,a|αθ) ← Qj(s,a|αθ) + δj
        update the critic policies
        for j = 1,...,N do
            θj ← θj + αΩj(πj|N)
    until episode ends or t ≥ T
    return o1:k

Learning Abstract Options

Hierarchical Option-Critic with Tabular Intra-option Q-Learning
procedure LEARN.Options (i, α , β , k)
    s ← s0
    repeat
        take action a according to π(a|s)
        choose option j according to πj|N
        –calculate the expected discounted return
        if s is non-terminal then
            r ← r + γQ(s′,a|αθ)
        else
            r ← 0
        δ ← γQ(s,a|αθ) + αδ
        Q(s,a|αθ) ← Q(s,a|αθ) + δ
        update the termination policies
        for j = 1,...,N do
            Qj(s,a|αθ) ← Qj(s,a|αθ) + δj
        update the critic policies
        for j = 1,...,N do
            θj ← θj + αΩj(πj|N)
    until episode ends or t ≥ T
    return o1:k

Asynchronous Advantage Hierarchical Option-Critic
procedure LEARN.Options (i, α , β , k)
    s ← s0
    repeat
        take action a according to π(a|s)
        choose option j according to πj|N
        –calculate the expected discounted return
        if s is non-terminal then
            r ← r + γQ(s′,a|αθ)
        else
            r ← 0
        δ ← γQ(s,a|αθ) + αδ
        Q(s,a|αθ) ← Q(s,a|αθ) + δ
        update the termination policies
        for j = 1,...,N do
            Qj(s,a|αθ) ← Qj(s,a|αθ) + δj
        update the critic policies
        for j = 1,...,N do
            θj ← θj + αΩj(πj|N)
    until episode ends or t ≥ T
    return o1:k

update global parameters with thread gradients
until T > Tmax
procedure CHOOSE TERMINATED OPTIONS (i, α , β , k)
    if β(s,a|k) = 1
      if k = 1
        a ← π1(s)
      else
        a ← CHOOSE TERMINATED OPTIONS (i, α , β , k − 1)
    else
      a ← CHOOSE TERMINATED OPTIONS (i, α , β , k − 1)
    return a

Section 3
Optimal Action
to produce an option policy. The perceptual module was a 100 unit fully connected layer with relu activations. This perceptual module is processed by a 256 unit LSTM network with gradients truncated at 20 steps. Every intra-option policy, termination policy, and critic simply consists of one linear layer on top of this core module followed by a softmax in the case of intra-option policies and a sigmoid in the case of termination policies.

Hyperparameters: We found optimal to use a learning rate of 1e-4 for all models a well as 16 parallel asynchronous threads, a deliberation cost of 0.01 for each termination policy, and entropy regularization of 0.01 on the intra-option policies [1, 6].

Learning curve details: We set our implementation of A3C to report recent learning performance after approximately 1 minute of training. Each minute we report the rolling mean reward calculated using a horizon of 0.99. To plot learning performance we take the average and standard deviation of the reported rewards over the past 1 million frames.

B.4 Atari game experiments

Architecture details: A core perceptual and contextualization model is shared across all policies and critics for each model to transform observations into conceptual states that can be processed to produce an option policy. The perceptual module consists of two convolutional layers with strides of 2 and 4, and with 8 and 4 filters respectively each followed by an relu activation function. This perceptual module is processed by a fully connected layer and a 256 unit LSTM network with gradients truncated at 20 steps. Every intra-option policy, termination policy, and critic simply consists of one linear layer on top of this core module followed by a softmax in the case of intra-option policies and a sigmoid in the case of termination policies.

Hyperparameter search: In our experiments on Atari we followed conventions from past work using 16 parallel asynchronous threads, a deliberation cost of 0.01 for each termination policy, and entropy regularization of 0.01 on the intra-option policies [1, 6]. The one parameter we conduct a search over is the learning rate in the range \{1e-4, 5e-4, 1e-3\} for each model. The exceptions include Centipede where the primitive action A3C policy benefits from a learning rate of 5e-4, and Berzerk where the hierarchical option-critic model benefits from a rate of 5e-4. Finally, in Zaxxon we find that the primitive action learning rate should be 1e-3 and the hierarchical option-critic learning rate should be 5e-4.

Learning curve details: We set our implementation of A3C to report recent learning performance after approximately 1 minute of training. Each minute we report the rolling mean reward calculated using a horizon of 0.99. To plot learning performance we take the average and standard deviation of the reported rewards over the past 1 million frames.
Semi-Black Box: Rapid Development of Planning Based Solutions

Michael Katz
IBM T.J. Watson Research Center
Yorktown Heights, NY, USA
michael.katz1@ibm.com

Abstract

Software developers nowadays not infrequently face a challenge of solving problems that essentially sum up to finding a sequence of deterministic actions leading from a given initial state to a goal. This is the problem of deterministic planning—one of the most basic and well studied problems in artificial intelligence. Two of the best known approaches to deterministic planning are the black box approach, in which a program implements a successor generator, and the model-based approach, in which a user describes the problem symbolically, e.g., in PDDL. While the black box approach is usually easier for programmers who are not experts in AI to understand, it does not scale up without informative heuristics. We propose an approach that we baptize as semi-black box (SBB) that combines the strength of both. SBB is implemented as a set of Java classes, which a programmer can inherit from when implementing a successor generator. Using the known characteristics of these classes, we then automatically derive heuristics for the problem. Our empirical evaluation shows that these heuristics allow the planner to scale up significantly better than the traditional black box approach.

Introduction

The field of artificial intelligence has spent considerable effort on the seemingly simple problem of deterministic planning. At a high level, this problem can be formulated as: given an initial state, a desired goal, and a set of possible (deterministic) actions, find a sequence of actions which leads from the initial state to a state satisfying the goal. One popular approach to solving deterministic planning problems is heuristic search. However, two very different ways of using heuristic search algorithms to solve deterministic planning problems have become prevalent: using heuristic search algorithms to solve planning problems which have been pursued throughout the history of the field. The first approach, which we will refer to as the "black box" approach, involves implementing a piece of software to represent the planning problem. While the details of deterministic planning problems can be quite complex, it is enough to implement a very simple interface consisting of three functions: GET-INIT-STATE(s), which returns an object representing the initial state, GET-SUCCESSORS(s), which returns the successors of a given state s, and IS-GOAL?(s), which checks whether the given state s is a goal state. Standard forward search algorithms, such as breadth first search,

Copyright © 2018, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
In this page, we describe such a framework, which brings the benefits of the black-box approach, namely automatically derived heuristics, into black box successor generator planning. The key insight behind our framework is that, while planning problems can vary in their details, there are some common underlying principles behind the vast majority of these problems (Mcdermott and Fox 2000). Our framework provides an implementation of these common principles, which is transparent to the model-based view, yet can still be used inside a “black box” implementation.

Background

We now describe the two approaches we mention above in more detail. We begin by defining a deterministic planning problem over a state space, which is a tuple $\mathcal{P} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, f)$, where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of action labels, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the set of goal states, and $f : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the transition function, such that $f(s,a,s')$ is the state which applying action $a$ in state $s$ leads to. To a solution to such a problem is a sequence of action labels $\pi = (a_0, a_1, \ldots, a_n)$ such that $f(s_0(a_0, a_1, \ldots, a_n), s_n) = s_1$, that is, a sequence of action labels which achieve a goal state in some goal state, using the transition function $f$.

While deterministic planning over a state space provides a nice mathematical model, the question of how the state space is described has more than one answer. The “black box” approach is described in an entirely abstract way. The “black box” description of state space is described as a tuple $(\mathcal{S}_0, \mathcal{S}, f)$, where $\mathcal{S}_0$ is the initial state, $\mathcal{S}$ is the set of states, and $f : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the transition function. Note that it is always possible to create a PDDL description $\mathcal{P}$ of this black box of state space planning problem II, by defining a predicate of arity 0 for each state $s \in \mathcal{S}$. However, the number of states that is described by this planning problem is exponential in $|\mathcal{S}|$. Finding a compact PDDL description of a state space planning problem II requires understanding of the structure of $\mathcal{S}$, and is not always an easy task.

Additionally, PDDL is not always easy to deal with. For example, the occasional need to define actions with many parameters has been addressed by automatic domain transformations (Arceas et al. 2014). PDDL also makes the “closed structure” assumption, that states in the current state can be represented by a set of variables. Different mathematical formalisms for such models exist (Fikes and Nilsson 1971; Bacătomir and Nebel 1995), but we will only describe on PDDL (Meder et al. 1998), which includes both a mathematical formalism and a syntax for writing text files describing a planning problem in this formalism.

For ease of presentation, we describe a limited subset of PDDL, which captures what Fikes and Nilsson (1971) describe a planning task in PDDL is described by a tuple $\mathcal{P} = (\mathcal{S}_0, \mathcal{S}, f)$, where $\mathcal{S}_0$ is the initial state, $\mathcal{S}$ is the set of states, and $f : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the transition function, such that $f(s,a,s')$ is the state which applying action $a$ in state $s$ leads to. A solution to such a problem is a sequence of action labels $\pi = (a_0, a_1, \ldots, a_n)$ such that $f(s_0(a_0, a_1, \ldots, a_n), s_n) = s_1$, that is, a sequence of action labels which achieve a goal state in some goal state, using the transition function $f$.

Additionally, the framework provides operator stereotypes, which correspond to common operations on the entity stereotypes: move, which moves a mobile entity from one place to another, load which changes the location of a mobile entity inside a mobile entity, and unload, which changes the location of the entity and action stereotypes, and implement the desired additional behavior. Thus, the framework implements a black box description of the planning problem $\mathcal{P}$, and the only way to interact with it is through the abstraction $f$. This framework is implemented in an object-oriented language (specifically, in Java), the developer can inherit from the entity and action stereotypes, and implement the desired additional behavior, on top of what the framework already offers. Additionally, the framework provides operator stereotypes, which correspond to common operations on the entity stereotypes: move, load, and unload.

Examples

We now demonstrate the advantages of the Semi-Black-Box approach on two concrete examples.

Commuter Pooling Domain

Our first example demonstrates the simplicity of modeling with the Semi-Black-Box approach. We model a commuter pooling planning problem, where co-workers share rides on their way to work and back home. A commuter pooling planning problem is defined by a set of participants $P$, which are mobile entities, as well as their home locations $L$ and a work location $w$ which are places. Some participants have a vehicle with limited capacity available, which is also a mobile entity. Figure 1 describes how these are implemented in our Semi-Black-Box framework, and shows how Vehicles and Participants are the sharing domain from MobileEntity. Each participant $p \in P$ is initially at her home location $hd(p)$, which she can leave no earlier than $hd(p)$, and arrive to the work location no later than $w1(p)$. On her way home, she can leave her work location no earlier than $w1(p)$.
### Figure 2: An example fragment visualizing a commuter pooling problem with 8 participants, comparing (a) a basic solution of duration and distance of moving from participant’s home location. Each participant with an available vehicle as a rider, as long as its full capacity is not (satisfied at (time (E, T))) can be inside another entity (in (E, E’))

### Figure 3: Drive action in commuter pooling domain.

than hapl(p). These are implemented as temporally extended goals on p. Figure 2 exemplifies solution visualization on a map, comparing a naive solution of each participant taking her own vehicle (a solution that is not guaranteed to exist in general) to a solution found by our planner, which is described below. To model the roadmap, we implement the fLoca-
tionService interface, which specifies travel time and cost between different places. For each pair of locations, l₁, l₂, a duration and distance of moving from l₁ to l₂ are given by T(l₁, l₂) and D(l₁, l₂), respectively.

Each participant’s vehicle is initially located at that partic-

### Figure 4: Board action in commuter pooling domain.

### Evolution Domain

Our second example demonstrates that the Semi-Black Box approach is more expressive than PDDL. Our objective here is to create an organism that will emerge the qualities of sev-
eral organisms, a common task in evolutionary biology. An Evolution planning task is defined by a set of organ-

### Figure 5: Example implementation of Reproduce action in Evolution domain.

### Planning with Semi Black Box

Having described our representation framework, we must now describe how we can solve problems formulated in this representation. We have already described how we can im-

ple
t a successor generator and a goal test, and there-

we can use any uninformed search algorithm, such as BFS, DFS, ID-DFS (Korf 1985), etc. to solve the problem. However, uninformed search will not scale to large problem sizes. In order to be able to scale up, we must make use of the extra information we have available — the model-based portion of the representation. Since we already have some known operator stereotypes, we exploit our knowledge of how these affect some aspects of the entities they are ap-

plied to, which also have known stereotypes. We do this by deriving a heuristic evaluation function, which estimates the distance from a given state to the goal. This allows us to use informed search algorithms, such as GBFS or weighted A* and solve larger problems.

Our framework provides operator stereotypes with known behavior, and entity stereotypes with known properties. Therefore, we derive a heuristic estimate of the distance to the goal by first “projecting” the problem onto its known aspects (that is, the known properties of entities and known behavior of operators), and then deriving a heuristic estimate for this projection. Note that this projection is not a true ab-

traction in the formal sense of the word, as an operator with a known stereotype can modify its inherited behavior in ar-

bitrary ways. However, that would constitute poor software engineering, and our purpose here is to provide a useful tool for software developers. Furthermore, even if the program-

mer did do this, it would only lead to inaccurate heuristic estimates, but will never affect the correctness of the plan that is returned.

We illustrate this point for our prototype implementation on mobile entities, and provide a PDDL-like description of this projection. The objects in our PDDL description are the set of entities and locations. The predicates we use are:

- Each mobile entity $E$ can be in location $L$ (at($E$, $L$))
- Each mobile entity $E$ can be inside another entity $E’$ (in($E$, $E’$))
- Each temporal entity (including mobile ones) has a clock with value $T$ (time($T$))
- For each mobile entity’s temporally extended goal loca-
tions $G$, we need to indicate whether it was satisfied or not (satisfied($G$))

Finally, we can describe the effects of move, load, and un-

load using the above predicates.

While one might think it is possible to use any of the ex-

isting heuristics from the model-based planning community, there is a subtle issue here — unlike in PDDL, it is pos-
able to add or delete entities on the fly in our framework.

Therefore, if we ground the projection according to the ini-
tial state, as is commonly done in model-based planning, we might end up deriving a heuristic for the wrong problem as soon as some entity is added or deleted. Another issue is that with temporal entities, we can not ground their clocks.

```java
public class Reproduce extends AbstractTemporalAction {
    private Sex sexOfTheChild;
    private long duration;
    public Reproduce(String actionId, Sex sexOfTheChild, long duration) {
        this.sexOfTheChild = sexOfTheChild;
        this.duration = duration;
    }

    @Override
    public boolean isApplicable(IState s, IEntity[] pars) {
        return super.isApplicable(s, pars) &&
                (female.getSex() != Sex.FEMALE) return false;
    }

    @Override
    public void apply(IState s, IEntity[] params) {
        Vehicle v = (Vehicle) params[1];
        if (capacity < v.getMaximalCapacity())
            v.setVehicleCurrentCapacity(capacity+1);
    }
}
```
Table 1: Empirical Results on Commuter Pooling Domain.

<table>
<thead>
<tr>
<th>Task</th>
<th>LAMA</th>
<th>SBB</th>
<th>BB</th>
<th>LAMA</th>
<th>FF</th>
<th>SBB</th>
<th>BB</th>
<th>Optic</th>
<th>LAMA</th>
<th>FF</th>
<th>SBB</th>
<th>BB</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost</td>
<td>1.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>time</td>
<td>0.56</td>
<td>0.31</td>
<td>0.11</td>
<td>0.27</td>
<td>0.12</td>
<td>0.42</td>
<td>0.68</td>
<td>3.98</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Empirical Results for SBV on Evolution Domain.

<table>
<thead>
<tr>
<th>Task</th>
<th>SBV</th>
<th>Optic</th>
<th>LAMA</th>
<th>FF</th>
<th>SBB</th>
<th>BB</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost</td>
<td>0.37</td>
<td>0.24</td>
<td>0.32</td>
<td>0.36</td>
<td>0.14</td>
<td>0.12</td>
</tr>
<tr>
<td>time</td>
<td>0.50</td>
<td>0.20</td>
<td>0.10</td>
<td>0.20</td>
<td>0.20</td>
<td>285</td>
</tr>
</tbody>
</table>

Figure 6: An example fragment of the relaxed planning graph used for heuristic computation, for a commuter pooling problem.

There are two participants, P1, starting at 9:00 earlier than P2, and P2, starting at 9:20. There is a work location C, and two vehicles V1 of P2 and V2 of P0. Participants are described by a pair of (time, location), with 1 – standing for no driver in the vehicle.

Figure 6: An example fragment of the relaxed planning graph used for heuristic computation, for a commuter pooling problem.

There are two participants, P1, starting at 9:00 earlier than P2, and P2, starting at 9:20. There is a work location C, and two vehicles V1 of P2 and V2 of P0. Participants are described by a pair of (time, location), with 1 – standing for no driver in the vehicle.

Optimal Action

There are two participants, P1, starting at 9:00 earlier than P2, and P2, starting at 9:20. There is a work location C, and two vehicles V1 of P2 and V2 of P0. Participants are described by a pair of (time, location), with 1 – standing for no driver in the vehicle.

Discussion and future work

We introduce a framework that brings the benefits of the model based approach into black box successor generator.
planning by allowing annotating planning problem entities and actions with certain predefined stereotypes. By that, we take a major step toward making solving deterministic planning problems accessible to software developers who are not necessarily experts in artificial intelligence.

For future work, we intend to extend our framework by both introducing and exploring additional stereotypes, and by introducing additional search enhancements, such as additional automatically derived heuristics (landmarks, abstraction) and search boosting techniques, such as preferred operators. Further, state space search pruning techniques such as novelty based pruning (Lipovetzky and Geffner 2012; Katz et al. 2017) have a potential of contributing greatly to our framework, especially in domains like Evolution, where our framework fails to automatically derive a meaningful search guidance. As many real life problems exhibit nondeterministic behavior, we intend to generalize our framework to support non-deterministic action outcomes.

Yet another promising direction is introducing stereotypes into other systems, such as state space search with PSVN.

References


Abstract

It is well known that the problems of stochastic planning and probabilistic inference are closely related. This paper makes two contributions in this context. The first is to provide an analysis of the recently developed SOGBOFA heuristic planning algorithm, that was shown to be effective for problems with large factored state and action spaces. It is shown that SOGBOFA can be seen as a specialized inference algorithm that computes its solutions through a combination of a symbolic variant of belief propagation and gradient ascent. The second contribution is a new solver for Marginal MAP (MMAP) inference. We introduce a new reduction from MMAP to maximum expected utility problems which are suitable for the symbolic computation in SOGBOFA. This yields a novel algebraic gradient-based solver (AGS) for MMAP. An experimental evaluation illustrates the potential of AGS in solving difficult MMAP problems.

1 Introduction

The connection between planning and inference is well known. Over the last decade multiple authors have introduced explicit reductions showing how stochastic planning can be solved using probabilistic inference (for example, [4, 25, 5, 17, 23, 12, 8, 19, 26, 10, 18]) with applications in robotics, scheduling and environmental problems. However, heuristic methods and search are still the best performing approaches for planning in large combinatorial state and action spaces [9, 7, 2].

This paper makes two contributions in this context. We first analyze a recent heuristic planning algorithm that was shown to be effective in practice. SOGBOFA [2] builds an approximate algebraic computation graph capturing marginals of state and reward variables under independence assumptions. It then uses automatic differentiation [6] and gradient based search to optimize action choice. Our analysis shows that the value computed by SOGBOFA’s computation graph is identical to the solution of Belief Propagation (BP) when conditioned on actions. This provides an explicit connection between heuristic planning algorithms and approximate inference. Inference through algebraic expressions has been explored before [16] and even applied to planning but both the symbolic representation and algorithms are different from the one in SOGBOFA.

Our second contribution is in showing how planning algorithms can be used to solve inference problems, making use of the correspondence in the reverse direction from prior work. The original construction for SOGBOFA can be seen to solve Maximum expected Utility (MEU) problems with...
decision variables as roots of the corresponding graphical model and one leaf node representing the value which is being optimized. This corresponds to solving MMAP problems with MAP variables at the roots and a single evidence node at a leaf. We provide a new reduction from MMAP problems to MEU whose output satisfies these requirements. When combined with the SOGSOFA solver this provides a novel inference algorithm, algebraic gradient-based solver (AGS), that can solve general MMAP problems. AGS effectively uses a symbolic variant of BP with gradient search. AGS provides an alternative to the mixed-product BP algorithm of [13] and the stochastic local search algorithm of [20]. An experimental evaluation compares AGS to state of the art algorithms for MMAP [14] and illustrates its potential in solving hard inference problems.

2 Preliminaries

Belief Propagation in Bayesian Networks: For our results it is convenient to refer to the BP algorithm for directed graphs [21]. A Bayesian Network (BN) is given by a directed acyclic graph where each node \( x \) is associated with a random variable and a corresponding conditional probability table (CPT) capturing \( p(x|\text{parents}(x)) \). The joint probability of all variables is given by \( \prod p(x|\text{parents}(x)) \). In this paper we assume that all state and action variables are binary.

Assume first that the directed graph is a polytree (no underlying undirected cycles). For node \( x \), BP calculates an approximation of \( p(x|\text{parents}(x)) \) for \( x \in \text{nodes}(x) \) where \( c \) is the total evidence in the graph. Let \( \sigma(x) = \text{p}(x|\text{parents}(x)) \) and \( \lambda(x) = \text{p}(x|\text{parents}(x)) \), where \( c \), \( e \) are evidence nodes reachable from \( x \) through its parents and children respectively. We use \( \alpha \) to represent a normalization constant and \( \beta \) to denote some constant. For a polytree, \( x \) separates its parents from its children and we have

\[
\begin{align*}
\lambda(x) & = \sum_{j \in \text{parents}(x)} \lambda_j(x) \\
\sigma(x) & = \sum_{j \in \text{parents}(x)} \sigma_j(x)
\end{align*}
\]

where \( \lambda(x) \) and \( \sigma(x) \) incorporate evidence through children and parents respectively. In (3) the sum variable \( w \) ranges over all assignments to the parents of \( x \) and \( w_k \) is the induced value to the \( k \)-th parent. \( \lambda_j(x) \) is the message that a child \( x \) sends to its parent \( x \) and \( \sigma_j(w_k) \) is the message that a parent \( w_k \) sends to \( x \). The messages are given by

\[
\begin{align*}
\lambda_j(w_k) = \beta \sum_{\lambda(x)} \sigma(w_k) \prod_{i \in \text{parents}(x)} \lambda_i(x) \\
\sigma_j(w_k) = \alpha \sum_{\lambda(x)} \lambda(x)
\end{align*}
\]

where \( \lambda(x) \) is a set of conjunctions having the same conditional probability and \( \lambda_j(w) \) is the sum of values to other parents \( w_k \). Since the nodes are binary the messages have two values (i.e., \( \lambda \) and \( \sigma \)) and the sum is over values to other parents \( w_k \) of \( x \). The algorithm is initialized by fixing \( \sigma \) and \( \lambda \) of evidence nodes to agree with the evidence, setting \( \sigma \) to root nodes equal to the prior probability, and setting \( \lambda \) to leaves of \( \text{1}\text{,1} \), i.e., an uninformed value. A node can send a message along an edge if all messages from its other edges have been received. If the graph is a polytree then two passes of messages on the graph yield \( \lambda = \sigma \) for all \( x \) [21].

The loopy BP algorithm updates the same updates even if the graph is not a polytree. In this case we initialize all messages to \( (1,1) \) and follow the same update rules for messages according to some schedule. The algorithm is not guaranteed to converge but it often does and it is known to perform well in many cases. The following property of BP is well known.

**Lemma 1.** If loopy BP is applied to a BN with no evidence, i.e., \( \lambda(x) = (1,1) \) for all \( x \) at initialization, then for any order of message updates and any time in the execution of loopy BP, for any node \( x \), \( \lambda(x) \approx (1,1) \) and \( \lambda(w) \) is a constant independent of \( w \) for any parent of \( x \). In addition, a single pass spading \( \pi \) messages in topological order converges to the final output of BP.

**Proof.** We prove the claim by induction. Assume that \( \lambda(x) = (\pi, \omega) \) for some value \( \omega \) and consider the next \( \lambda \) message from \( x \). From Eq (4) we have

\[
\lambda_j(w_k) = \beta \sum_{\lambda(x)} \sigma(w_k) \prod_{i \in \text{parents}(x)} \lambda_i(x) = \beta \sum_{\lambda(x)} \sigma_j(w_k) \prod_{i \in \text{parents}(x)} \lambda_i(x) = \beta \sum_{\lambda(x)} \sigma_j(w_k) \prod_{i \in \text{parents}(x)} \lambda_i(x) = \beta \lambda_j(w_k)
\]

where to get the second equality we extract the constant \( \lambda(x) = (1,1) \) and reorder the summations. The last equality is true because \( \sum_i p(x|\text{parents}(x)) = \lambda(x) = (1,1) \) and \( \prod_i \sum_j p(x|\text{parents}(x)) = \sum_j p(x|\text{parents}(x)) = 1 \). Therefore, \( \lambda_j(w_k) \) is a constant independent of \( w_k \). Now from Eq 2 we see that \( \lambda(w_k) = (1,1) \) as well, and from Eq 5 and 3 we see that it suffices to update \( \pi \) messages in topological order.

**AROLLout and SOGSOFA:** Stochastic planning is defined using Markov decision processes [22]. A MDP [22] is specified by \( \{ S, A, T, R, \gamma \} \), where \( S \) is a finite state space, \( A \) is a finite action space, \( T(x, a, x') = p(x'|a, x) \) defines the transition probabilities, \( R(x, a) \) is the immediate reward of taking action \( a \) in state \( x \), and \( \gamma \) is the discount factor. A policy \( \pi: S \to A \) is a mapping from states to actions, indicating which action to choose at each state. Given a policy \( \pi \), the value function \( V(x, \pi) \) is the expected discounted total reward \( E_p \sum_{k=1}^{n} \gamma^{k-1} R(x_k, \pi(x_k)) \), where \( x_k \) is the \( k \)-th state visited by following \( \pi \) (and \( \gamma \) is the discount rate). The action-value function \( Q(x, a, \pi) \) is the expected discounted total reward when taking action \( a \) in state \( x \) and following \( \pi \) thereafter. In this paper we consider finite horizon planning where the trajectories are taken to a fixed horizon \( h \) and \( \gamma = 1 \), i.e., no discounting is used.

In factored spaces [1] the state is specified by a set of variables and the number of states is exponential in the number of variables. Similarly in factored action spaces an action is specified by a set of variables. We assume that all state and action variables are binary. Finite horizon planning can be captured using a dynamic Bayesian network (DBN) where state and action variables at each time step are represented explicitly and the CPTs of variables are given by the transition probabilities. In off-line planning the task is to compute a policy that optimizes the long term total reward. In on-line planning we are given a fixed limited time \( t \) per step and cannot compute a policy in advance. Instead, given the current state, the algorithm must decide on the next action with time \( t \). Then the action is performed, a transition and reward are observed and the algorithm is presented with the next state. This process repeats and the long term performance of the algorithm is evaluated.

On-line planning has been the standard evaluation method in recent planning competitions. AROLLOut and SOGSOFA perform on-line planning by estimating the value of initial actions at the current state \( s \), \( Q(s, a) \), where a fixed rollout policy \( \pi \), typically a random policy, is used in future steps. The AROLLOut algorithm [3] introduced the idea of algebraic simulation to estimate values but optimized over actions by enumeration. Then [2] showed how algebraic rollouts can be computed symbolically and that the optimization can be done using automatic differentiation [6]. We next review these algorithms. Finite horizon planning can be translated from a high level language (e.g., RDDL [24]) to a dynamic Bayesian network (DBN). AROLLOut transforms the CPT of a node \( x \) into a disjoint sum form. In particular, the CPT for \( s \) is represented in the form \( \sum_{\text{cond}(s)} Q(c(s)) \), where \( \text{cond}(s) \) are conjunctions of parent values which are are mutually exclusive and exhaustive. In this notation \( Q(c(s)) \) is a set of conjunctions having the same conditional probability \( p(w)\cdot wp \). The algorithm then performs a forward pass computing \( p(z) \), an approximation of the true marginal \( p(z) \), for any node \( x \) in the graph. \( p(z) \) is calculated as a function of \( Q(c(s)) \), an estimate of the probability that \( c(s) \) is true, which assumes the parents are independent. This is done using the following equations where nodes are processed in the topological order of the graph:

\[
\begin{align*}
\hat{p}(z) = \sum_{w_k} \hat{p}(w_k)(Q(c(s))) \\
\hat{p}(c(s)) = \prod_{w_k} \hat{p}(w_k) \prod_{w_k} (1 - \hat{p}(w_k)).
\end{align*}
\]

The following example from [2] illustrates AROLLOut and SOGSOFA. The problem has three state variables \( x_1, x_2, x_3 \) and three action variables \( a_1, a_2, a_3 \) respectively. In addition we have two intermediate variables \( cond_1 \) and \( cond_2 \) which are not part of the state and reward are given by the following RDDL [24] expressions where primed variables of variables represent the value of the variable after performing the action.

\[
\begin{align*}
\text{cond}_1 &= \text{Bernoulli}(0.7) \\
\text{cond}_2 &= \text{Bernoulli}(0.5) \\
\text{s'}(1) &= \text{if } \text{cond}_1 \text{ then } a(1) \text{ else false} \\
\text{s'}(2) &= \text{if } \text{cond}_1 \text{ then } a(2) \text{ else false} \\
\text{s'}(3) &= \text{if } \text{cond}_2 \text{ then } a(2) \text{ else false} \\
\text{reward} &= s(1) + s(2) + s(3)
\end{align*}
\]
3 AROLLOUT is Equivalent to BP

We first show that the computation of AROLLOUT can be rewritten as a sum over assignments of \( x \) to satisfy
\[
\hat{p}(x) = \sum_{W} \hat{p}(x|W) \prod_{k} \hat{p}(w_k) w_k (1 - \hat{p}(w_k))^{1-w_k},
\]
which is identical to (8).

Proof. The sum in (8) can be divided into disjoint sets of assignments according to the \( x \) they satisfy. Consider one fixed \( x \). Let \( W_i \) be a parent of \( x \) which is not in \( c_j \). Let \( W_{ij} = \{c_j \} \) be the assignments to the parents of \( x \) which satisfy \( c_j \). Since \( W_i \) is not in \( c_j \), \( W_{ij} = \{c_j \} \) is well defined. We have that
\[
\sum_{W_{ij}} \hat{p}(x|W) \prod_{k} \hat{p}(w_k) w_k (1 - \hat{p}(w_k))^{1-w_k} \leq \sum_{W} \hat{p}(x|W) \prod_{k} \hat{p}(w_k) w_k (1 - \hat{p}(w_k))^{1-w_k}.
\]

By Lemma 1, the marginals for the action variables and using automatic differentiation. We refer the reader to [6] for details on automatic differentiation; the basic idea is similar to backpropagation of gradients in neural network learning which can be generalized to arbitrary graphs. In this manner we can perform gradient search over marginals for action variables in \( c_j \) and effectively select values for the action variables at the first step. Such graphs include several additional heuristics including dynamic control of simulation depth, dynamic selection of gradient step size, maintaining domain constraints, and a balance between gradient search and random restarts. Most of this is orthogonal to the topic of this paper and we omit the details.
4 Algebraic Solver for Marginal MAP

Marginal MAP [20, 13, 11, 14] is a complex inference problem seeking a configuration of a subset of variables that maximizes their marginal probability. Recall that the graph construction in SOGBOFA evaluates exactly to the value returned by AGROOL. Therefore, the result in the previous section shows that SOGBOFA can be understood as using gradient search for the best action where the evaluation criterion is given using BP but calculated symbolically. In this section, we show that this approach can be used for MMAP yielding a novel solver for these problems.

The input to a MMAP problem is a Bayesian network $G$ where the nodes in the network are divided into 3 sets $E$, $D$, $S$ standing for evidence nodes, MAP (or decision) nodes and sum nodes, with a specification of values to be used for evidence nodes $E = e$. The goal is to find $\arg\max_{\pi} \sum_{V} p(V|\pi) \cdot \prod_{d \in D, s \in S} p(d)$. Anytime algorithms are typically scored using the log of marginal probability; the score for solution $\pi$ is $Q = \log \sum_{V} p(V|\pi) \cdot \prod_{d \in D, s \in S} p(d)$. Current state of the art exact algorithms use branch and bound techniques (e.g., [11]). Various approximation algorithms for MMAP exist including mixed product belief propagation [13], an extension of BP that directly addresses MMAP and is therefore closely related to the algorithms in this paper.

To make the connection more precise note that the evaluation problem in SOGBOFA is maximization over all possible actions, whereas for exact algorithms use branch and bound techniques (e.g., [11]). Various approximation algorithms for MMAP exist including mixed product belief propagation [13], an extension of BP that directly addresses MMAP and is therefore closely related to the algorithms in this paper.

The influence diagram problem is to find the setting for variables in $(D^{eq}, D^{out})$ which maximizes the expected utility $E[V[D^{eq}, D^{out}]] = p(D^{eq}, D^{out})$. An example of this construction with one evidence node $E$ and one MAP node $D$ is shown in Figure 1. We have:

**Proposition 5.** Let $G_1$ represent the original MMAP problem, $G_2$ the transformation into MEU, and let $E$ be the evidence for the MMAP problem and $D$ an assignment to the MAP variables. Then, $p_1(D = d, E = e) = E_{G_2}[V[D^{eq} = 0]]$.\footnote{Proof. (sketch) We illustrate how the claim can be proved for the example from Figure 1. In this case, $p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(E = e | A, D, D = d)$. Now in $G_2$, $E[V[D^{eq} = 0]] = p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(C(D^{eq} = 0) | A, D = d, C = 0) = \sum_{p(A)} p(D = d, A) p(C|D^{eq} = 0) | A, D = d, C = 0)!}$

Now replace the sum over $D^{eq} \in \{0, 1\}$ with a sum over the cases $D^{eq} = D^{eq}_0$ and $D^{eq} \neq D^{eq}_0$ and observe that $p(D^{eq}|D^{eq}_0, D^{eq}_0) = 1$ in the first case and 0 in the second. Therefore the last expression can be simplified to $p(D^{eq}|D^{eq}_0, D^{eq}_0) \sum_{p(A)} p(D = d, A) p(C|D^{eq} = 0) | A, D = d, C = 0)!$ which by construction is identical to the value for $G_1$.

The proof for the general case follows along the same steps. The crucial point is to replace the sum over $D^{eq}$ in the cases where it is the same vs. not equal to $D^{eq}_0$. This shows that the irrelevant terms cancel out and the remaining terms are identical to the original ones.\footnote{Proof. (sketch) We illustrate how the claim can be proved for the example from Figure 1. In this case, $p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(E = e | A, D, D = d)$. Now in $G_2$, $E[V[D^{eq} = 0]] = p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(C(D^{eq} = 0) | A, D = d, C = 0) = \sum_{p(A)} p(D = d, A) p(C|D^{eq} = 0) | A, D = d, C = 0)!}$
}

The reduction allows us to solve general MMAP problems using the SOGBOFA heuristics:

**AGS – Algebraic Gradient Based Solver for MMAP:**

1. Given a MMAP problem $G_1$ with evidence $E = e$, decision nodes $D$ and sum nodes $S$. Use the reduction to obtain a MEU problem $G_2$ with utility node $V$ and decision nodes $D^{eq}$.\footnote{Proof. (sketch) We illustrate how the claim can be proved for the example from Figure 1. In this case, $p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(E = e | A, D, D = d)$. Now in $G_2$, $E[V[D^{eq} = 0]] = p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(C(D^{eq} = 0) | A, D = d, C = 0) = \sum_{p(A)} p(D = d, A) p(C|D^{eq} = 0) | A, D = d, C = 0)!$}

2. Generate the SOGBOFA graph $G_{SOGBOFA}$ from the MEU problem where decision nodes are treated as action nodes and $V$ is the node of the planning problem.

3. Use the gradient based method in SOGBOFA (gradient ascent with random restarts) to optimize the marginal probabilities of variables $(D^{eq}_0)$.\footnote{Proof. (sketch) We illustrate how the claim can be proved for the example from Figure 1. In this case, $p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(E = e | A, D, D = d)$. Now in $G_2$, $E[V[D^{eq} = 0]] = p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(C(D^{eq} = 0) | A, D = d, C = 0) = \sum_{p(A)} p(D = d, A) p(C|D^{eq} = 0) | A, D = d, C = 0)!$}

4. Extract a discrete solution from the marginal probabilities by thresholding: $D^{eq} = 1$ if $p(D^{eq} > 0.5)$, otherwise $D^{eq} = 0$.\footnote{Proof. (sketch) We illustrate how the claim can be proved for the example from Figure 1. In this case, $p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(E = e | A, D, D = d)$. Now in $G_2$, $E[V[D^{eq} = 0]] = p(D = d, E = e) = \sum_{p(A)} p(D = d, A) p(C(D^{eq} = 0) | A, D = d, C = 0) = \sum_{p(A)} p(D = d, A) p(C|D^{eq} = 0) | A, D = d, C = 0)!$}

5 Experimental Validation

In this section, we explore the potential of AGS in solving complex MMAP problems. Specifically, we evaluate the anytime performance of AGS and two natural baselines. The first is the Mixed Product BP (MPBP) algorithm [13]. MPBP uses belief propagation and is therefore related to AGS, but in MPBP the search over MAP variables is integrated into the messages of BP and it is similarly derived from the corresponding optimization problem. The second algorithm is the recently developed Alternating best-first with depth-first AND/OR search (AAOBF) [14]. AAOBF interleaves best-first and depth-first search over an AND/OR search space to compute both anytime solutions (corresponding to all the outputs of $D^{eq}$) as well as upper bounds on the optimal MMAP value. AAOBF was shown to have excellent anytime performance and dominate other algorithms.

For the evaluation we use several problems from the UAI competition 2008. The original challenge problems were for sum inference, specifying the network and evidence variables. Following previous work, we use these for MMAP by selecting a subset of the variables as MAP nodes. To explore the performance of the algorithms we vary the proportion of MAP variables in each instance, and for each fixed ratio we generate 20 MMAP problems by picking the MAP nodes at random.
For AAOBF we use the implementation of [14] that can process the UAI competition problems directly. AGS requires CPTs as expressions and our implementation extracts such expressions from the tabular representation of the UAI problems as a preprocessing step. This is not computation-
ally demanding because the tabular representation is naturally restricted to have a small number of parents. We use our own implementation of MPBP, and for consistency the MPBP implementation benefits from the same expression representation of CPTs as AGS. More specifically, we use the join graph version of MPBP (algorithm 5 of [13]) and run it on the factor graph which is obtained from the BN. Since the factor graph is not a cluster tree we are running loopy MPBP. The max clusters of MPBP correspond to individual MAP variables, and sum nodes include both individual sum vari-
ables and factors in the original BN. Factor nodes and sum nodes perform the same computations as in standard loopy BP. The Max cluster with node \( i \) calculates a message to factor \( j \) as follows: first calculate the product of all incoming messages from factors other than \( j \). Then, noting that we have binary variables and thus only two entries in a message, zero out the smaller entry if \( t_i = 0 \) is strictly smaller.

MPBP keeps iterating over updates to nodes until it runs out of time or the maximal change of the messages becomes smaller than 0.0001. While [13] introduce annealing and restarts to improve the performance of this algorithm we do not use them here. Note that MPBP can get into a “contradiction state” when the graph has logical constraints, i.e., messages can become (0,0) or legal states are ruled out. AGS does not suffer from this problem. However, to enable the compar-
ison we modified the UAI instances changing any 0 probability to 0.0001 and (1 to 0.9999). The implementation of AAOBF replaces every 0 with 0.000001 for similar reasons. The solutions of all algorithms are evaluated off line using an exact solver which uses the same code base as AAOBF.

Figure 2 shows the results. Each algorithm is scored using log marginal probability. The plot shows a relative score \( \phi(t) = \frac{P(t)}{P_*} \) where \( \phi(t) \) is the score of algorithm \( a \) at time \( t \) and \( P_* \) is the best score found by any of the algorithms for this instance. This guarantees that relative scores are between 0 and 1, where the best value is 0. When an algorithm finds an inconsistent solution (probability 0) or does not find a solution we replace the score with 1. We show results for 3 problems, where for one problem (top row) we show results for a single run and for two problems we show results averaged over 20 runs. Comprehensive results with individual runs and aggregated runs on more instances are given in the supplementary material. The results for individual runs show more clearly transitions between no solution and the first solution for an algorithm whereas this is averaged in aggregate results. But the trends are consistent across these graphs. A first observation is that AAOBF has a larger initial overhead and AGS and MPBP are faster to find the first solutions.

AAOBF is significantly affected by the complexity of the conditional sum inference problems (i.e., evaluating the score of a specific MAP assignment). For the problems with 50% of MAP variables (and only 50% sum variables) the complexity is not too high and the search successfully finds high

quality solutions. For these problems AAOBF dominates both AGS and MPBP. On the other hand, with 70% and 80% of sum variables the summation problems are harder and AAOBF is slower to find solutions. In this case AGS dominates as it finds reasonable solutions fast and improves with time. To further illustrate the impact of summation difficulty we ran the algorithms in the same setup but with a fixed bound on run time varying the proportion of MAP variables from 0.1 to 0.9. Figure 3 shows results for the same 3 problems averaged over 20 runs, for run time of 1.5, 10 seconds in corresponding columns. Here, we clearly see the transition in relative performance of the algorithms as a function of the proportion of MAP variables. We also see that with shorter run time AGS dominates for a large range of problems. To summarise, given enough time AAOBF will find an optimal solution and can dominate AGS which is limited by the approximation inherent in BP. However, with a limited time and difficult conditional summation problems AGS provides a better tradeoff in finding solutions quickly.

6 Conclusions
The paper identifies a connection between a successful heuristic for planning in large factored spaces and belief propagation. The AGS/SOF heuristic performs its estimation symbolically and through that performs its search using gradients. This suggests a general scheme for approximate MINLP algorithms where the MAP value is represented using an explicit computation graph which is op-
timized directly through automatic differentiation. The instantiation of this scheme in AGS shows that it improves over the anytime performance of state of the art algorithms on problems with hard

summation sub-problems. In addition, while previous work has shown how inference can be used for

planning, this paper shows how ideas from planning can be used for inference. We believe that these connections can be further explored to yield improvements in both fields.

Acknowledgments
This work was partly supported by NSF under grant IIS-1616280. Some of the experiments in this paper were performed on the Tufts Linux Research Cluster supported by Tufts Technology Services.
References


Perception
Perception

Perception starts with sensing but also includes the additional facets of representation, interpretation and understanding. The introduction of deep learning transformed machine perception in areas like computer vision and speech recognition. Yet state-of-the-art machine perception systems primarily address identification and some level of interpretation. They are still far from truly understanding semantics – especially for images. In video, with the additional aspects of time and multi-modality introduced by audio, machine perception systems do not yet truly understand and make intelligent use of the data. We aim to advance research from analyzing static scenes and tagging images to dynamic scenes, which will eventually allow machine perception systems to understand videos or analyze the semantic aspects of human speech. The integration of audio-visual perception and the interaction between perception and language are major goals of our research.

This chapter covers papers related to AI and perception, including work that goes beyond object recognition in static images to video understanding and multi-modal recognition as shown in [37] and work on multiple object tracking in videos, facial expression recognition and action recognition in videos, combined with speech recognition and tone analysis in the audio channel [18]. Multi-modal input integrating images and natural language into conversational systems and questions answering is covered in [32] and [19]. Other works in this chapter include addressing scarcity of data and the large amount of computational resources required for AI in vision with few-shot learning [4], weak supervision [29] and reinforcement learning [21].

Selected Publications

17 — The Excitement of Sports: Automatic Highlights Using Audio/Visual Cues
p — 274

18 — Dialog-based Interactive Image Retrieval
p — 280

19 — A Prior-Less Method for Multi-Face Tracking in Unconstrained Videos
p — 298

20 — BlockDrop: Dynamic Inference Paths in Residual Networks
p — 319

21 — Revisiting Dilated Convolution: A Simple Approach for Weakly- and Semi-Supervised Semantic Segmentation
p — 322

"The eye sees only what the mind is prepared to comprehend."
Robertson Davies
The Excitement of Sports: Automatic Highlights Using Audio/Visual Cues

Michele Merler 1 Dhiraj Joshi 1 Khoi-Nguyen C. Mac 2 Quoc-Bao Nguyen 1 John Kent 1
Stephen Hammer 3 Jinjun Xiong 3 Minh N. Do 2 John R. Smith 1 Rogerio S. Feris 1

1 IBM T. J. Watson Research Center 2 University of Illinois at Urbana-Champaign 3 IBM iX

Abstract

The production of sports highlight packages summarizing a game’s most exciting moments is an essential task for broadcast media. Yet, it requires labor-intensive video editing. We propose a novel approach for auto-curating sports highlights, and demonstrate it to create a first of a kind, real-world system for the editorial aid of golf and tennis highlight reels. Our method fuses information from the players’ reactions (action recognition such as high-fives and fist pumps), players’ expressions (aggressive, tense, smiling and neutral), spectators (crowd cheering), commentator (tone of the voice and word analysis) and game analytics to determine the most interesting moments of a game.

1. Introduction

The tremendous growth of video data has resulted in a significant demand for tools that can accelerate and simplify the production of sports highlight packages for more effective browsing, searching, and content summarization. We present a novel approach for auto-curating sports highlights, showcasing its application to major golf and tennis tournaments (Masters, Wimbledon and US Open). Our approach combines information from the player, spectators, commentator and game analysis to determine a game’s most exciting moments. Video segments are then added to an interactive dashboard for quick review and retrieval by a video editor or broadcast producer, speeding up the process by which these highlights can then be shared with fans. Figure 1 shows the interface of our system, called High-Five (Highlights From Intelligent Video Engine), H5 in short.

The first prototype of IBM H5 [5, 3] was deployed at the 2017 Masters golf tournament extracting highlights live from multiple video streams over four days. Based on its success, H5 was further adapted to tennis content to be employed during the 2017 Wimbledon and US Open tennis tournaments. Personalized highlight extraction and retrieval is another unique feature of our system, using meta-data information about players extracted automatically from video graphics or provided by analysts and court-side statisticians.

Several methods have been proposed to automatically extract highlights from sports videos based on audio and visual cues, with approaches using the analysis of replays, crowd cheering, closed captioning and social media reactions [6, 8, 7]. More recently, Bettadapura et al. [2] used contextual cues from the environment to understand the excitement levels within a basketball game. Our proposed approach offers a unique combination of excitement measures to produce highlights, including information from the spectators, the commentator, and the player reaction.

2. Excitement Markers

2.1. Audio and Text based Markers Detection

Crowd cheering is perhaps the purest form of approval of a player’s shot within the context of any sport. Another important one is excitement in the commentators’ tone while describing a shot. Together those two audio cues play a key role in determining the position and excitement level of a potential highlight clip. We leverage SoundNet[1] to construct audio-based classifiers for crowd-cheering and commentator tone excitement. It uses a deep 1-D convolutional neural network architecture to learn representations of environmental sounds from nearly 2 million unlabeled videos. We learn linear SVM models atop the deep features to classify crowd- cheer/commentator tone excitement. The tone based commentator excitement measure is complemented by a text-based excitement marker which leverages a text-to-speech engine and uses a dictionary of 60 expressions indicative of excitement (e.g. “great shot”, “fantastic”).
2.2. Visual Marker Detection

Player Reaction is another important cue to determine interesting moments of a game. In our work, we train an action recognizer to detect a player celebrating. Inspired by [4], we use still images which are much easier to annotate and allow training with less computational resources compared to video-based classifiers. At test time, the classifier is applied at every frame and the scores aggregated for the highlight segment. Classifiers to detect player’s celebration are based upon the VGG-16 and the ResNet-50 architectures pretrained on Imagenet. Positive examples are sampled from 2016 Masters, Wimbledon, and US Open videos, and also from the web. Negative examples are sampled from the videos, as explained in Section 2.4.

Facial Expression carries valuable information that can augment or correct predictions from the player reaction models. For example, a tennis player might be raising his arm to collect a ball instead of celebrating a point. In this case, detecting a neutral facial expression can help reject a false positive instance. Training data to build a facial expression classifier was collected by extracting faces from the action celebration training images using a SSD detector. The extracted faces were then categorized into four types of expression: aggressive, tense, smiling, and neutral. The first three are associated with celebratory events, the last considered as non-celebratory. The classifier was trained by fine-tuning a VGG-face model on a dataset of tennis players faces.

2.3. Game Analytics

In tennis not every point has equal relevance within a game. For example match points and set points are more valuable than others, and business rules require them to be included in official highlights packages. During the tournaments we received live information about the points from side court statisticians and compiled it into a single analytics box in the following manner, which was devised following expert advice concerning the significance and difficulty of each item: (1) -0.1 for a point won due to unforced error or rally count smaller than 3, (2) +0.1 for a point won due to positive play, volley winner, smash winner, match point, break point, or rally count greater than 5, (3) +0.20 for a point won due to unforced error, negative reaction detected, or rally count greater than 10, (4) +0.25 for a game winning point. Positive play means a point won thanks to a player’s active effort, not an opponent’s mistake. The sum of values for any given point was then normalized in the range 0 to 1.

2.4. Cross Modality Bootstrapping

While the audio and visual classifiers were independent modules within the system, the training data gathering process proceeded through several rounds of bootstrapping, which exploited the correlation among modalities. In particular, we applied this principle for the visual player celebration classifier training. The underlying assumption is that the likelihood of finding frames showing players celebrating is higher in frames close to a detected crowd cheer. As shown in Figure 3 in each bootstrapping round, we fed unlabeled videos to the current classifiers, and sent to annotate the samples with highest positive or negative scores for each individual classifier, plus for the visual one the samples close in time with the highest scores of the cheer model. The newly labeled data was then used to finetune the classifiers for the next bootstrapping round, until a certain accuracy level on a held out validation set was reached.

3. Experiments

3.1. Experimental Setting

We evaluated our system in three real world championships, namely the 2017 Masters, Wimbledon, and US Open tournaments. For Masters, we analyzed in near real-time the content of four channels lasting simultaneously over the course of four consecutive days, for a total of 124 hours of content. The system ran on a RedHat Linux box with two K40 GPUs and produced 741 highlights in total. We extracted frames directly from the video stream at a rate of 1fps and audio in 6 seconds segments encoded as 16bit PCM at rate 22.050. The Wimbledon and US Open system ran on two Ubuntu nodes with four K80 GPUs each providing a total of 16 stream services to rank candidate highlight clips during the tournaments. Videos were chunked into 10 seconds segments and analyzed in less than 2.5 seconds each through our service APIs.

3.2. Individual Markers

All training for the individual markers was performed on content from the 2016 tournaments videos and images downloaded from the web, while testing was done on video data from the 2017 tournaments.

Player Celebration: We used Caffe for training VGG-16 and ResNet models with stochastic gradient descent. The data collected via multiple rounds of bootstrapping was augmented by random cropping and horizontal flipping. For Masters we used 2,908 positive examples and 6,744 negative ones, for Wimbledon 13,263 and 33,372, and for US Open 11,330 and 12,516. We evaluated on clips randomly selected from each of the 2017 tournaments and manually labeled. The number of frames, positive examples and negative examples for each tournament were 1,066, 59 and 1,005 for Masters, 4,777, 78 and 4,699 for Wimbledon, 8,963 and 52 and 9,911 for US Open, respectively. The imbalance of positive and negative examples reflects the actual distribution of data, since occurrences of a player celebrating are relatively rare within a match. Classification accuracies on 2017 Masters, Wimbledon, and US Open data were 98.4%, 98.12% and 90.33% respectively, while AUC were 0.9018, 0.9465, and 0.9313.

Facial expression: This marker was tested on faces extracted from frames in the 2017 tennis test set videos. Ex- pressions on the players faces were manually labeled as excited (combining aggressive, tense and smiling) or neutral. Classification accuracy was measured as AUC of 0.81 versus 0.75. Recognition accuracies of expressive against neutral expression were 82.42% and 82.23%, respectively.

Crowd Cheering Marker: Cheer samples from 2016 Masters and Wimbledon replicate videos as well as examples of cheer obtained from YouTube were used in order to train the audio cheer classifier using a linear SVM on top of deep features. For negative examples, we used audio tracks containing regular speech, music, and other non-cheer sounds found in Masters and Wimbledon replay videos. In total our final training set consisted of 453 positive and 454 negative samples (6 seconds each). We manually annotated random sets of six-second audio clips from the 2017 Masters (69 cheer, 336 non-cheer), Wimbledon (158, 915) and US Open (627, 937) tournaments videos to evaluate the performance of the model. The performance of the audio cheer model was measured as AUC of 0.9, 0.94 and 0.93 for 2017 Masters, US Open, and Wimbledon respectively.

3.3. Highlights Detection

Evaluating the quality of sports highlights is a challenging task, since a clearly defined ground truth does not exist. We approached this problem by comparing the clips automatically generated by our system to human judgments [2].

Human Rankings: We conducted user studies on Amazon Mechanical Turk in which workers were asked to evaluate the excitement level of several clips randomly sampled from the ones selected and scored by H5. We asked each participant to assign a score from 0 to 5 to a clip, with 0 meaning no interesting content and 5 being the most exciting shots. We then averaged the scores of the users for each clip. The resulting scores determined that 92.68% of the clips produced by our system were legitimate high-
lights (scores 2 and above), while 7.32% were mistakes. We then compared the rankings of the clips according to the scores of each individual marker, as well as their fusion, to the ranking obtained through the users votes. The performance of each ranking is computed at different depth $k$ with the normalized discounted cumulative gain (nDCG) metric (Figure 4). Naive-Fusion used equal weights to combine the normalized scores from each component, while Fusion (used by the live system) used weights optimized through cross-validation on a separate training set. For all tournaments the system’s Fusion outperforms the Naive one.

A/B Testing: Besides the ranking, for Tennis we also wanted to determine whether the selection made by the system about which clips should go into the compiled highlights aligned with human preferences. Thus we evaluated the clip selection process through another AMT experiment. For each tournament we randomly selected 500 pairs of clips. In each pair both clips belonged to the same game: one clip which had been selected to be part of the highlights, and one clip which had been discarded. We presented each pair to the workers and asked them to pick which clip in the pair was more exciting and/or interesting. Each pair was voted on by 15 workers, and a total of 234 unique users participated in the study. From Figure 5 (a) and (b) we can observe how for both tournaments the majority of voters picked the clips which were selected by the system to be part of the highlights of a game (blue curves) overwhelmingly over the non-highlight worthy ones (red curves). Naturally the fraction of clips on which a larger number of users agrees decreases as we move from 8 (the majority of voters) to 15 (all the voters), a trend clearly visible in the growth of the grey curves representing an indecision.

4. Conclusion
We presented a novel approach for automatically extracting highlights from sports videos based on multimodal sport-independent excitement measures, for which models were learned with reduced cost in training data annotation by exploiting the correlation of different modalities. We demonstrated the first-of-a-kind H5 system in three major golf and tennis tournaments in 2017, and showed that it agrees with human preferences.

References
Abstract

Inspired by the enormous growth of huge online media collections of many types (e.g., images, audio, video, e-books, etc.), and the paucity of intelligent retrieval systems, this paper introduces a novel approach to interactive visual content retrieval. The proposed retrieval framework is guided by free-form natural language feedback from users, allowing for more natural and effective communication. Such a system constitutes a multi-modal dialog protocol where in each dialog turn, a user submits a natural language request to a retrieval agent, which then attempts to retrieve the optimal object. We formulate the retrieval task as a reinforcement learning problem, and reward the dialog system for improving the rank of the target object during each dialog turn. This framework can be applied to a variety of visual media types (images, videos, graphics, etc.), and in this paper, we study in-depth its application on the task of interactive image retrieval. To avoid the cumbersome and costly process of collecting human-machine conversations as the dialog system learns, we train the dialog system with a user simulator, which itself is trained to describe the differences between target and retrieved images. The efficacy of our approach is demonstrated in a footwear image retrieval application. Extensive experiments on both simulated and real-world data show that: 1) our proposed learning framework achieves better accuracy than other supervised and reinforcement learning baselines; and 2) user feedback based on natural language rather than pre-specified attributes leads to more effective retrieval results, and a more natural and expressive communication interface.

1 Introduction

The volume of searchable visual media has grown tremendously in recent years, and has intensified the need for retrieval systems that can more effectively identify relevant information, with applications in domains such as e-commerce [1, 2], surveillance [3, 4], and internet search [5, 6]. Despite significant progress made with deep learning based methods [7, 8, 9], achieving high performance in such retrieval systems remains a challenge, due to the well-known semantic gap between feature representations and high-level semantic concepts, as well as the difficulty of fully understanding the user’s search intent. A typical approach to improve search efficacy is to allow the user a constrained set of possible interactions with the system [10, 11]. In particular, the user provides iterative feedback about retrieved objects, so that the system can refine the results, allowing the user and system to engage in a “conversation” to resolve what the user wants to retrieve. For example, as shown in Figure 1.

These two authors contributed equally to this work.
Figure 1: In the context of interactive image retrieval, the agent incorporates the user’s feedback to iteratively refine retrieval results. Unlike existing work which are based on relevance feedback or relative attribute feedback, our approach allows the user to provide feedback in natural language.

Relevance feedback about relevance [12] allows users to indicate which images are “similar” or “dissimilar” to the desired image, and relative attribute feedback [13] allows the comparison of the desired image with candidate images based on a fixed set of attributes. While these feedback paradigms are effective, the restrictions on the specific forms of user interaction largely constrain the information that a user can convey to benefit the retrieval process.

In this work, we propose a new approach to interactive visual content retrieval by introducing a novel form of user feedback based on natural language. This enables users to directly express, in natural language, the most prominent conceptual differences between the preferred search object and the already retrieved content, which permits a more natural human-computer interaction. We formulate the task as a reinforcement learning (RL) problem, where the system directly optimizes the rank of the target object, which is a non-differentiable objective.

We apply this RL based interactive retrieval framework to the task of image retrieval, which we call dialog-based interactive image retrieval. In particular, a novel end-to-end dialog manager architecture is proposed, which takes natural language responses as user input, and delivers retrieved images as output. To avoid the cumbersome, inefficient, and costly process of collecting and annotating machine as the system learns, we utilize a model-based RL approach by training a user simulator based on a corpus of human-written relative descriptions. Specifically, to emulate a single dialog turn, where the user provides feedback regarding a candidate image relative to what the user has in mind, the user simulator generates a relative caption describing the difference between the candidate image and the user’s desired image.1 Whereas there is a lot of prior work in image captioning [14, 15, 16], to our knowledge we are the first to explore the problem of relative image captioning, a general approach to more expressive and natural communication of relative preferences to machines, and to use it as part of a user simulator to train a dialog system.

The efficacy of our approach is evaluated in a real-world application scenario of interactive footwear retrieval. Experimental results with both real and simulated users show that the proposed reinforcement learning framework achieves better retrieval performance than existing techniques. Particularly, we observe that feedback based on natural language is more effective than feedback based on pre-defined relative attributes by a large margin. Furthermore, the proposed RL training framework of directly optimizing the rank of the target image shows promising results and outperforms the supervised learning approach which is based on the triplet loss objective. The main contributions of this work are as follows:

- A novel end-to-end deep dialog manager architecture, which addresses the above problem setting in the context of image retrieval. The network is trained based on an efficient policy optimization strategy, employing triplet loss and model-based policy improvement [17].
- The introduction of a novel computer vision task, relative image captioning, where the generated captions describe the salient visual differences between two images.
- The contribution of the first dataset, to our knowledge, which supports further research on the task of relative image captioning.

2 Related Work

Interactive Image Retrieval. Methods for improving image search results based on user interaction have been studied for more than 20 years [18, 11, 19]. Relevance feedback is perhaps the most popular approach, with user input specified either as binary feedback (“relevant” or “irrelevant”) [12] or based on multiple relevance levels [20]. More recently, relative attributes (e.g., “more formal than these,” “shinier than these”) have been exploited as a richer form of feedback for interactive image retrieval [13, 21, 22, 23, 24]. All these methods rely on a fixed, pre-defined set of attributes, whereas our method relies on feedback based on natural language, providing more flexible and more precise descriptions of the items to be searched. Further, our approach offers an end-to-end training mechanism which facilitates deployment of the system in other domains, without requiring the explicit effort of building a new vocabulary of attributes.

Image Retrieval with Natural Language Queries. Significant progress has been recently made on methods that lie in the intersection of computer vision and natural language processing, such as image captioning [15, 16], visual question-answering [25, 26], visual-semantic embeddings [27, 28], and grounding phrases in image regions [29, 30]. In particular, our work is related to image or video retrieval methods based on natural language queries [31, 32, 33, 34]. These methods, however, retrieve images and videos in a single turn, whereas our proposed approach aggregates history information from dialog-based feedback and iteratively provides more refined results.

Visual Dialog. Building conversational agents that can hold meaningful dialogs with humans has been a long-standing goal of Artificial Intelligence. Early systems were generally designed based on rule-based and slot-filling techniques [35, 36, 37], whereas modern approaches have focused on end-to-end training, leveraging the success of encoder-decoder architectures and sequence-to-sequence learning [38, 39, 40, 41]. Our work falls into the class of visually-grounded dialog systems [42, 43, 44, 45, 46]. Das et al [42] proposed the task of visual dialog, where the system has to answer questions about images based on a previous dialog history. De Vries et al. [43] introduced the GuessWhat! game [45] demonstrated emergence of grounded language and communication among visual dialog agents with no human supervision, using RL to train the agents in a goal-driven dialog setting. However, these dialogs are purely text-based for both the questioner and answerer agents, whereas we address the interactive image retrieval problem, with an agent presenting images to the user to seek feedback in natural language.

3 Method

Our framework, which we refer to as the dialog manager, considers a user interacting with a retrieval agent via iterative dialog turns. At the t-th dialog turn, the dialog manager presents a candidate image ao selected from a retrieval database D = {d1, ..., D} to the user. The user then provides a feedback sentence o1 describing the differences between the candidate image ao and the desired image. Based on the user feedback and the dialog history up to turn t, H = {o1, o2, ..., oT}, the dialog manager selects another candidate image ao+1 from the database and presents it to the user. This process continues until the desired image is selected or the maximum number of dialog turns is reached. In practice, the dialog manager could provide multiple images per turn to achieve better retrieval performance. In this work, we focus on a simplified scenario with a single image per interaction. We note that the same framework could be easily extended to the multiple-image case by allowing the user to select one image out of a list of candidate images to provide feedback on.
Figure 2: The proposed end-to-end framework for dialog-based interactive image retrieval.

3.1 Dialog Manager: Model Architecture

Our proposed dialog manager model consists of three main components: a Response Encoder, a State Tracker, and a Candidate Generator, as shown in Figure 2. At the t-th dialog turn, the Response Encoder embeds a candidate image and the corresponding user feedback, $(\alpha_t, o_t)$, into a joint visual-semantic representation $s_t \in \mathbb{R}^D$. The State Tracker then aggregates this representation with the dialog history from previous turns, producing a new feature vector $s_t \in \mathbb{R}^D$. The Candidate Generator uses the aggregated representation $s_t$ to select a new candidate image $o_{n+1}$ that is shown to the user. Below we provide details on the specific design of each of the three model components.

Response Encoder

The goal of the Response Encoder is to embed the information from the t-th dialog turn $(\alpha_t, o_t)$ into a visual-semantic representation $s_t \in \mathbb{R}^D$. First, the candidate image is encoded using a deep convolutional neural network (CNN) followed by a linear transformation: $x_t^v = \text{ImNet}(x_t) \in \mathbb{R}^D$. The CNN architecture in our implementation is an ImageNet pre-trained ResNet-101 [47] and its parameters are fixed. Words in the user feedback sentence are represented with one-hot vectors and then embedded with a linear projection followed by a CNN in an [48]: $x_t^w = \text{TextNet}(s) \in \mathbb{R}^D$. Finally, the image feature vector and the sentence representation are concatenated and embedded through a linear transformation to obtain the final response representation at time $t$: $z_t = W^{v} x_t^v \oplus x_t^w$, where $\oplus$ is the concatenation operator and $W \in \mathbb{R}^{D \times 2D}$ is the linear projection. The learnable parameters of the Response Encoder are denoted as $\theta$.

State Tracker

The State Tracker is based on a gated recurrent unit (GRU) [49], which receives as input the response representation $z_t$ as well as the history representation of previous dialog turns, and outputs the aggregated feature vector $\alpha_t$. The forward dynamics of the State Tracker is written as: $g_t = h_t = \text{GRU}(z_t, h_{t-1}), s_t = W^s g_t$, where $h_t \in \mathbb{R}^D$ and $g_t \in \mathbb{R}^D$ are the hidden state and the output of the GRU respectively, $h_t$ is the updated hidden state, $W^s \in \mathbb{R}^{D \times D}$ is a linear projection, and $s_t \in \mathbb{R}^D$ is the history representation updated with the information from the current dialog turn. The learnable parameters of the State Tracker (GRU model) are denoted as $\theta$. This memory-based design of the State Tracker allows our model to sequentially aggregate information from user feedback to localize the candidate image to be retrieved.

Candidate Generator

Given the feature representation of all images from the retrieval database, $(x_i^v, x_i^w)$, we can compute a sampling probability based on the distances between the history representation $s_t$ to each image feature, $x_i^v$. Specifically, the sampling probability $p_i$ is modeled using a softmax distribution over the top-$K$ nearest neighbors of $s_t$: $p_i = \frac{e^{-d(x_i^v, s_t)^2}}{\sum_{j=1}^{K} e^{-d(x_j^v, s_t)^2}}$, where $d$ is the L2 distance of $s_t$ to its K nearest neighbor in $(x_i^v)_{i=1}^N$. Given the sampling distribution, two approaches can be taken to sample the candidate image, denoted as $o_{n+1} = f'_\theta(s_t)$. (1) stochastic approach (used at training time), where $f' = \pi$, and (2) greedy approach (used at inference time), where $f' = \arg\max_{o} p_i(x_i)$. Combining the three components of the model architecture, the overall learnable parameters of the dialog manager are denoted as $\theta = \{\theta, \theta'_R\}$. Next, we explain how the network is trained end-to-end.

3.2 Training the Dialog Manager

Directly optimizing the ranking percentile metric in a supervised learning scheme is challenging since it is a non-differentiable function. Instead, we model the ranking percentile as the environment reward received by the agent and frame the learning process in a reinforcement learning setting. The reward $r_t$ of maximizing the expected sum of discounted rewards: $\max_{\theta} \mathbb{E} [r_t] = \sum_{t=1}^{T} \gamma^{t-1} r_t(\theta)$, where $\gamma \in [0, 1]$ is the discount factor determining the trade-off between short-term and long-term rewards, $T$ is the maximum number of dialog turns, and $r_t$ is the policy determined by network parameters $\theta$. Training the RL model for this problem requires extensive exploration of the action space, which is only feasible if a large amount of training data is available. However, collecting and annotating human-machine dialog data for our task is expensive. This problem is exacerbated in the situation of natural language based user feedback, which incurs an even larger exploration space as compared to approaches based on a fixed set of attributes. In test-based dialog systems, it is common to circumvent this issue by relying on user simulators [50]. We adopt a similar strategy, where a user simulator, trained on human-written relative descriptive sentences, substitutes the role of a real user in training the dialog manager (illustrated in Figure 3a). Below we further describe our user simulator, as well as the reinforcement learning techniques that we used to optimize our dialog manager.

3.2.1 User Simulator based on Relative Captioning

Here we propose the use of a relative simulator to simulate the user, which is a novel computer vision task in and of itself. It acts as a surrogate for real human users by automatically generating sentences that can describe the prominent visual differences between any pair of target and candidate images. We note that at each turn, our user simulator generates feedback independent of previous user feedback, and previously retrieved candidate images. While more sophisticated models that consider the dialog history could potentially be beneficial, training such systems well may require orders of magnitude more annotated data. In addition, back-referencing in dialog can inherently be ambiguous, inefficient, and complex to resolve, even for humans. Based on these considerations, we decided to first investigate the use of a single-turn simulator. While a few related tasks have been studied previously, such as content-aware image captioning [51] and referring expression generation [52], to the best of our knowledge, learning to compare and contrast visual differences and describe them in natural language is a novel computer vision task. Since there is no existing dataset supporting this task, we introduce a new dataset that consists of the dialog history could potentially be beneficial, training such systems well may require orders of magnitude more annotated data. In addition, back-referencing in dialog can inherently be ambiguous, inefficient, and complex to resolve, even for humans. Based on these considerations, we decided to first investigate the use of a single-turn simulator. While a few related tasks have been studied previously, such as content-aware image captioning [51] and referring expression generation [52], to the best of our knowledge, learning to compare and contrast visual differences and describe them in natural language is a novel computer vision task. Since there is no existing dataset supporting this task, we introduce a new dataset that consists of the dialog history.
the architecture of ResNet101 [47] pre-trained on ImageNet, and to better capture the localized visual differences, we added a visual attention mechanism; the loss function of the relative captioner is the sum of the negative log likelihood of the correct words [53].

3.2.2 Policy Learning

Supervised Pre-training When the network parameters are randomly initialized at the beginning, the history representations \( s \) are nearly random. To facilitate efficient exploration during RL training, we first pre-train the policy using a supervised learning objective, which is common in practice [54, 16]. While maximum likelihood-based pre-training is more common, here we pre-train using the more discriminative triplet loss objective:

\[
L^{\text{triplet}} = \mathbb{E} \left[ \sum_{i=0}^{n} \max(0, |s_i - s^*|_2 - |s_i - x^*_i|_2 + m) \right]
\]

where \( x^* \) and \( s^* \) are the image features of the target image and a random image sampled from the retrieval database respectively, and \( m \) is a constant for the margin. Intuitively, by ensuring the proximity of the target image and the images returned by the system, the rank of the target image can be improved without costly policy search from random initialization. However, entirely relying on this supervised learning objective deviates from our main learning objective, since the triplet loss objective does not jointly optimize the set of candidate images to maximize expected future reward. 2

Model-Based Policy Improvement Given the known dynamics of the environment (in our case, the user simulator), it is often advantageous to leverage its behavior for policy improvement. Here we adapt the policy improvement [17] to our model-based policy learning. Given the current policy \( \pi \) and the user simulator, the value of taking an action \( a \) using test-time configuration can be efficiently computed by look-ahead policy value estimation \( Q^\pi(s, a) = \mathbb{E}^{\text{test}} \left[ r_{t+1} + \gamma \cdot v^\pi(s_{t+1}) \right] \). Because our user simulator is essentially deterministic, one trajectory is sufficient to estimate an action value. Therefore, an improved policy \( \pi^* \) can be derived from the current policy \( \pi \) by selecting the best action given the value of the current policy, \( \pi^*(s) \equiv \arg \max_a Q^\pi(s, a) \). Specifically, we minimize the cross entropy loss given the derived action, \( \alpha^* \),

\[
L^{\text{ent}} = \mathbb{E} \left[ \sum_{t \leq T} \log \left( \pi^*(s_t) \right) \right]
\]

Compared to traditional policy gradient methods, the model-based policy improvement methods have much lower variance, and converge much faster. In Section 5, we further demonstrated the effectiveness of model-based policy improvement by comparing it with a recent policy gradient method. Figure 3b illustrates our policy learning method as described above.

4 Dataset: Relative Captioning

Our user simulator aims to capture the rich and flexible language describing visual differences of any given image pair. The relative captioning dataset thus needs this property. We situated the data collection procedure in a scenario of a shopping chatting session between a shopping assistant and a customer. The annotator was asked to take the role of the customer and provide a natural expression to inform the shopping assistant about the desired product item. To promote more regular, specific, and relative user feedback, we provided a sentence prefix for the annotator to complete when composing their response to a retrieved image. Otherwise the annotator response is completely free-form: no other constraints on the response were imposed. We used Amazon Mechanical Turk [55] to crowdsource the relative expressions. After manually removing erroneous annotations, we collected in total 10,721 captions, with one caption per pair of images. More details about the dataset collection procedure and the analysis on dataset statistics are included in Appendix A and Appendix B. To facilitate future research on relative captioning and reproducible results, we will make our dataset and codes available for public use.

5 Experimental Results

In Section 5.1, we assess the contribution of each component of our pipeline for policy learning. To evaluate the value of using free-form dialog feedback, we show experiments considering both simulated user feedback (Section 5.2) and real-world user feedback (Section 5.3).

All experiments were performed on the Shoe dataset [56], with the same training and testing data split for all retrieval methods and for training the user simulator. 10,000 database images were used during training, and 4,656 images for testing. The retrieval models are tested by retrieving images on the fly, starting from a randomly selected candidate image for the given dialog turn. Image retrieval performance is quantified by the average rank percentile of the image returned by the dialog manager on the test set. For details on architectural configurations, parameter settings, baseline implementation, please refer to Appendix D.

5.1 Analysis of the Learning Framework

We use our proposed user simulator to generate data and provide extensive quantitative analysis on the contribution of each model component.

Results on Relative Captioner. Figure 5 provides examples of simulator generated feedback and the collected user annotations. An interesting observation is that even though the user simulator only occasionally generates descriptions that exactly match the human annotations (the third example in Figure 5), it can still summarize the main visual differences between the images, since there is no other way to describe differences between two images. Qualitative examination of the generated relative expressions showed that the user simulator can approximate feedback of real users at a very low annotation cost (more analysis is included in Appendix C).

Policy Learning Results. To investigate how retrieval performance is affected by each component of the dialog manager, we compare our approach, denoted as Ours, against two variants: (1) SL, supervised learning where the agent is trained only with triplet loss; (2) RL-SCST, policy learning using Self-Critical Sequence Training (SCST) [16] after pre-training the network using the triplet loss objective. As shown in Figure 4 (solid lines), the average ranking percentile of the target image in all methods increases as the number of dialog turns increases. Both RL-based retrieval algorithms outperform the supervised pre-training, SL, which is expected since the triplet loss function does not directly optimize the retrieval ranking objective. Finally, Ours achieves 58.6% average ranking percentile with only two dialog turns and consistently outperforms RL-SCST across different dialog turns, which demonstrates the benefit of the model-based policy improvement component.
5.2 Effectiveness of Natural Language Feedback

In this section, we empirically evaluate the effect of natural language feedback, compared to predefined, relative attribute-based user feedback.

Generating Attribute Feedback. Each image in the dataset maps to a 10-D attribute vector, as described in [13]. We adopted a rule-based feedback generator which concatenates the respective attribute words with “more” or “less”, depending on the relative attribute values of a given image pair. For example, if the “shiny” value of the candidate image and the target image are 0.9 and 1.0 respectively, then the rule-based feedback is “more shiny.” Attributes are randomly sampled, similar to the relative attribute feedback generation in [13]. To simulate the scenario when users provide feedback using multiple attributes, individual attribute phrases are combined using “and”.

Results. We trained the dialog manager using both dialog-based feedback and attribute-based feedback (KL-attribute), where the subscript number denotes the number of attributes used in the rule-based feedback generator. The empirical result is summarized in Figure 4, including relative attribute feedback using one, three and five attribute phrases. Across different numbers of dialog turns, the natural language based agent produced significantly higher target image average ranking percentile than the attribute based methods. The results suggest that feedback based on unstructured natural language is more effective for retrieval than the predefined set of relative attributes used in [13]. This is expected as the vocabulary of relative attributes in [13] is limited. In fact, free-form dialog feedback obviates constructing a reliable and comprehensive attribute taxonomy, which in itself is a non-trivial task [57].

5.3 User Study of Dialog-based Image Retrieval

In this section, we demonstrate the practical use of our system with real users. We compare with an existing method, WhiiteSearch [13], on the task of interactive footwear retrieval. WhiteSearch represents images as feature vectors in a pre-defined 10-D attribute space, and interactively refines retrieval by iteratively updating the user’s relative feedback on attributes to narrow down the search space of the target image. For each method, we collected 50 five-turn dialogues; at each turn, one image is presented to the user to seek feedback. For WhiteSearch, the user can choose to use any amount of attributes to provide relative feedback on each during interaction. The resulting average ranking percentile of the dialog manager and WhiteSearch are 89.9% and 70.3% respectively. In addition to improved retrieval accuracy, users also reported that providing dialog-based feedback is more natural compared to selecting the most relevant attributes from a pre-defined list.

Figure 6 shows examples of retrieval dialogues from real users (please refer to Appendix E for more results and discussions). We note that users often start the dialog with a coarse description of the main visual features (color, category) of the target. As the dialog progresses, users give more specific feedback on fine-grained visual differences. The benefit of free-form dialog can be seen from the flexible usage of rich attribute words (“leopard print on shoes”), as well as relative phrases (“thinner”, “higher heel”). Overall, these results show that the proposed framework for the dialog manager exhibits promising behavior on generating to real-world applications.

6 Conclusions

This paper introduced two novel and practical tasks residing at the intersection of computer vision and language understanding: relative captivation, and dialog-based interactive image retrieval. Ultimately, techniques that are successful on such tasks will form the basis for the high fidelity, multi-modal, intelligent conversational systems of the future, and thus represent important milestones in this quest. We demonstrated the value of these tasks and the proposed learning architecture on the application of interactive fashion footwear retrieval. Our approach, enabling users to provide natural language feedback, significantly outperforms traditional methods relying on a pre-defined vocabulary of relative attributes, while offering more natural communication. As future work, we plan to leverage side information, such as textual descriptions associated with images of product items, and to develop user models that are conditioned on dialog histories, enabling more realistic interactions. We are also optimistic that our approach for image retrieval can be extended to other media types such as audio, video, and e-books, given the performance of deep learning on tasks such as speech recognition, machine translation, and activity recognition.

Acknowledgement We would like to give special thanks to Professor Kristen Grauman for helpful discussions.

References

10


Appendix: Dialog-based Interactive Image Retrieval

A Data Collection

In the following, we explain the details on how we collected the relative captioning dataset for training the user simulator and provide insights on the dataset properties. Unlike existing datasets which aim to capture the visual differences purely using “more” or “less” relations on visual attributes [13], we want to collect data which captures comparative visual differences that are hard to describe merely using a pre-defined set of attributes. As shown in Figure 8, we designed the data collection interface in the context of fashion footwear retrieval, where a conversational shopping assistant interacts with a customer and whose goal is to efficiently retrieve and present the product that matches the user’s mental image of the desired item.

Collecting Relative Expressions. The desired annotation for relative captioning should be free-form and introduce minimum constraints on how a user might construct the feedback sentence. On the other hand, we want the collected feedback to be concise and relevant for retrieval and avoid casual and non-informative phrases (such as “thand you”, “OK, well”). Bearing the two goals in mind, we designed a data collection interface as shown in Figure 8, which provided the beginning phrase of the user’s response (“Unlike the provided...”) and the annotators only needed to complete the sentence by giving an informative relative expression. This way, we can achieve a balance between sufficient lexical flexibility and avoiding irrelevant and casual phrases. After manual data cleaning, we are left with 30,751 relative expressions with one annotation per image pair.

Augmenting Dataset with Single-Image Captions. During our data collection procedure for relative expressions, we observed that when the target image and the reference image are visually distinct (fourth example in Figure 7(b)), users often only implicitly use the reference image by directly describing the visual appearance of the target image. Inspired by this, we asked annotators to give direct descriptions on 3940 images without the use of reference images. We then paired each image in this set with multiple visually distinct reference images (selected using deep feature similarity). This data augmentation procedure further boosted the size of our dataset at a relatively low annotation cost.

B Dataset analysis

Figure 7(a) shows the length distribution of the collected captions. Most captions are very concise (between 4 to 8 words), yet composing a large body of highly rich vocabularies as shown in Figure 9. Interestingly, although annotators have the freedom to give feedback in terms of comparison on a single visual attribute (such as “is darker”, “is more formal”), most feedback expressions consist of compositions of multiple phrases that often include spatial or structural details (Table 1).

1A few high-frequency words are removed from this chart, including “has/have”, “is/are”, “a”, “with”.

Figure 7: Length distribution of the relative captioning dataset (a), and examples of relative captions collected in the dataset (b). The leading phrase “Unlike the provided image, the ones I want” is omitted for brevity.
Section 4: Perception

Dialog-based Interactive Image Retrieval

Table 1: Examples of relative expressions. Around two thirds of the collected expressions contain composite feedback on more than one type of visual feature, and 40% of the expressions contain propositional phrases that provide information containing spatial or structural details.

<table>
<thead>
<tr>
<th>Single Phrase</th>
<th>Composition of Phrases</th>
<th>Propositional Phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>are brownish</td>
<td>is more athletic and is white</td>
<td>is lower on the ankle and blue</td>
</tr>
<tr>
<td>have a zebra print</td>
<td>has a larger sole and to not a high heel</td>
<td>have rhinestones across the toe and a strap</td>
</tr>
<tr>
<td>have a thick foot shaft</td>
<td>has better feel and exposes more foot and toe</td>
<td>are brown with a sole cut out</td>
</tr>
<tr>
<td>are low-top canvas sneakers</td>
<td>is white, and has high heels, not flats</td>
<td>is in neutrals with buckled strap and platforms</td>
</tr>
<tr>
<td>have polka dot linings</td>
<td>are adjustible, not stake prints, and is a pointed toe</td>
<td>is more rugged with textured sole</td>
</tr>
</tbody>
</table>

Figure 10: Ratings of relative captions provided by humans and different relative captioner models. The ratings were asked to give a score from 1 to 4 on the quality of the captions: no errors (4), minor errors (3), somewhat related (2) and unrelated (1).

Examples of the collected relative expressions are shown in Figure 7b. We observed that, in some cases, users apply a concise phrase to describe the key visual difference (first example); but most often, users adopt more complicated phrases (second and third examples). The benefit of using free-form feedback can be seen in the second example: when the two shoes are exactly the same on most attributes (white color, flat heeled, clog shoes), the user resorts to using composition of a fine-grained visual attribute (“toes”) with spatial reference (“on the top”). Without free-form dialog based feedback, this intricate visual difference would be hard to convey.

C Human Evaluation of Relative Captioning Results

We tested a variety of relative captioning models based on different choices of feature fusion and the use of attention mechanism. Specifically, we tested our Show and Tell [15] based model, RC-FC (using concatenated deep features as input), and three Show, Attend and Tell [53] based models, including RC-FC (feature concatenation), RC-LNA (feature fusion using a linear layer) and RC-CNA (feature fusion using a convolutional layer). For all methods, we adopted the architecture of ResNet101 [47] pre-trained on ImageNet to extract deep feature representation.

Given the intrinsic flexibility in describing visual differences between two images, and the lack of comprehensive variations of human annotations for each pair of images, common image captioning metrics (such as BLEU) cannot provide reliable evaluation of the actual quality of the generated captions. Therefore, to better evaluate the caption quality, we directly conducted human evaluation,
following the same rating scheme used in [15]. We collected user ratings on captions generated by each model and those provided by humans on 1000 image pairs. The results are summarized in Figure 10. All relative captioning models produced similar performance with RC-CNA exhibiting marginally better performance. It is also noticeable that there is a gap between human provided descriptions and all automatically generated captions, and we observed some captions with incorrect attribute descriptions or are not entirely sensible to humans, as shown in Figure 11. This indicates the inherent complexity of task of relative image captioning and room for improvement of the user simulator, which will lead to more robust and generalizable dialog agents.

D Experimental Configurations

Since no official training and testing data split was reported on Shoes dataset, we randomly selected 10,000 images as the training set, and the rest 4,606 images as the held-out testing set. The user simulator adopts the same training and testing data split as our dialog manager: it was trained using image pairs sampled from the training set with no overlap with the testing images. Since the four models for relative image captioning produced similar qualitative results in the user study, we selected RC-FCA model as our user simulator since it leads to more efficient training time for the dialog manager than the RC-CNA model. The baseline method, RL-SCST, uses the same network architecture and the same supervised pre-training step as our dialog manager and also utilizes the user simulator for training. The idea of RL-SCST is to use test-time inference reward as the baseline for policy gradient learning by encouraging policies performing above the baseline while suppressing policies under-performing the baseline. Given the trained user simulator, we can easily compute the test-time rewards for RL-SCST by greedy decoding rather than stochastically sampling the image to return at each dialog turn.

For all methods, the embedding dimensionality of the feature space is set to $D = 356$; the MLP layer of the image encoder is fine-tuned using the simple image captions to better capture the domain-specific image features. For SL training, we used the ADAM optimizer with an initial learning rate of $0.001$ and the margin parameter $m$ is set to 0.3. For all reinforcement learning based methods, we employed the RMSprop optimizer with an initial learning rate of $10^{-5}$, and the discount factor is set to 1. For our dialog manager, we set the number of nearest neighbors as 5 for the Candidate Generator.

E Discussions on the Dialog Manager

In this section, we provide more discussions on the proposed dialog manager framework and point out a few directions for improvement.

Dialog-based User Interaction. Figure 12 shows more examples of the dialog interactions on human users. In all three examples, the target image reached a final ranking within the top 100 images (about 97%) in ranking percentile) in the fifth dialog turn. These examples indicate that, visible improvement of retrieval results often comes from a flexible combination of direct reference to distinctive visual attributes of the target image, and comparison to the candidate image based on relative attributes. Ideally, feedback based on a pre-defined attribute set can achieve similar performance if the attribute vocabulary is sufficiently comprehensive and descriptive (which often consists of hundreds of words as in our footware retrieval application). But in practice, it is infeasible to ask the user to scroll through a list of hundreds of attribute words and select the optimal one to provide feedback on.

Further, we observe that the system tends to be less responsive to certain low-frequency words generated by the use simulator (such as “shoelace” in the third example). This is as expected, since the dialog manager is trained on the user simulator, which in itself has limitations (such as the fixed size of vocabulary after being trained, and the lack of memory for dialog history). We are interested in fine-tuning the dialog manager on real users, so that it can directly adapt to new vocabularies from the user. In summary, results on real users demonstrated that few-form dialog feedback is able to capture various types of visual differences with great lexical flexibility and can potentially result in valuable applications in real-world image retrieval systems.

Dialog Manager Learning Framework. One main advantage of the proposed RL-based framework is to train the agent end-to-end with a non-differentiable objective function (the target image rank). While triplet loss based objective makes it efficient to pre-train the dialog manager, it still deviates from the ranking objective. As illustrated in Figure 13: two examples exhibit similar triplet loss objectives, but the target image ranks differ greatly.

We noticed that the dialog manager based on the current learning architecture sometimes forgets information from past turns. For example, in the second example of Figure 12, the second turn imposes a “yellow accents” requirement to the target image. While this feedback is reflected in the immediate next turn, it is missing from the later turns of the dialog. We think that model is able to better incorporate the dialog history is able to alleviate this issue. We could in principle investigate more variations of the network design to further improve its performance. Overall, the proposed network architecture is effective in demonstrating the applicability of dialog-based interactive image retrieval.
A Prior-Less Method for Multi-Face Tracking in Unconstrained Videos

Chung-Ching Lin
IBM Research AI
cclin@us.ibm.com

Ying Hung
Rutgers University
yhung@stat.rutgers.edu

Abstract

This paper presents a prior-less method for tracking and clustering an unknown number of human faces and maintaining their individual identities in unconstrained videos. The key challenge is to accurately track faces with partial occlusion and drastic appearance changes in multiple shots resulting from significant variations of makeup, facial expression, head pose and illumination. To address this challenge, we propose a new multi-face tracking and re-identification algorithm, which provides high accuracy in face association in the entire video with automatic cluster number generation, and is robust to outliers. We develop a co-occurrence model of multiple body parts to seamlessly create face tracklets, and recursively link tracklets to construct a graph for extracting clusters. A Gaussian Process model is introduced to compensate the deep feature insufficiency, and is further used to refine the linking results. The advantages of the proposed algorithm are demonstrated using a variety of challenging music videos and newly introduced body-worn camera videos. The proposed method obtains significant improvements over the state of the art [51], while relying less on handling video-specific prior information to achieve high performance.

1. Introduction

The task of Multiple Object Tracking (MOT) or Multiple Target Tracking (MTT) is to recover the trajectories of a varying number of individual targets while the status of targets is estimated at different time steps. Multi-face tracking is one of the important domains enabling high-level video content analysis and understanding, e.g., crowd analysis, semantic analysis, and event detection. In this paper, our goal is to track an unknown number of human faces and maintain their identities in unconstrained videos (e.g., movies, TV series, music videos [51], body-worn camera videos). Our method does not assume any extra prior knowledge about the videos or require manual efforts (e.g., input underlying number of clusters in videos). Despite having different methods proposed to address this topic, this problem remains challenging due to the inherent unconstrained settings in videos. The videos might contain multiple shots captured by one or multiple moving cameras, irregular camera motion and object movement, arbitrary camera setting and object appearance, and people may move in-and-out camera field of view multiple times. The appearance of faces change drastically owing to significant variations of lighting condition, camera angle, expression, and head pose. Commonly, partial occlusions are caused by accessories and other body parts, such as glasses and hair, as well as hand gestures.

This is a difficult task and has a different focus from tracking in constrained videos (e.g., surveillance videos captured by steady or slowly-moving cameras), where the main challenge is to deal with different viewpoints, lighting conditions, and crowded pedestrian crossings. Many methods have been proposed [1, 3, 30, 41, 47, 49, 54]. In those papers, three popular datasets, MOT Challenge [29], PETS [16] and KITTI [17], are usually used to evaluate the performance of MOT methods. The videos, however, do not include multiple shot changes and appearance changes. These MOT methods attempt to solve different challenges and cannot be easily applied to unconstrained videos with large camera movement or multiple abrupt shot changes.

Due to the fast-growing popularity of unconstrained videos, especially on the Web, solutions to this problem are in high demand and have attracted great interest from researchers. The recently proposed methods [22, 23] enable users to track persons in unconstrained videos. These methods focus on the tracking accuracy within each shot,
but the scope does not include persons association across shots. They assign a new ID when a person reappears in the videos.

Figure 1 gives some sample frames from datasets used in this paper. We test our algorithm on two different types of challenging unconstrained video datasets. The first dataset, provided by [51], contains eight edited music videos with significant variation in expression, scale, pose, expression, and illumination in multiple shots. The second dataset is newly introduced in this paper. It includes four highly challenging unedited videos captured by people using body-worn cameras. The videos depict complex events but have limited quality control, and therefore, include severe illumination changes, camera motion, poor lighting, and heavy occlusion. In both datasets, persons are in-and-out of camera fields of view multiple times, and the proposed method is designed to track the faces across shots while maintaining the assigned identities, as shown in Figure 1.

Our framework incorporates three major components: achieve high accuracy face tracking, and it is robust to substantial face rotations, from frontal to profile. First, we develop a co-occurrence model of multiple body parts to create longer face tracklets. We then develop a recursive algorithm to link tracklets with strong associations. Finally, a Gaussian process model is designed to refine the algorithm to link tracklets with strong associations. The recursively linked tracklets help associate persons in tracking tracklets, but they do not directly applicable to videos with significant variations in facial appearance.

**CNN-based representation learning**. Many areas have gained performance improvement from advances in deep learning. Several CNN-based models for face recognition provide biometrics-solutions: VGG-Face [37], DeepFace [42], and FaceNet [40]. The datasets that are used to train these CNN models are generally chosen from good conditions, e.g., high image resolution, frontal faces, rectified faces, and full faces. However, in an unconstrained video, a face can be profiled, cropped, or blurry. In these cases, measuring the similarity with extracted deep face features might yield inferior performance.

**State-of-the-art**. Recently, Zhang et al. [51] propose a CNN-based approach of learning video-specific facial discriminative features for multi-face tracking and demonstrate state-of-the-art performance. The main limitation of this method is that it has difficulty in handling videos where many shots contain only one single person. In these cases, the method cannot generate sufficient negative face pairs to train the network, thus different persons might be incorrectly identified as the same person across shots. Additionally, the method requires prior knowledge of the videos to provide actual number of clusters in advance. In reality, the correct and optimal choice of cluster numbers is often ambiguous in application to some videos of minor characters. If cluster numbers are ill-initialized, clustering purity would degrade. Further, an essential prerequisite of this method is to apply an effective shot change detection technique to partition each input video into non-overlapping shots.

In contrast, we propose an algorithm to analyze raw video data and generate final clustering and tracking results automatically in a data-driven fashion. The proposed method seeks to eliminate the sensitivities of handling video-specific prior information.

**3. Algorithm**

To achieve better tracking results, longer tracklets of each person are desired. The longer the tracklets are, the greater the number of possible facial variations of each person could be captured. However, longer tracklets usually contain more noise, and thus might incur more tracking linking errors. Considering the pros and cons, we propose a framework, as illustrated in Figure 2, that includes three core algorithmic components: (1) Create tracklets. We develop a co-occurrence model of multiple body parts to create longer face tracklets. A face is temporally missing when a person turns his/her head, or the view of their face is blocked by another object (e.g., hand, or others’ head). The model is designed to prevent tracks from being terminated when an image of a face temporarily disappears (Section 3.1). (2) Link tracklets. We recursively link tracklets with strong associations. The recursively linked tracklets construct a constrained graph for extracting clusters, and generating initial clustering results (Section 3.2). (3) Detect and reassign outlier tracklets. We design a Gaussian Process model to capture the richness of data and compensate the deep feature insufficiency. Our model will detect and re-assign outlier tracklets (Section 3.3).

### 3.1. Tracking by Co-occurrence Model

Typically, the performance of detectors is greatly affected by pose, occlusion, rotation, size and image resolution. For example, when a person turns his/her head away from the camera or the face is occluded, a face might not be detected. However, the head belonging to that person could be still detected and tracked. We build an idea that using multiple body parts simultaneously could create longer tracklets. To this end, we developed a co-occurrence model which obtains information of multiple body parts to help continue the tracker during moments when faces are not captured by the camera or not detected by the detector, but the person remains in the video frames.

Our starting point is the multiple body parts detections estimated by off-the-shelf body-part detector [5]. Note that the detection method could be replaced by other sophisticated body-part detectors [6, 39]. For each video frame, we extract localization of face, head, torso, and whole body. We denote $v_t^{x,y}$ a discrete set of outputs of body-part detections in a frame $t$ where $v_t^{x,y} = \{x_1, y_1, h_1, \ldots, x_n, y_n, h_n\}$ is the index of the detection, $e$ is the index of the detection, $x_t, y_t, h_t$ are center, width and height of a bounding box, $y$ denotes the type of body part, such as $y = \{1, \ldots, N\}$ for each body part detection, two thresholds are applied [21]. (1) Detection results filtered by a high threshold are
used to create new tracklets, (2) detection results filtered by a low threshold is used to track objects. When no correspondence is used to track objects, we formulate the multi-person tracking problem as a graph structure $G = (V,E)$ with two types of edges, $e_i$ and $v_i$, as shown in Figure 2 (a). Spatial edges $e_i$ denote the connections of different body parts of a candidate within a frame. The spatial edges $e_i$ are used to generate hypothesized states of a candidate. Temporal edges $v_i$ denote the connections of the same body parts over adjacent frames. The state of each individual person in different frames are estimated using temporal edges. $e_i = \{(v_{i1}, v_{i2}), \beta_i \neq \gamma_i, v_i = \{(v_{i1}, v_{i2}), \beta_i = \gamma_i \}$.

3.2. Recursive Constrained Tracklet Linking

After face tracklets are generated, each face tracklet is taken as a node $T_i$, which includes various face poses of a person with extracted feature $f_i$ and frame indexes $\{f_i\}_{i=1}^{|T_i|}$. We aim to infer the underlying pairwise similarity between nodes to construct meaningful affinity graphs for face clustering. Specifically, we use the VGG-face descriptors [17] to extract features from the FAC7 layer. We design a unified and generalized linking framework based on how the VGG-face network was trained to avoid less discriminative features from the FC7 layer.

The spatial edges $\varepsilon_i$ are defined as:
$$\varepsilon_i = \{(v_{i1}, v_{i2}), \beta_i \neq \gamma_i\},$$
(1)

The spatial edges $\varepsilon_i$ are defined as:
$$\varepsilon_i = \{(v_{i1}, v_{i2}), \beta_i \neq \gamma_i\},$$
where $\phi(x_t, x_j)$ is the correlation function and $\theta$ is a vector of unknown correlation parameters. There are

$$\Lambda(T_i, T_j) = \{1 - D(T_i, T_j), if D(T_i, T_j) < \phi\}$$
$$0, otherwise,$$ (6)

where $D(T_i, T_j)$ is measured by the distance between VGG-face features $d(f_i, f_j)$. All linkages are built when $D(T_i, T_j)$ is smaller than a threshold $\phi$.

$$\Lambda(T_i, T_j) = \frac{1}{\max\{d(f_i, f_j), f_j\}}.$$ (7)

where $d(f_i, f_j)$ is the Euclidean distance between $f_i$ and $f_j$.

$\Lambda(T_i, T_j)$ enforces the resolution constraint and builds linkages among tracklets that have larger image size. We defined $\Lambda(T_i, T_j)$ as:

$$\Lambda(T_i, T_j) = \begin{cases} 1, & \text{if } T_i \cap T_j \neq \emptyset \\ 0, & \text{otherwise} \end{cases}$$ (8)

We apply k-means method to separate all tracklets based on the average image size of each tracklet and obtain group $G_k$, which consists of tracklets with larger image size. Another type of linkage is $L_{ij}$, which is built by the relative distances among coexisting tracklets. First, we search all sets of coexisting tracklets by the procedures described in Algorithm 1. For each tracklet $T_i$ in $T_i$, we search the corresponding nearest neighbor tracklet in $T_j$ and build a linkage between them using the similarity measurement $M(T_i, T_j)$, which takes into account for appearance affinity and relative distance constraints. Constraints $M(T_i, T_j)$ is the same as Equation 6. $\Lambda(T_i, T_j)$ is used to impose relative distance constraints. Because coexisting tracklets should be exclusive, the connection between a tracklet in $T_i$ and a tracklet in $T_j$ should be one or none. We use this property to prevent false connections. When the relative distance between two tracklets $T_i$ and $T_j$ is smaller than $\eta$, we do not want to build a linkage between them.

$$\Lambda(T_i, T_j) = \begin{cases} 1, & \text{if } \|T_i - T_j\| < \eta \\ 0, & \text{otherwise} \end{cases}$$ (9)

We formulate the multi-person tracking problem as a graph structure $G = (V,E)$ with two types of edges, $e_i$ and $v_i$, as shown in Figure 2 (a). Spatial edges $e_i$ denote the connections of different body parts of a candidate within a frame. The spatial edges $e_i$ are used to generate hypothesized states of a candidate. Temporal edges $v_i$ denote the connections of the same body parts over adjacent frames. The state of each individual person in different frames are estimated using temporal edges. $e_i = \{(v_{i1}, v_{i2}), \beta_i \neq \gamma_i, v_i = \{(v_{i1}, v_{i2}), \beta_i = \gamma_i \}$.

3.3. Dimension Reduction Using GP

Gaussian process (GP) models, also known as kriging, are commonly used in many applications including machine learning and geostatistics [11]. Different from CNN-based approaches, GPs provide a flexible parametric approach to capture the nonlinearity and spatial-temporal correlation of the underlying system. Therefore, it is an attractive tool to be combined with the CNN-based approach to further reduce the dimension without losing complex, and important spatial-temporal information. Here, we illustrate the idea of reducing the dimension by fitting a GP model for each color channel with the spatial information. Three GP models are constructed obtained and the dimension is reduced to 18 parameters captured by the GP models. Note, the reduction ratio is empirically determined and range from 18 and may be flexibly determined by changing the number of parameters in the GP models.

A Gaussian process model can be written as $p(y|x) = N(x; \mu(x), \kappa(x, x'))$, where $y \in \mathbb{R}$ is the intensity of a color and $x \in \mathbb{R}^d$ is the input. In this research, $x$ represents the spatial information, $y$ represents the color value. The mean function $\mu(x)$ is modeled as $\mu(x) = a^T \beta + b^T \beta_3 + x \theta$ as a function of $w$ with unknown parameters $\beta$, say, $\mu(x) = a^T \beta + b^T \beta_3 + x \theta$. In addition, $\ell(x, y)$ is a Gaussian process with mean $0$ and Cov$\{x, x'\} = \sigma^2(x, x')$, where $\phi(x, x')$ is the correlation function and $\theta$ is a vector of unknown correlation parameters.

3.3.1 Dimension Reduction Using GP

Gaussian process (GP) models, also known as kriging, are commonly used in many applications including machine learning and geostatistics [11]. Different from CNN-based approaches, GPs provide a flexible parametric approach to capture the nonlinearity and spatial-temporal correlation of the underlying system. Therefore, it is an attractive tool to be combined with the CNN-based approach to further reduce the dimension without losing complex, and important spatial-temporal information. Here, we illustrate the idea of reducing the dimension by fitting a GP model for each color channel with the spatial information. Three GP models are constructed obtained and the dimension is reduced to 18 parameters captured by the GP models. Note, the reduction ratio is empirically determined and range from 18 and may be flexibly determined by changing the number of parameters in the GP models.

A Gaussian process model can be written as $p(y|x) = N(x; \mu(x), \kappa(x, x'))$, where $y \in \mathbb{R}$ is the intensity of a color and $x \in \mathbb{R}^d$ is the input. In this research, $x$ represents the spatial information, $y$ represents the color value. The mean function $\mu(x)$ is modeled as $\mu(x) = a^T \beta + b^T \beta_3 + x \theta$ as a function of $w$ with unknown parameters $\beta$, say, $\mu(x) = a^T \beta + b^T \beta_3 + x \theta$. In addition, $\ell(x, y)$ is a Gaussian process with mean $0$ and Cov$\{x, x'\} = \sigma^2(x, x')$, where $\phi(x, x')$ is the correlation function and $\theta$ is a vector of unknown correlation parameters.

3.3.2 Finding Maximum LCs

Global maximum $\max\{\ell(T^*_i, T^*_j)\}$ given $\sigma^2$ is calculated by fitting $\ell(x, y)$ model for each channel to the spatial data. In addition, the reduction ratio is empirically determined and range from 18 and may be flexibly determined by changing the number of parameters in the GP models.

A Gaussian process model can be written as $p(y|x) = N(x; \mu(x), \kappa(x, x'))$, where $y \in \mathbb{R}$ is the intensity of a color and $x \in \mathbb{R}^d$ is the input. In this research, $x$ represents the spatial information, $y$ represents the color value. The mean function $\mu(x)$ is modeled as $\mu(x) = a^T \beta + b^T \beta_3 + x \theta$ as a function of $w$ with unknown parameters $\beta$, say, $\mu(x) = a^T \beta + b^T \beta_3 + x \theta$. In addition, $\ell(x, y)$ is a Gaussian process with mean $0$ and Cov$\{x, x'\} = \sigma^2(x, x')$, where $\phi(x, x')$ is the correlation function and $\theta$ is a vector of unknown correlation parameters.
3.3.2 Outlier Detection and Reassignment by GP

We introduce the outlier detection and reassignment scheme in this section. Our idea is to measure how isolated a tracklet is when compared to the spatial surrounding neighborhood. More precisely, by comparing the local density of a tracklet to the local densities of its neighbors, we can identify tracklets that have a substantially lower density than their neighbors, as shown in Figure 2 (c2). These exceptionally low-density tracklets are likely to belong to other clusters. We detect these outlier tracklets and reconnect them to one of the clusters by extracting GP features \( r \in R^d \). For each cluster, we use a simple unsupervised outlier detection method, Local Outlier Factor (LOF) estimator [4], to compute the local density deviation of a given tracklet with respect to its neighbors. After detecting outliers, we refine the original clusters by disconnecting the linkages among outliers and other tracklets as there might be incorrect association among those linkages.

We further use extracted GP features to link all isolated tracklets to the refined clusters. We evaluate the appearance similarity between the isolated tracklet and every refined cluster. For any given isolated tracklet, we evaluate all pairwise distances between the isolated tracklet and every tracklet in one cluster and use the shortest distance as the similarity measure between the isolated tracklet and each cluster. We also enforce a temporal constraint to prevent multiple tracklets with overlapping frame indexes in the same cluster. Next, we determine the cluster that has the shortest distance to the given isolated tracklet and assign the tracklet into that cluster.

After all tracklets have been connected into one of the clusters, we obtain final clusters. Finally, we assign a specific identity to each cluster and generate final tracking.

4. Experiments

We empirically demonstrate the effectiveness of our proposed method on two distinct types of challenging unconstrained video datasets and compare with state-of-the-art methods, especially with variants in [51].

4.1. Details

**Dataset:** Experiments are conducted on two datasets: (1) Edited High-quality Music Video Dataset. The dataset is introduced by [51] contains 8 edited music videos. The videos contain dramatic facial appearance changes, frequent camera view and shot changes, and rapid camera motion. (2) Unedited Body-worn Camera Video Dataset. We introduce a new highly-challenging dataset of 4 realistic and unedited body-worn camera videos. All videos were captured by police officers in different incidents, and thus have limited quality control. The videos in this dataset have very severe camera movement and heavy occlusion. There is a large number of dark scenes and many tracks with non-frontal faces.

**Experiment settings:** All parameters have the same settings and ranges in the following experiments. The high detection threshold for creating new track is \( 0.8 \), low detection threshold for tracking is \( 0.1 \). \( n \) is 0.5; \( c \) is 0.9; \( \theta \) is 0.7; \( \alpha \) is 2. Note that, in contrast to [51], our method does not apply shot change detection and does not assume that the total cluster number is known a priori.

**Evaluation metrics:** (1) Clustering. We use Weighted Clustering Purity (WCP) [51] to evaluate the extent to which faces can be clustered automatically according to their identities. WCP is given as \( WCP = \frac{1}{|C|} \sum_{c \in C} P(c,\hat{c}) \), where \( P(c,\hat{c}) \) is the number of total faces in the video, \( n_c \) is the number of faces in the cluster \( c \in C \), and its purity, \( P(c) \) is measured as the fraction of the largest number of faces from the same person to \( n_c \), and \( C \) is the total number of clusters. (2) Tracking. We report tracking results based on the most widely accepted evaluation metrics, the CLEAR MOT [36] (including Recall, Precision, F1, MAP, MT, IDS, MOTA, and MOTA).

### Table 1: Clustering purity comparisons with the state-of-the-art methods on music videos. The best results are highlighted with the bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Precision</th>
<th>F1</th>
<th>MT</th>
<th>MOTA</th>
<th>MOTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triplet[51]</td>
<td>0.69</td>
<td>0.78</td>
<td>0.73</td>
<td>0.82</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>Pre-trained[51]</td>
<td>0.68</td>
<td>0.75</td>
<td>0.72</td>
<td>0.74</td>
<td>0.80</td>
<td>0.82</td>
</tr>
<tr>
<td>IHTLS[13]</td>
<td>0.68</td>
<td>0.64</td>
<td>0.66</td>
<td>0.73</td>
<td>0.80</td>
<td>0.82</td>
</tr>
<tr>
<td>VGG-face[37]</td>
<td>0.23</td>
<td>0.46</td>
<td>0.35</td>
<td>0.27</td>
<td>0.27</td>
<td>0.29</td>
</tr>
<tr>
<td>AlexNet[27]</td>
<td>0.25</td>
<td>0.31</td>
<td>0.29</td>
<td>0.18</td>
<td>0.22</td>
<td>0.24</td>
</tr>
<tr>
<td>HOG[12]</td>
<td>0.25</td>
<td>0.40</td>
<td>0.33</td>
<td>0.18</td>
<td>0.21</td>
<td>0.23</td>
</tr>
<tr>
<td>ADMM[2]</td>
<td>75.5</td>
<td>61.8</td>
<td>68.0</td>
<td>0.50</td>
<td>23</td>
<td>2982</td>
</tr>
<tr>
<td>Triplet[51]</td>
<td>71.8</td>
<td>88.8</td>
<td>79.4</td>
<td>0.20</td>
<td>19</td>
<td>902</td>
</tr>
<tr>
<td>Pre-trained[51]</td>
<td>68.6</td>
<td>81.8</td>
<td>76.5</td>
<td>0.48</td>
<td>0.48</td>
<td>0.56</td>
</tr>
<tr>
<td>IHTLS[13]</td>
<td>69.3</td>
<td>82.4</td>
<td>77.2</td>
<td>0.48</td>
<td>0.48</td>
<td>0.56</td>
</tr>
<tr>
<td>VGG-face[37]</td>
<td>61.3</td>
<td>74.5</td>
<td>67.8</td>
<td>0.35</td>
<td>0.35</td>
<td>0.43</td>
</tr>
<tr>
<td>AlexNet[27]</td>
<td>60.7</td>
<td>72.6</td>
<td>66.6</td>
<td>0.32</td>
<td>0.32</td>
<td>0.40</td>
</tr>
<tr>
<td>Ours</td>
<td>85.7</td>
<td>94.5</td>
<td>90.2</td>
<td>0.85</td>
<td>27</td>
<td>3208</td>
</tr>
</tbody>
</table>

### Table 2: Quantitative comparisons with the state-of-the-art tracking methods on music video dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Precision</th>
<th>F1</th>
<th>MT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triplet[51]</td>
<td>0.69</td>
<td>0.78</td>
<td>0.73</td>
<td>0.82</td>
</tr>
<tr>
<td>Pre-trained[51]</td>
<td>0.68</td>
<td>0.75</td>
<td>0.72</td>
<td>0.74</td>
</tr>
<tr>
<td>IHTLS[13]</td>
<td>0.68</td>
<td>0.64</td>
<td>0.66</td>
<td>0.73</td>
</tr>
<tr>
<td>VGG-face[37]</td>
<td>0.23</td>
<td>0.46</td>
<td>0.35</td>
<td>0.27</td>
</tr>
<tr>
<td>AlexNet[27]</td>
<td>0.25</td>
<td>0.31</td>
<td>0.29</td>
<td>0.18</td>
</tr>
<tr>
<td>HOG[12]</td>
<td>0.25</td>
<td>0.40</td>
<td>0.33</td>
<td>0.18</td>
</tr>
<tr>
<td>ADMM[2]</td>
<td>75.5</td>
<td>61.8</td>
<td>68.0</td>
<td>0.50</td>
</tr>
<tr>
<td>Triplet[51]</td>
<td>71.8</td>
<td>88.8</td>
<td>79.4</td>
<td>0.20</td>
</tr>
<tr>
<td>Pre-trained[51]</td>
<td>68.6</td>
<td>81.8</td>
<td>76.5</td>
<td>0.48</td>
</tr>
<tr>
<td>IHTLS[13]</td>
<td>69.3</td>
<td>82.4</td>
<td>77.2</td>
<td>0.48</td>
</tr>
<tr>
<td>VGG-face[37]</td>
<td>61.3</td>
<td>74.5</td>
<td>67.8</td>
<td>0.35</td>
</tr>
<tr>
<td>AlexNet[27]</td>
<td>60.7</td>
<td>72.6</td>
<td>66.6</td>
<td>0.32</td>
</tr>
<tr>
<td>Ours</td>
<td>85.7</td>
<td>94.5</td>
<td>90.2</td>
<td>0.85</td>
</tr>
</tbody>
</table>

These results of our algorithm. In some frames, we can see by other cast members. As shown, the proposed algorithm is capable of generating invariant face identities and tracking them reliably across different shots in the entire unconstrained video.

**Speed:** We have measured execution speed of the proposed method on music videos that typically have several faces to be tracked in each frame. In one 5-minutes music video, there are 21,747 face observations over a sequence of 5,000 frames, our implementation takes about 25 minutes after feeding the detection results. The running time is implemented with unoptimized C++ and Matlab code, simple thread execution on a Mac with Intel 2.5 GHz 7 CPU and 16 GB memory.

4.3. Experiments on Unedited Realistic Body-worn Camera Dataset

To further test the capability of our method, we conduct experiments on unedited realistic body-worn camera dataset and compare the results with variants in [51].

**Clustering:** We compare the clustering results with HOG[12], AlexNet[27], VGG-face[37], pre-trained, Siamese and SymTriplet in [51]. Table 3 shows our method outperforms other methods with noticeable margin on all videos in the body-worn camera dataset. This problem is particularly challenging. For example, in Westlife, our method is capable of generating invariant face identities and tracking them reliably across different shots in the entire unconstrained video.

**Qualitative Results:** Figure 3 shows sample tracking results of our algorithm. In some frames, we can see that different persons have very similar face appearance, multiple main singers appear in a cluttered background filled with audiences, or some faces have heavy occlusions by other cast members. As shown, the proposed algorithm is capable of generating invariant face identities and tracking them reliably across different shots in the entire unconstrained video.
methods cannot cope with such low resolution. In addition, SymTplet [51] requires sufficient negative pairs generated from trackers that co-occur in the same shot. But in body-worn camera videos, many shots contain only a single person. Consequently, they are unable to train their network and fine-tune features well. However, these problems are addressed by our proposed method. We believe the significant performance difference lies in our ability to fine-tune the networks or manual video analysis to obtain their correct identities in the shaking and low resolution unconstrained videos. More qualitative results are available in the supplementary material.

5. Conclusions
We have introduced a prior-less algorithm for reliably tracking multiple faces in unconstrained videos, where extensive motion and variations exist and affect the way by which many heretofore existing methods perform. Experiments on two distinct video datasets demonstrated the superiority of the proposed method when compared to the state-of-the-art methods that require intensive training to fine-tune the networks or manual video analysis to obtain the number of clusters. In the future, we intend to explore modeling the similarity of other body parts to extend our framework’s capability.

Acknowledgement
Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00341. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government.

References


Abstract

Very deep convolutional neural networks offer excellent recognition results, yet their computational expense limits their impact for many real-world applications. We introduce BlockDrop, an approach that learns to dynamically choose which layers of a deep network to execute during inference so as to best reduce total computation without degrading prediction accuracy. Exploiting the robustness of Residual Networks (ResNets) to layer dropping, our framework selects on-the-fly which residual blocks to evaluate for a given novel image. In particular, given a pretrained ResNet, we train a policy network in an associative reinforcement learning setting for the dual reward of utilizing a minimal number of blocks while preserving recognition accuracy. We conduct extensive experiments on CIFAR and ImageNet. The results provide strong quantitative and qualitative evidence that these learned policies not only accelerate inference but also encode meaningful visual information. Built upon a ResNet-101 model, our method achieves a speedup of 20% on average, going as high as 36% for some images, while maintaining the same 76.4% top-1 accuracy on ImageNet.

1. Introduction

Deep neural networks are now ubiquitous in computer vision owing to their recent successes in several important tasks. However, great strides in accuracy have been accompanied by increasingly complex and deep network architectures. This presents a problem for domains where fast inference is essential, particularly in delay-sensitive and real-time scenarios such as autonomous driving, robotic navigation, or user-interactive applications on mobile devices.

Most existing work pursues model compression techniques to speed up a deep network [19, 4, 25, 36, 32, 16, 54, 31]. While significant speed-ups are possible, the approach yields a one-size-fits-all network that requires the same fixed set of features to be extracted for all novel images, no matter their complexity. In contrast, an important feature of the human perception system is its ability to adaptively allocate time and scrutiny for visual recognition [49]. For example, a single glimpse is sufficient to recognize some objects and scenes, whereas more time and attention is required to clearly understand occluded or complicated ones [52].

In this spirit, we explore the problem of dynamically allocating computation across a deep network. In particular, we consider Residual Networks (ResNets) [18] both due to their strong track record for recognition tasks [18, 8, 17] as well as their tolerance to removal of layers [50]. ResNets are composed of residual blocks, consisting of two or more convolutional layers and skip-connections, which enable direct paths between any two residual blocks. These skip-connections make ResNets behave like ensembles of relatively shallow networks, and hence the removal of a certain residual block generally has only a modest impact on performance [50]. However, the preliminary study of block dropping in ResNets [50] applies a global, manually defined dropping scheme (the same blocks for all images), which leads to increased errors when more blocks are dropped.

We propose to learn optimal block dropping strategies...
that simultaneously preserve both prediction accuracy and minimal block usage based on image-specific decisions. When a novel input is presented to the network trained for recognition, a dynamic inference path is followed, selectively choosing which blocks to compute for that instance. See Figure 1. The approach not only improves computational efficiency during inference (i.e., for a similar prediction accuracy, being able to drop more residual blocks than a static global scheme), but also facilitates further insights into ResNets, e.g., whether different blocks encode information about objects, whether the computation needed to classify depends on the difficulty level of the example.

This work introduces BlockDrop, a reinforcement learning approach to derive instance-specific inference paths in ResNets. The main idea is to learn a model (referred to as the policy network) that, given a novel input image, outputs the posterior probabilities of all the binary decisions for dropping or keeping each block in a pretrained ResNet. The policy network is trained using curriculum learning to maximize a reward that incentivizes the use of as few blocks as possible while preserving the prediction accuracy. In addition, the pretrained ResNet is further jointly finetuned with the policy network to produce feature transformations tailored for block dropping behavior. Our approach can be seen as an instantiation of associative reinforcement learning [46] where all the decisions are taken in a single step given the input (i.e., the input instance); this makes policy execution lightweight and scalable to very deep networks. We conduct extensive experiments on CIFAR [27] and ImageNet [10]. BlockDrop achieves 93.6% and 73.7% accuracy using just 33% and 55% of blocks in a pretrained ResNet-101 and Cifar-100, respectively, outperforming state-of-the-art methods [14, 15, 12, 32] by clear margins. Furthermore, BlockDrop speeds up a ResNet-101 model on ImageNet by 20% while maintaining the same 76.4% top-1 accuracy. Qualitatively, we observe that the dropping policies learned with BlockDrop are correlated with the visual patterns in the images, e.g., within the “orange” class, images containing a pile of oranges take an inference path that is different from that taken by the “closed orange” images. Furthermore, BlockDrop policies for easy images with clearly visible objects utilize fewer residual blocks compared to the difficult images that contain other occluding or background objects. Note that although our analysis in this paper is focused on ResNets, the proposed approach could also be applied to other recently proposed ResNet variants such as ResNeXt [55] or Multi-Residual Networks [1], as well as other tasks beyond image classification.

2. Related Work

Layer Dropping in Residual Networks. Dropping layers in residual networks has been used as a regularization mechanism, similar to Dropout [44] or DropConnect [53], for training very deep networks (e.g., over 1000 layers) with stochastic depth [22]. Unlike our method, residual layer dropping in stochastic depth networks happens only during the training stage, but at test time the layers remain fixed. Veit et al. [50] show that ResNets are resilient to layer dropping at test time, which motivates our approach; however, they do not provide a way to dynamically choose which layers could be removed from a network without sacrificing accuracy. More recently, Huang and Wang [23] propose a method for selecting a subset of residual blocks to be executed based on a sparsity constraint. In contrast to these approaches, we propose an instance-specific residual block removal scheme to speed up ResNets during inference.

Model Compression. The need to deploy top-performing deep neural network models on mobile devices motivates techniques that can effectively reduce the storage and computational costs of such networks, including knowledge distillation [19, 40, 4], low-rank factorization [25, 47, 41], filter pruning [30, 36, 32, 57], quantization [16, 54, 31], compression with structured matrices [6, 43], network binarization [38, 7, 33], and hashing [5]. Efficient network architectures such as SqueezeNet [24] and MobileNetv2 [31] have also been explored for training compact deep nets. In contrast to this line of work where the same amount of computation is applied to all images, we focus on efficient inference by dynamically choosing a subset of blocks to be executed conditioned on the input image. More importantly, our method is complementary to these model compression techniques: the residual blocks that are kept for execution can be further pruned for even greater speedup.

Conditioned Computational Several conditioned computation methods have been proposed to dynamically execute different modules of a network on a per-example basis [1, 2]. Sparse activations in combination with gating functions are usually adopted to selectively turn on and off a subset of modules based on the input. These gating functions can be learned with reinforcement learning [2, 34, 31]. These models typically associate a reward with a series of decisions computed after each layer/patient, the resulting policy execution overhead makes it expensive to scale them up to very deep models with hundreds or thousands of layers. In contrast, our policy network makes all routing decisions in a single step resulting in lower overhead cost for the routing itself and thus larger computational savings. Reinforcement learning has also been applied for dynamic feature prioritization in images [26] and videos [45, 56], actively deciding which frames or image regions to visit next. These techniques could be used in tandem with our approach.

Figure 2: Illustration of our proposed framework. Given a new image, the policy network outputs dropping and keeping decisions for each block in a pretrained ResNet, which then makes a prediction by evaluating the active blocks only. Policy rewards account for both block usage and prediction accuracy. The policy network is trained to optimize the expected reward with a curriculum learning strategy, and then jointly finetuned with the ResNet.

Early Prediction. Our work relates more strongly to early prediction models, a class of conditional computation models that exist once a criterion (e.g., sufficient confidence for classification) is satisfied at early layers. Cascade detectors [13, 51] are among the earliest methods that exploit this idea in computer vision, often relying on handcrafted control decisions learned separately from visual features. More recently, joint learning of features and early decisions has been studied for deep neural networks. Terenzi et al. [48] propose Branchynet, a network composed of branches at each layer to make early classification decisions. Similarly, Adaptive Computation Time (ACT) [15] augments an RNN with a halting unit whose activation determines the probability that computation should continue. Figurnov et al. [4] further extend this idea to the special domain in ResNets by applying ACT to each spatial position of multiple image blocks. Like our work, their formulation identifies instance-specific ResNet configurations, but it only allows configurations that use early, contiguous blocks in each predefined segment of the ResNet. These early blocks usually encode low-level features in high-dimensional feature maps, and may lack the discriminative power required for the task. This issue can be mitigated by using images at different scales [55, 21], but at a higher computational cost. Instead, we allow any block to contribute to our network, allowing for a much higher variability in potential ResNet configurations and policies.

3. Approach

Given a test image, our goal is to find the best configuration of computational blocks in a pretrained ResNet model, such that a minimum number of blocks is used, without incurring a decrease in classification accuracy. Treating the task of finding this configuration as a search problem quickly becomes intractable for deeper models as the number of potential configurations grows exponentially with the number of blocks. Learning a soft-attention mask over the blocks also presents problems, namely the difficulty of converting this mask into binary decisions which would require carefully handcrafted thresholds. In addition, such a thresholding operation is non-differentiable, making it non-trivial to directly adopt a supervised learning framework. We therefore leverage policy search methods from reinforcement learning to derive the optimal block dropping schemes that encourage correct predictions with minimal block usage. To this end, we first revisit the architecture of ResNet in Sec. 3.1, and discuss why it is a good fit for block dropping. Then we introduce our policy network in Sec. 3.2, which learns to dynamically select inference paths conditioned on the input image. Finally, we present the training algorithm of our model in Sec. 3.3.

3.1. Pretrained Residual Networks

ResNets consist of multiple stacked residual blocks which are essentially regular convolutional layers that are bypassed by identity skip-connections. If we denote the input to the i-th residual block as $y_i$, and the function represented by its residual block as $F_i$, the output of this residual block is given by $y_{i+1} = F_i(y_i) + y_i$, which is directly fed as the input to the next residual block. The presence of identity skip-connections induces direct paths between any two residual blocks, and hence top layers in the network are able to access information from bottom layers during a forward pass while gradients can be directly passed from higher layers to lower layers in the back-propagation phase. Veit et al. [50] demonstrated that removing (or dropping) a residual block at test time (i.e., having
yi+1 = yi, does not lead to a significant accuracy drop. This behavior is due to the fact that ResNets can be viewed as an ensemble of many paths—as opposed to single-path models like AlexNet [28] and VGGNet [42]—and so information can be preserved even with the deletion of paths.

The results in [50] suggest that different blocks do not share strong dependencies. However, the study also shows classification errors do increase when more blocks are removed from the model during inference. We contend this is the result of their adopting a global dropping strategy for all images. We posit the best dropping schemes, which lead to correct predictions with the minimal number of blocks, must be block-specific.

3.2. Policy Network for Dynamic Inference Paths

The configurations in the context of ResNets represent decisions to keep/drop each block, where each decision to drop a block corresponds to removing a subset of paths from the network. We therefore formulate these decisions as our dropping strategy. To derive the optimal dropping strategy given an input instance, we develop a policy network to output a binary policy vector, representing the actions to keep or drop a block in a pretrained ResNet. During training, a reward is given considering both block usage and prediction accuracy, which is generated by running the ResNet with only active blocks in the policy vector. See Figure 2 for an overview.

Unlike standard reinforcement learning, we train the policy to predict all actions at once. This is essentially a single-step Markov Decision Process (MDP) given the input state and can also be viewed as contextual bandit [29] or associative reinforcement learning [46]. We examine the positive impact of this design choice on scalability in Sec. 4.2.

Formally, given an image x and a pretrained ResNet with K residual blocks, we define a policy of block dropping as a K-dimensional Bernoulli distribution:

\[
\mathbf{w}(\mathbf{x}) = \prod_{k=1}^{K} \left( 1 - s_k \right)^{-w_k}(1 - s_k)
\]

where \(f_{\pi}(\mathbf{x}, \mathbf{w})\) denotes the policy network parameterized by weights \(\mathbf{W}\) and \(\mathbf{s}\) is the output of the network after the \(\pi(\mathbf{x}, \mathbf{w})\) function. We choose the architecture of \(f_{\pi}\) (details below in Sec. 4) such that the cost of running it is negligible compared to ResNet, i.e., so that policy execution overhead remains low. The \(k\)-th entry of the vector, \(s_k \in \{0,1\}\), represents the likelihood of its corresponding residual block of the original ResNet being dropped. An action \(\mathbf{u} \in \{0,1\}^K\) is selected based on \(\mathbf{s}\). Here, \(\mathbf{w} = -\infty\) and \(\mathbf{u} = -\infty\) indicate dropping and keeping the \(k\)-th residual block respectively.

Only the blocks that are not dropped according to \(\mathbf{u}\) will be evaluated in the forward pass. To encourage both correct predictions as well as minimal block usage, we associate the actions taken with the following reward function:

\[
R(\mathbf{u}) = \begin{cases} 1 - \frac{(\mathbf{u})^2}{2} & \text{if correct} \\ \gamma & \text{otherwise} \end{cases}
\]

where \(\gamma \rightarrow 0\) and \(\mathbf{u} = 0\) is the maximally probable configuration under the current policy, i.e., \(u_k = 1\) if \(s_k > 0.5\), and \(u_k = 0\) otherwise [19].

We further encourage exploration by introducing a parameter \(\alpha\) to bound the distribution and prevent it from saturating, by creating a modified distribution \(s' = \alpha \cdot u + (1 - \alpha) \cdot (1 - u)\). This bounds the distribution in the range \(1 - \alpha \leq s' \leq \alpha\), from which we then sample the policy vector.

Curriculum learning. Policy gradient methods are typically extremely sensitive to their initialization. Initially, we found that starting from a randomly initialized policy and optimizing for both accuracy and block usage is not effective, due to the extremely large dimension of the search space, which scales exponentially with the total number of blocks (there are \(2^K\) possible orderings of the blocks).

Note that in contrast with applications such as image captioning where ground-truth action sequences (captions) can be used to train an initial policy [39], here such “expert examples” are available, other than the standard single execution path that executes all blocks.

3.3. Training the BlockDrop Policy

Expected gradient. To maximize Eqn. 4, we utilize policy gradient [60], one of the seminal policy search methods [57], to compute the gradients of \(J\). In contrast to typical reinforcement learning methods where policies are sampled from a multimodal distribution [46], our policies are generated from a K-dimensional Bernoulli distribution. With \(u_k \in \{0,1\}\), the gradients can be derived similarly as:

\[
\nabla_{\mathbf{W}} J = \mathbb{E}_{\mathbf{R}(\mathbf{u})} \nabla_{\mathbf{W}} \log \mathbb{E}_{\mathbf{R}(\mathbf{u})} \left[ \sum_{k=1}^{K} \log s_k u_k + (1 - s_k)(1 - u_k) \right]
\]

where again \(\mathbf{w}\) denotes the parameters of the policy network. We approximate the expected gradient in Eqn. 5 with Monte-Carlo sampling using all samples in a mini-batch. These gradient estimators are unbiased, but exhibit high variance [66]. To reduce variance, we utilize a self-critical baseline line \(R(\mathbf{u})\) as in [99], and rewrite Eqn. 5 as:

\[
\nabla_{\mathbf{W}} J = \mathbb{E}_{\mathbf{R}(\mathbf{u})} \sum_{k=1}^{K} \log [s_k u_k + (1 - s_k)(1 - u_k)]
\]

where \(\mathbf{A} = \mathbf{R}(\mathbf{u}) - \mathbf{R}(\mathbf{0})\) and this is the maximally probable configuration under the current policy, i.e., \(u_k = 1\) if \(s_k > 0.5\), and \(u_k = 0\) otherwise [19].

We further encourage exploration by introducing a parameter \(\alpha\) to bound the distribution and prevent it from saturating, by creating a modified distribution \(s' = \alpha \cdot u + (1 - \alpha) \cdot (1 - u)\). This bounds the distribution in the range \(1 - \alpha \leq s' \leq \alpha\), from which we then sample the policy vector.
Table 1: Accuracy and block usage with our policies vs heuristic baselines, with and without jointly finetuning (ft) for all methods. For fair comparisons, K is selected based on the average block usage of our method, and this can be different before and after finetuning. Note that the average value of K for our method is reported here for brevity. It is determined dynamically per image, and can be as low as 3 (out of 54) in ResNet-110 on CIFAR-10.

<table>
<thead>
<tr>
<th></th>
<th>CIFAR-10</th>
<th>CIFAR-100</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc</td>
<td>K</td>
</tr>
<tr>
<td>FirstK</td>
<td>18.6</td>
<td>10</td>
</tr>
<tr>
<td>RandomK</td>
<td>20.5</td>
<td>10</td>
</tr>
<tr>
<td>DistributeK</td>
<td>23.4</td>
<td>10</td>
</tr>
<tr>
<td>Ours</td>
<td>26.9</td>
<td>9.4</td>
</tr>
</tbody>
</table>

For ResNet-32 and ResNet-110 respectively, outperforming the baselines by a large margin. Furthermore, the instance-specific nature of our method allows us to capture the inherent variance in the computational requirements of our dataset. We notice a wide distribution in block usage depending on the image. With ResNet-110, nearly 15% of the images use fewer than 10 blocks, with some images using as few as 3 blocks. This variance cannot be captured by any static policies. Similar trends are observed on CIFAR-100. This confirms that dropping residual blocks with policies computed in a learned manner is indeed significantly better than heuristic dropout behaviors. The fact that RandomK performs better than FirstK is interesting, suggesting the value of having residual blocks at different segments to learn feature representations at different scales.

4.2. Quantitative Results

Learned policies vs. heuristics. We compare our block dropping strategy to the following alternative methods: (1) FirstK, which keeps only the first K residual blocks active; (2) RandomK, which keeps K randomly selected residual blocks active; (3) DistributeK, which evenly distributes K blocks across all segments. For all baselines, we choose K to match the average number of blocks used by DropBlock, rounding up as needed. DistributeK allows us to see if feature combinations of different blocks learned by DropBlock are better than features learned from the restricted set of early blocks of each segment. This setting resembles the allowable feature combinations from early stopping models applied to ResNets.

The results in Table 1 highlight the advantage of our instance-specific policy. On CIFAR-10, the learned policies give an accuracy of 88.6% and 75.4% using an average of 9.4 and 20.1 blocks from the original ResNet-32 and ResNet-110 respectively, outperforming the baselines by a large margin. Furthermore, the instance-specific nature of our method allows us to capture the inherent variance in the computational requirements of our dataset. We notice a wide distribution in block usage depending on the image. With ResNet-110, nearly 15% of the images use fewer than 10 blocks, with some images using as few as 3 blocks. This variance cannot be captured by any static policies. Similar trends are observed on CIFAR-100. This confirms that dropping residual blocks with policies computed in a learned manner is indeed significantly better than heuristic dropout behaviors. The fact that RandomK performs better than FirstK is interesting, suggesting the value of having residual blocks at different segments to learn feature representations at different scales.

Impact of joint finetuning. Next we analyze the impact of joint finetuning (cf. Sec. 3.3) for both our approach and the baselines, denoted P in Table 1. Joint finetuning further significantly improves classification accuracy using fewer (or almost the same) number of blocks. In particular, on CIFAR-10, it offers absolute performance gains of 18.6% using 2.5 and 3.2 fewer blocks with ResNet-32 and ResNet-110 respectively compared with curriculum training alone. Similarly, on CIFAR-100, joint finetuning improves accuracy and brings down block usage with ResNet-110. For ResNet-32, we observe 0.7 more blocks on average are used after finetuning, which might be due to the challenging nature of CIFAR-100 requiring more blocks to make correct predictions. Comparing ResNet-101 with ResNet-32, we observe that the computational speed-ups are more dramatic for deeper ResNets owing to the fact that there are more blocks with potentially diverse features to select from. When built upon ResNet-101, our method outperforms the pretrained model by 0.4% and 1.5% (absolute) using 51% and 55.9% of the original blocks on CIFAR-10 and CIFAR-100, respectively. Additionally, we observe that some images use as few as 5 blocks for inference. These results confirm that joint finetuning can indeed assist the ResNet to adapt to the removal of blocks by refining its features and cost, while maintaining its capacity for instance-specific variation.

BlockDrop vs. state-of-the-art methods. We next compare BlockDrop to several techniques from the literature. We vary γ, which controls our algorithm’s trade-off between block usage and accuracy, to get a range of models with varying computational requirements. We compute the average FLOPs utilized to classify each image in the test set; FLOPs are a hardware independent metric, allowing for fair comparisons across models.

We compare to the following state-of-the-art methods: (1) ACT and (2) SACT \cite{zhao2018production}, (3) PFEC \cite{molchanov2016pruning}, (4) LCCL \cite{wu2018lcc}, ACT and SACT learn a halting score at the end of each block, and exit the model when a high-confidence is obtained. PFEC and LCCL reduce the parameters of convolutional layers by either pruning or sparsity constraints, which is complementary to our method. Other model compression methods cited earlier do not report results on larger ResNet models, and hence are not available to compare here.

Figure 3 (a) presents the results on CIFAR. We observe that our best model offers 0.4% performance gain in accuracy (93.6% vs. 93.2%) using 65% fewer FLOPs on average (1.73 × 10^8 vs. 5.08 × 10^8) over the original ResNet-110 model. The performance gains might result from the regularization effect of dropping blocks when finetuning the network as in \cite{zhao2018production}. Compared to ACT and SACT, our method only requires 50% of the FLOPs to achieve the same level of precision (-9.0%) . BlockDrop also exhibits a much higher variance in its FLOPs over other methods. Compared to SACT, this variance is 3 times larger, allowing some samples to achieve a speedup as high as 85% with correct predictions. Further, BlockDrop also outperforms PFEC \cite{molchanov2016pruning} and LCCL \cite{wu2018lcc}, which are complementary compression techniques and can be utilized together with our framework to speed up convolution operations.

Figure 3 (b) presents the results for ImageNet. Compared with the original ResNet-101 model, BlockDrop again achieves slightly better results (78.8% vs. 78.4%) with 6% speed up (1.47 × 10^10 vs. 1.56 × 10^10 FLOPs). BlockDrop performs on par with the full ResNet with a 20% speed up (1.25 × 10^10 vs. 1.56 × 10^10 FLOPs) when we relax γ slightly. This result without degradation in accuracy is quite promising. For example, in a high-precision
4.3 Qualitative Results

Finally, we provide qualitative results based on our learned policies to visualize the learned features and compare them to the results from standard ResNets, which seem to cater to different “styles” of images of the particular class. For volcanoes, these include features like smoke, lava, etc., while for orange objects they indicate whether it is sliced/whole, quantity.

Figure 5. Samples from ImageNet classes. Easy and hard samples from goldfish, arctos, space/craft and bridge to illustrate how block usage translates to instance difficulty. occlusion, or clattered background in samples that require more blocks. This confirms our hypothesis that block usage is a function of instance difficulty. We stress that this “sorting” into easy or hard cases falls out automatically: it is learned by BlockDrop.

5. Conclusion

We presented BlockDrop, an approach for faster inference in ResNets by selectively choosing residual blocks to evaluate in a learned and optimized manner conditioned on inputs. In particular, we trained a policy network to predict blocks to drop in a pretrained ResNet while trying to retain the prediction accuracy. The ResNet is further jointly finetuned to produce smooth feature representations tailored for block dropping behaviour. We conducted extensive experiments on CIFAR and ImageNet, observing considerable gains over existing methods in terms of the efficiency-accuracy trade-off. Further, we also observe that the policies learned encode semantic information in the images.

Acknowledgments. Required Funds is supported by ISPAR via DOD/IIRS contact number D17PS0841J. The U.S. Government is authorized to reproduce and distribute photoduplication of this work for Governmental purposes notwithstanding any copyright notation thereon. Disclaimer. The views and conclusions contained herein are those of the authors and shall not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ISPAR, DOD/ IIRS, or the U.S. Government. Krishnendu De is supported in part by an IBM Faculty Award and IBM Open Collaboration Award.

References

Supplemental Materials

Details of BlockDrop-seq (Ours-seq)

We construct a sequential version of BlockDrop for dropping blocks, where the decision $a_t \in \{0, 1\}$ to drop or keep the $t$-th block is conditioned on the activations of its previous block, $y_{t-1}$. Unlike BlockDrop, where all the actions are predicted in one shot, this model predicts one action at a time, which is a typical reinforcement learning setting. We follow the procedure to generate the halting scores in [11], and arrive at an equivalent per-block skipping score according to:

$$p_t = \text{softmax}(\tilde{W} \cdot \text{pool}(y_{t-1}) + b)$$

where pool is a global average pooling operation. For fair comparisons, Ours-seq is compared to a BlockDrop model, which attains equivalent accuracy, with the same number of blocks.

Implementation Details

- On CIFAR, we train the model for 5000 epochs during curriculum learning with a batch size of 2048 and a learning rate of $e^{-1}$ for 500 epochs.
- On ImageNet, the policy network is trained for 45 epochs for curriculum learning with a batch size of 2048 and a learning rate of $e^{-4}$, which is annealed to $e^{-4}$ for 400 epochs.

Detailed Results on CIFAR-10 and ImageNet

We present detailed results of our method on CIFAR-10 (Table 3) and ImageNet (Table 4). We highlight the accuracy, block usage and speed up for variants of our model compared to full ResNets.

<table>
<thead>
<tr>
<th>Network</th>
<th>FLOPs</th>
<th>Block Usage</th>
<th>Accuracy</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-32</td>
<td>1.38E+08</td>
<td>0.00E+00</td>
<td>76.1</td>
<td>92.3</td>
</tr>
<tr>
<td>ResNet-110</td>
<td>5.00E+08</td>
<td>0.00E+00</td>
<td>75.8</td>
<td>93.2</td>
</tr>
<tr>
<td>BlockDrop-32 ($\gamma=5$)</td>
<td>8.66E+07</td>
<td>1.40E+07</td>
<td>76.8</td>
<td>91.3</td>
</tr>
<tr>
<td>BlockDrop-110 ($\gamma=2$)</td>
<td>1.18E+08</td>
<td>2.46E+07</td>
<td>76.8</td>
<td>91.9</td>
</tr>
<tr>
<td>BlockDrop-110 ($\gamma=3$)</td>
<td>1.51E+08</td>
<td>3.24E+07</td>
<td>76.8</td>
<td>93.0</td>
</tr>
<tr>
<td>BlockDrop-110 ($\gamma=4$)</td>
<td>1.82E+08</td>
<td>3.43E+07</td>
<td>76.8</td>
<td>93.6</td>
</tr>
</tbody>
</table>

Table 3: Results of different architectures on CIFAR-10. Depending on the base ResNet architecture, speedups ranging from 37% to 76% are observed with little to no degradation in performance.

<table>
<thead>
<tr>
<th>Network</th>
<th>FLOPs</th>
<th>Block Usage</th>
<th>Accuracy</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-72</td>
<td>1.17E+08</td>
<td>0.00E+00</td>
<td>76.4</td>
<td>75.8</td>
</tr>
<tr>
<td>ResNet-75</td>
<td>1.21E+08</td>
<td>0.00E+00</td>
<td>76.4</td>
<td>75.9</td>
</tr>
<tr>
<td>ResNet-84</td>
<td>1.34E+08</td>
<td>0.00E+00</td>
<td>76.4</td>
<td>76.1</td>
</tr>
<tr>
<td>BlockDrop-72 ($\gamma=2$)</td>
<td>9.85E+08</td>
<td>3.34E+08</td>
<td>76.4</td>
<td>75.2</td>
</tr>
<tr>
<td>BlockDrop-72 ($\gamma=3$)</td>
<td>1.25E+08</td>
<td>4.26E+08</td>
<td>76.4</td>
<td>75.2</td>
</tr>
<tr>
<td>BlockDrop-72 ($\gamma=4$)</td>
<td>1.47E+08</td>
<td>4.02E+08</td>
<td>76.4</td>
<td>75.7</td>
</tr>
</tbody>
</table>

Table 4: Results of different architectures on ImageNet. BlockDrop is built upon ResNet-101, and can achieve around 20% speedup on average with $\gamma = 5$. 

---

Revisiting Dilated Convolution: A Simple Approach for Weakly- and Semi-Supervised Semantic Segmentation

Yunchao Wei1, Huaxin Xiao2, Honghui Shi3, Zequn Jie4, Jiashi Feng2, Thomas S. Huang1
1 UIUC 2 NUS 3 IBM Thomas J. Watson Research Center 4 Tencent AI Lab
{wychao1987, huaxinxiao89, shihonghui3, zequn.nus, jshfeng}@gmail.com huang@ifp.uiuc.edu

Abstract

Despite the remarkable progress, weakly supervised segmentation approaches are still inferior to their fully supervised counterparts. We observe the performance gap mainly comes from their limitation on learning to produce high-quality dense object localization maps from image-level supervision. To mitigate such a gap, we revisit the dilated convolution [1] and reveal how it can be utilized in a novel way to effectively overcome this critical limitation of weakly supervised segmentation approaches. Specifically, we find that varying dilation rates can effectively enlarge the receptive fields of convolutional kernels and more importantly transfer the surrounding discriminative information to non-discriminative object regions, promoting the emergence of these regions in the object localization maps. Then, we design a generic classification network equipped with convolutional blocks of different dilated rates. It can produce dense and reliable object localization maps and effectively benefit both weakly- and semi-supervised semantic segmentation. Despite the apparent simplicity, our proposed approach obtains superior performance over state-of-the-arts. In particular, it achieves 60.8% and 67.6% mIoU scores on Pascal VOC 2012 test set in weakly- (only image-level labels are available) and semi- (1,464 segmentation masks are available) supervised settings, which are the new state-of-the-arts.

1. Introduction

Weakly-supervised image recognition approaches [11, 15, 16, 19, 23, 38–42, 44] have been extensively studied as they do not require expensive human effort. Among them, the most attractive one is learning to segment images from only image-level annotations. For such approaches, the arguably most critical challenge remaining unsolved is how to accurately and densely localize object regions to obtain high-quality object cues for initializing and improving the segmentation model training [1, 20, 45].

Recently, some top-down approaches [43, 46] propose to leverage a classification network to produce class-specific attention cues for object localization. However, directly employing attentions produced by image classification models can only identify a small discriminative region of the target object, which is not sufficiently dense and extensive for training a good segmentation model. For instance, some samples of class-specific region localization produced by the state-of-the-art Class Activation Mapping (CAM) [46] are shown in the second row of Figure 1 (b). One can ob-
serve that CAM hardly generates dense object regions in usual cases where large objects are present, which deviates from requirement of the semantic segmentation task. Those regions discovered by CAM are usually scattered around the target object, e.g., some discriminative parts such as head and hands of the child. Inability to learn to produce dense object localization from image-level supervision is a critical obstacle to developing well performing weakly supervised segmentation models. Based on such an observation, we propose to transfer discriminative knowledge from those surprisingly highlighted regions to adjacent object regions and thus form dense object localization, which can essentially lift segmentation model learning favorably.

To this end, we revisit the popular dilated convolution and find it indeed provides promising solution up to proper utilization. Dilated convolution was initially introduced by Chen et al. [1, 2] for semantic segmentation. One key advantage is that it can effectively enlarge receptive field size to incorporate context without introducing extra parameters or computation cost. We find such a feature well fits propagating discriminative knowledge from image regions and highlighting non-discriminative object regions to produce dense object localization. Motivated by this, we introduce multiple dilated convolutional blocks to augment a standard classification model, as shown in Figure 1 (a).

In particular, our proposed approach expands receptive fields at multiple scales by varying dilated rates of convolutional kernels. In general, classification networks are able to identify one or more small discriminative parts with high response for correctly recognizing images. By enlarging the receptive field, object regions with low response can still gain improved discriminativeness through perceiving the context surrounding them. In this way, the discriminative information of high response parts of the target object can propagate to adjacent object regions at multiple scales, which can be identified by classification models. We utilize CAM [46] to generate an object localization map for each convolutional block. As shown in Figure 1 (a), the convolution block can only localize two small discriminative regions without enlarging dilation rate, i.e., d = 1. By gradually increasing the dilated rates (from 3 to 9), more object-related regions are discovered.

However, some true negative regions may be falsely highlighted with large dilated rates (e.g., the localization map corresponding to d = 9). We then propose a simple yet effective anti-noise fusion strategy to address this issue. This strategy can effectively suppress object-irrelevant regions. We observe that multiple receptive fields and fuse the localization maps produced by different dilated blocks into an integral one which sharply highlights object regions. As examples shown in Figure 1 (b), it can be observed that our approach is very robust to scale variation and is able to densely localize the target objects.

We use the localization maps generated by our proposed approach to produce segmentation masks for training segmentation models. Our approach is general and can be applied for learning semantic segmentation networks in both weakly- and semi-supervised manner. Despite its apparent simplicity, our approach indeed provides dense object localization that can easily boost the weakly- and semi-supervised segmentation to new state-of-the-arts, as demonstrated in extensive experiments. To sum up, the main contributions of this work are three-fold:

We revisit the dilated convolution and reveal that it naturally fits the requirement on densely localizing object regions for building a good weakly supervised segmentation model, which is new to weakly/semi-supervised semantic segmentation.

We propose a simple yet effective approach that leverages dilated convolution to densely localize objects by transferring discriminative segmentation information.

Our proposed approach is generic for learning semantic segmentation networks in both weakly- and semi-supervised manner. In particular, it achieves the mIoUs of 60.8% and 67.6% on test set of PASCAL VOC segmentation benchmark in weakly- and semi-supervised settings respectively, which are new state-of-the-arts.

2. Related Work

Segmentation with Coarse Annotations
Collecting a large number of pixel-level annotations for training semantic segmentation model is labor intensive. To reduce the burden of pixel-level annotation, Dai et al. [3] and Papandreou et al. [21] proposed to learn semantic segmentation with annotated bounding boxes. Lin et al. [28] employed semantic scribbles as supervision for semantic segmentation. More recently, the supervised annotation is further relaxed to instance points in [21].

Segmentation with Image-Level Annotations
Image-level label, which is easy to obtain, is the simplest supervision for learning to segment. Some works [22–24] proposed to utilize multiple instance learning for semantic segmentation with image-level labels. Papandreou et al. [21] proposed to dynamically predict foreground objects and background supervision based on an Expectation-Maximization algorithm. Recently, great progress [8, 9, 13, 14, 26, 29, 31, 34, 35] has been made on this challenging task. Wei et al. [35] and Qi et al. [26] utilized proposals to generate pixel-level annotations for supervision. However, making use of MCG [25] can improve the performance of their model. In this work, we propose a standard convolutional network to recognize this as a “bird” image. We adopt a 3×3 convolutional kernel to learn the following feature representation at the location indicated by the red cycle. By enlarging the dilated rate from 1 to 3 of a 3×3 kernel, the location near the head will be perceived and get their discriminativeness enhanced. By further increasing the dilated rates (d = 5, 9), the location near the head will get more and similarly facilitate the classification model to discover these regions. To prove the dilated convolution can indeed improve the discriminative ability of low response object regions, we produce the localization maps at different dilated rates using CAM [46]. We can observe that those low response object regions on the localization map of d = 1 can be effectively highlighted with various dilated rates. The produced localization maps are complementary according to different dilated rates, and thus integrating results from multiple dilated blocks is also necessary.

3.2. Multi-dilated Convolution for Localization

Motivated by the above findings, we present an augmented classification network with multi-dilated convolutional (MDC) blocks to produce dense object localization, as shown in Figure 3. The network is built upon the VGG16 [32] model pre-trained on ImageNet [4]. We remove those fully-connected layers, and one pooling layer to enlarge the resolution of feature maps. Then, convolutional blocks with multiple dilated rates (i.e., d = r = 1, 3, 5, 9) are appended to conv5 to localize object-related regions perceived by different receptive fields. After global average pooling (GAP), the produced representations further pass through several fully-connected layers for classification. We optimize the classification network by minimizing sigmoid cross-entropy loss, and the classification activation maps (CAM) [46] approach is then employed to produce the class-specific localization map for each block.

We implement two kinds of convolutional operations. 1) We apply the standard kernels, i.e., d = 1. In this case, we can obtain accurate localization maps in which some discriminative parts of the target object are highlighted but many object-related regions are missed. 2) To transfer the discriminative knowledge of sparsely highlighted regions to other object regions, we vary dilated rates to enlarge the receptive field of kernels. In this way, the discriminative features from the adjacent highlighted regions can be transferred to the object-related regions that have not been discovered. By enlarging the dilated rate from 1 to 3 of a 3×3 kernel, the location near the head will be perceived and get their discriminativeness enhanced. By further increasing the dilated rates (d = 5, 9), the location near the head will get more and similarly facilitate the classification model to discover these regions. To prove the dilated convolution can
3.3.1 Weakly-supervised Learning

For the weakly-supervised approach, we adopt a similar framework as the one proposed in [21, 33] to exploit those ignored pixels in the inferred segmentation masks and gain robustness to falsely labeled pixels, as shown in the upper part of Figure 4. In particular, we extract the confidence maps corresponding to ground truth image-level labels for inferring segmentation masks in an online manner, which together with the segmentation masks derived from dense localization maps serve as supervision.

We process the more formally. Let \( I_c \) denote an image from the weakly-supervised training set \( \mathcal{I}_w \). For any \( I_c \in \mathcal{I}_w \), \( m_c \) is the corresponding pseudo segmentation mask produced by a dense localization map and \( I \) is the label set where background category is also included. Our target is to train a segmentation model \((\theta)\) with learnable parameter \( \theta \). The FCN models the conditional probability of any label \( c \) at any location \( x \) of the class-specific confidence map \( f_{\theta}(I_c; \theta) \). Use \( \mathcal{M}_c \) to denote the online predicted segmentation mask of \( I_c \), which collaborates with \( m_c \) for supervision. The loss function for optimizing the weakly-supervised FCN is formulated as

\[
\min_{\theta} \sum_{I_c \in \mathcal{I}_w} J_{\theta}(f(I_c; \theta)) \tag{1}
\]

where

\[
J_{\theta}(f(I_c; \theta)) = 1 - \frac{1}{M_c} \sum_{c \in C} \log f_{\theta}(I_c; \theta)
\]

and \( I \) indicates the number of pixels.

3.3.2 Semi-supervised Learning

Along with a large quantity of images with image-level annotations, we are interested in utilizing pixel-level annotations over a small number of images to further push the segmentation performance. i.e. the semi-supervised learning setting. As shown in the bottom of Figure 4, both strongly and weakly annotated images can be easily combined to learn segmentation networks by sharing parameters. Let \( I_c \) denote an image from the strongly-supervised training set \( \mathcal{I}_s \) and \( M_c \) is the corresponding segmentation mask annotated by human. The loss function used for optimizing the semi-supervised FCN can be defined as

\[
\min_{\theta} \sum_{I_c \in \mathcal{I}_s} J_{\theta}(f(I_c; \theta)) + \sum_{I_c \in \mathcal{I}_w} J_{\theta}(f(I_c; \theta)) \tag{2}
\]

where

\[
J_{\theta}(f(I_c; \theta)) = 1 - \frac{1}{M_c} \sum_{c \in C} \log f_{\theta}(I_c; \theta)
\]

4. Experiments

4.1. Dataset and Settings

Dataset and Evaluation Metrics

The proposed approach is evaluated on the PASCAL VOC 2012 segmentation benchmark [5]. One background category and 20 object categories are annotated in this dataset. Following the common practice [1, 33], the number of training images is increased to 10,582 by augmentation. The validation and test subsets include 1,449 and 1,456 images, respectively. We evaluate the performance in terms of pixel mIoU averaged on 21 categories. For all experiments, only image-level labels are employed as supervision and detailed analysis is conducted on the validation set. We compare our approach with other state-of-the-arts on both validation and test sets. Those results on the test set are obtained by submitting the predicted results to the official PASCAL VOC evaluation server.

Training/Testing Setting

We adopt the convolutional layers of VGG16 [32] pre-trained on ImageNet [4] to initialize the classification network except for the newly added convolutional layers in the localization network. We use DeepLab v.2 for the localization network. 

4.2. Comparison with State-of-the-arts

Table 1. Comparison of weakly-supervised semantic segmentation methods on PASCAL VOC 2012 validation and test sets.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Training Set</th>
<th>validation</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervision: Scribbles</td>
<td>10K</td>
<td>63.1</td>
<td>-</td>
</tr>
<tr>
<td>Scribblesup [33]</td>
<td>10K</td>
<td>60.6</td>
<td>62.2</td>
</tr>
<tr>
<td>BoxSup [33]</td>
<td>10K</td>
<td>62.0</td>
<td>64.2</td>
</tr>
<tr>
<td>Supervision: Spot</td>
<td>10K</td>
<td>46.1</td>
<td>-</td>
</tr>
<tr>
<td>Scribblesup [17]</td>
<td>10K</td>
<td>51.6</td>
<td>-</td>
</tr>
</tbody>
</table>

4.3. Weakly- and Semi-Segmentation Learning

We apply the dense localization maps produced by the proposed approach for training weakly and semi-supervised segmentation models.
thus is more efficient as producing proposals and training on them are time consuming. Without any pixel-level supervision, our weakly-supervised results further approach those of scribble-based and box-based methods and outperform the spot-based approaches by more than 8.8%. We conduct additional comparison on PASCAL VOC testing set. Our method achieves the new state-of-the-art on this competitive benchmark, and outperforms the mIoU scores of others by more than 2.1%.

4.2.2 Semi-supervised Semantic Segmentation

For semi-supervised semantic segmentation, we mainly compare with WSSL [21] whose weakly annotations are image-level labels. To further validate the quality of dense localization maps, we also compare with approaches that have access to bounding boxes for supervision. We adopt the same strong/weak split as those baselines, i.e. 1:1K strongly annotated images and 9K weakly annotated images.

From Table 2, our approach achieves better results than WSSL under the same setting, i.e. 65.7% vs. 64.6% on the validation set and 67.6% vs. 66.2% on the test set. Furthermore, we also compare with other approaches which use object bounding boxes as weakly-supervised information instead of image-level labels. Even though our approach uses much weaker supervision, it still achieves competitive and better mIoU scores on validation and test sets, respectively.

4.3. Ablation Analysis

Then we analyze the effectiveness of the proposed dense object localization approach, and how it benefits both weakly- and semi-supervised semantic segmentation.

4.3.1 Strategy of Dense Object Localization

The adopted classification network for object localization is augmented with convolutional blocks with multiple dilation rates. The object-related cues from different dilated blocks can be integrated into dense and integral object regions. To verify this, samples of localization maps from different convolutional blocks and the fused results are visualized in Figure 5. We observe that the block \(d = 1\) is able to localize objects with high precision but low recall (most regions of the target object are missed). By making use of other blocks with larger dilations \((d = 3, 5, 6)\), some other object-related regions are highlighted, e.g. the body of the right cat \((d = 6)\) in the first row and some parts of the motorcycle in the first row \((d = 3, 5, 6)\). However, we note that some true negative regions are also highlighted if we adopt large dilation rates \((d\) localization maps corresponding to \(d = 6\) and \(d = 9\)). For instance, we can observe that the center region at the row 5, column 6 becomes discriminative for the category dog. The reason is that the enlarged kernel perceives the context around the dog when convolutional operation is conducted for the center pixels, which improves the discriminative ability of the produced convolutional features.

It can be observed that the true positive object-related regions are usually shared by two or more localization maps and the false positive regions are different according to dilation rates. To prevent the false object-related regions from being highlighted, we further make use of the proposed anti-noise fusion strategy, which can further demonstrate the effectiveness of this strategy for highlighting object and removing noise. Note that we also try to generate the dense localization map by averaging the localization maps from all convolutional blocks (including \(d = 1\)). The mIoU score drops almost 5% compared with using the current fusion strategy. Besides, there is no significant improvement in mIoU score using four convolution blocks that are with the same dilation rate \((d = 1)\) compared with that of using one block. Since conditional random field (CRF) has been considered as a standard post-processing operation for semantic segmentation and employed by all the previous works for further improving performance, we thus systematically use CRF to refine the predicted masks for a fair comparison with other state-of-the-arts. We can observe that our approach can finally achieve the mIoU score of 60.4% and 60.8% on validation and test sets respectively and outperform all the other weakly-supervised methods.

4.3.3 Semi-supervised Semantic Segmentation

Table 4 shows the results of using different strong/weak splits for learning segmentation networks. We observe that the performance is gradually improved (from 50.3% to 54.4%) by enlarging the dilation rate of the convolutional kernel, which can further validate the effectiveness of using dilated convolutional blocks for object localization. Furthermore, the mIoU score can be further improved to 57.1% based on the dense localization maps produced by the proposed anti-noise fusion strategy, which can further demonstrate the effectiveness of this strategy for highlighting object and removing noise. Note that we also try to generate the dense localization map by averaging the localization maps from all convolutional blocks (including \(d = 1\)). The mIoU score drops almost 1% compared with using the current fusion strategy. Besides, there is no significant improvement in mIoU score using four convolution blocks that are with the same dilation rate (e.g. \(d = 1\)) compared with that of using one block. Since conditional random field (CRF) has been considered as a standard post-processing operation for semantic segmentation and employed by all the previous works for further improving performance, we thus systematically use CRF to refine the predicted masks for a fair comparison with other state-of-the-arts. We can observe that our approach can finally achieve the mIoU score of 60.4% and 60.8% on validation and test sets respectively and outperform all the other weakly-supervised methods.
Table 4. Comparison of mIoU scores using different strong/weak splits on PASCAL VOC 2012.

<table>
<thead>
<tr>
<th>strong</th>
<th>weak</th>
<th>semi-500</th>
<th>semi-1K</th>
<th>semi-1.4K</th>
<th>ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>strong 500</td>
<td>90.1</td>
<td>78.3</td>
<td>59.6</td>
<td>74.6</td>
<td>63.2</td>
</tr>
<tr>
<td>strong 1K</td>
<td>90.3</td>
<td>78.5</td>
<td>59.8</td>
<td>74.8</td>
<td>63.4</td>
</tr>
<tr>
<td>strong 2K</td>
<td>90.5</td>
<td>78.6</td>
<td>60.0</td>
<td>75.0</td>
<td>63.6</td>
</tr>
</tbody>
</table>

Figure 6. Examples of predicted segmentation masks by our approach in weakly- and semi-supervised manner.

We revisited the dilated convolution and proposed to leverage multiple convolutional blocks of different dilated rates to generate dense object localization maps. Our approach is easy to implement and the generated dense localiza-

manner. We observe that the performance only drops 0.9% by decreasing the number of strong images from 1K to 500, which demonstrates that our method can easily obtain reliable segmentation results even with a small number of strong images. Based on the generated dense localiza-
tion maps, we achieve new state-of-the-art results (based on 1K-strong images) on validation and test sets with CRF post-processing. We also evaluate in another setting using where 2.9K strong images for training. We can see the corresponding mIoU score is 68.5%, which is the same as re-
ported in [21]. Since both [21] and this work have been based on the TITAN Xp GPU used for this research.

References

[1] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolu-
tional nets and fully connected crfs. *arXiv preprint arX-


[7] S. Hong, J. Oh, B. Han, and H. Lee. Learning transferable knowledge for semantic segmentation with deep convolu-


[14] A. Kolesnikov and C. H. Lampert. Seed, expand and con-

sume: Three principles for weakly-supervised image seg-


[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ra-
man, P. Dollár, and C. L. Zitnick. Microsoft coco: Common-


volutional neural networks for weakly supervised segmenta-

[22] D. Pathak, E. Shelhamer, J. Long, and T. Darrell. Fully con-
volutional multi-class multiple instance learning. *arXiv*

[23] P. Olah and R. Collobert. From image-level to pixel-

[24] A. Roy and S. Todorovic. Combining bottom-up, top-down, and smoothness cues for weakly supervised image seg-

[25] N. S. Liu, J. Shi, H. Zhao, and J. Jia. Augmented feed-
back in semantic segmentation under image level supervi-


[28] X. Qi, Z. Liu, J. Shi, R. Zhao, and J. Jia. Augmented feed-

[29] D. Pathak, P. Krahenbuhl, and T. Darrell. Constrained con-
volutional neural networks for weakly supervised segmenta-

volutional multi-class multiple instance learning. *arXiv*

[31] P. Olah and R. Collobert. From image-level to pixel-

[32] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolu-
tional nets and fully connected crfs. *preprint arX-

[33] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully con-

[34] J. Dai, K. He, and J. Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic seg-

[35] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ra-
man, P. Dollár, and C. L. Zitnick. Microsoft coco: Common-


---

**Section 4**

Perception

---
5 Mastering Language
"A special kind of beauty exists which is born in language, of language, and for language.”

Gaston Bachelard

The big breakthroughs seen in computer vision now extend to natural language processing as well with the development of new deep-learning based methods for tackling various natural language understanding tasks. We can now optimize vector representations of words and sentences to capture interesting semantic properties using neural networks. Yet computers are still very far from understanding or generating language at a human level. Our research explores holistic language understanding – moving from representing words to modeling whole sentences and even documents, allowing us to make progress in reading comprehension, question-answering and tasks that also involve natural language generation, such as rephrasing and even creating poetry. Reading comprehension, together with language generation, form the basis for conversational technologies.

In this chapter, we present our research work ranging from fundamental natural language processing to grasping human language complexities, focusing on those apparent in enterprise scenarios. These include problems such as coreference resolution [22], emotion identification in conversations [23], rephrasing and language generation [24][25] and building knowledge from text [26].

Project Debater, unveiled this year, demonstrates innovation in language, focusing on language generation, listening comprehension and modeling of knowledge. We highlight two papers published this year that present research supporting Project Debater. The first focuses on weak-learning methods for obtaining high quality data from small datasets [27] and the second addresses the additional aspect of listening comprehension, taking automatic speech recognition one step forward, allowing for semantic interpretation of human speech in a given context [28].

Selected Publications

22 – Detecting Egregious Conversations between Customers and Virtual Agents
p – 338

23 – Fighting Offensive Language on Social Media with Unsupervised Text Style Transfers
p – 350

24 – Exemplar Encoder-Decoder for Neural Conversation Generation
p – 358

25 – Neural Cross-Lingual Coreference Resolution And Its Application To Entity Linking
p – 370

26 – Exploiting Structure in Representation of Named Entities using Active Learning
p – 378

27 – Will it Blend? Blending Weak and Strong Labeled Data in a Neural Network for Argumentation Mining
p – 392

28 – Listening Comprehension over Argumentative Content
p – 400
Detecting Egregious Conversations between Customers and Virtual Agents

Tommy Sandbank,
Michal Shmueli-Scheuer,
Jonathan Herzig, David Konopnicki
IBM Research
Haifa 31905, Israel
{tommym,shmueli,hjon,davidko}@il.ibm.com

John Richards,
David Piorkowski
IBM Research
Yorktown Heights, NY USA
ajtr@us.ibm.com, david.piorkowski@ibm.com

Abstract

Virtual agents are becoming a prominent channel of interaction in customer service. Not all customer interactions are smooth, however, and some can become almost comically bad. In such instances, a human agent might need to step in and salvage the conversation. Detecting bad conversations is important since disappointing customer service may threaten customer loyalty and impact revenue. In this paper, we outline an approach to detecting such egregious conversations, using behavioral cues from the user, patterns in agent responses, and user-agent interaction. Using logs of two commercial systems, we show that using these features improves the detection F1-score by around 20% over using textual features alone. In addition, we show that those features are common across two quite different domains and, arguably, universal.

1 Introduction

Automated conversational agents (chatbots) are becoming widely used for various tasks such as personal assistants or as customer service agents. Recent studies project that 80% of businesses plan to use chatbots by 2020, and that chatbots will power 85% of customer service interactions by the year 2020. This increasing usage is mainly due to advances in artificial intelligence and natural language processing (Hirschberg and Manning, 2015)

1http://read.bi/2gU0szG
2http://bit.ly/2z428RS

along with increasingly capable chat development environments, leading to improvements in conversational richness and robustness. Still, chatbots may behave extremely badly, leading to conversations so off-the-mark that only a human agent could step in and salvage them. Consequences of these failures may include loss of customer goodwill and associated revenue, and even exposure to litigation if the failures can be shown to include fraudulent claims. Due to the increasing prevalence of chatbots, even a small fraction of such egregious conversations could be problematic for the companies deploying chatbots and the providers of chatbot services.

In this paper we study detecting these egregious conversations that can arise in numerous ways. For example, incomplete or internally inconsistent training data can lead to false classification of user intent. Bugs in dialog descriptions can lead to dead ends. Failure to maintain adequate context can cause chatbots to miss anaphoric references. In the extreme case, malicious actors may provide heavily biased (e.g., the Tay chatbot) or even hacked misbehaviors.

In this article, we focus on customer care systems. In such setting, a conversation usually becomes egregious due to a combination of the aforementioned problems. The resulting customer frustration may not surface in easily detectable ways such as the appearance of all caps, shouting to a speech recognizer, or the use of profanity or extreme punctuation. Consequently, the chatbot will continue as if the conversation is proceeding well, usually

1Defined by the dictionary as outstandingly bad.
Detecting Egregious Conversations between Customers and Virtual Agents

customer frustration (Amsel, 1992). Along with the previous responses could lead which is rejected by the system (“We don’t default response (“I’m not trained on that”). The chatbot has insufficient exposure to this turns, the chatbot misses the customer’s innumerous features that lead to egregious conversations. The works of (Sarkaya, 2017; Sano et al., 2017) studied reasons why users reformulated utterances in such systems. Specifically, in (Sarkaya, 2017) they reported on how the different reasons affect the users’ satisfaction. In (Sano et al., 2017) they focused on how to automatically predict the reason for user’s dissatisfaction using different features. Our work also explores user reformulation (or rephrasing) as one of the features to predict egregious conversations. We build on the previous work by leveraging some of the approaches in our classifier for egregious conversations. In (Walker et al., 2006; Hastie et al., 2002) the authors also looked for problems in a specific setting of spoken conversations. The main difference with our work is that we focus on chat logs for domains for which the expected user utterances are a bit more diverse, using interaction features as well as features that are not sensitive to any architectural aspects of the conversational system (e.g., ASR component). Several other approaches for evaluating chatbot conversations indirectly capture the notion of conversational quality. For example, several prior works borrowed from the field of pragmatics in various metrics around the principles of cooperative conversation (Chakrabarti and Luger, 2013; Saygin A. P., 2002). In (Steidl et al., 2017) they focused on how to automatically detect egregious conversations as they were unfolding in real time. To perform egregious conversations, features from both user inputs and agent responses are extracted, together with features related to the combination of specific inputs and responses. In addition, some of these features are contextual, meaning that they are dependent on where in the conversation they appear. Using this set of features for detecting egregious conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhelm the analyst to scan system logs by eye, the sheer volume of conversations may overwhel...
gious conversations is novel, and as our experimental results show, improves performance compared to a model based solely on features extracted from the conversation’s text. We now describe the agent, customer, and combined customer-agent features.

### 3.1 Agent Response Features

A virtual agent is generally expected to closely simulate interactions with a human operator (Reeves and Nass, 1996; Nass and Moon, Y, 2000; Kramer, 2008). When the agent starts losing the context of a conversation, fails in understanding the customer intention, or keeps repeating the same responses, the illusion of conversing with a human is lost and the conversation may become extremely annoying. With this in mind, we now describe the analysis of the agent’s responses and associated features (summarized in the top part of Table 1).

#### 3.1.1 Repeating Response Analysis

As typically implemented, the virtual agent’s task is to reliably detect the intent of each customer’s utterance and respond meaningfully. Accurate intent detection is thus a fundamental characteristic of well-trained virtual agents, and incorrect intent analysis is reported as the leading cause of user dissatisfaction (Sarikaya, 2017). Moreover, since a classifier (e.g., SVM, neural network, etc.) is often used to detect intents, its probabilistic behavior can cause the agent to repeat the same (or semantically similar) response over and over again, despite the user’s attempt to rephrase the same intent.

Such agent repetitions lead to an unnatural interaction (Kluwer, 2011). To identify the agent’s repeating responses, we measured similarity between agent’s subsequent (not necessarily sequential) turns. We represented each sentence by averaging the pre-trained embeddings of each word in the sentence, calculating the cosine similarity between the representations. Turns with a high similarity value are considered as repeating responses.

#### 3.1.2 Unsupported Intent Analysis

Given that the knowledge of a virtual agent is necessarily limited, we can expect that training would not cover all customer intents. If the classifier technology provides an estimate of classification confidence, the agent can respond with some variant of “I’m not trained on that” when confidence is low. In some cases, customers will accept that not all requests are supported. In other cases, unsupported intents can lead to customer dissatisfaction (Sarikaya, 2017), and cascade to an egregious conversation (as discussed below in Section 3.3). We extracted the possible variants of the unsupported intent messages directly from the system, and later matched them with the agent’s responses from the logs.

#### 3.2 Customer Inputs Features

From the customer’s point of view, an ineffective interaction with a virtual agent is clearly undesirable. An ineffective interaction requires the expenditure of relatively large effort from the customer with little return on the investment (Zeithaml et al., 1990; Mmoum et al., 2012). These efforts can appear as behavioral cues in the customer’s inputs, and include emotions, repetitions, and more. We used the following customer analysis in our model. Customer features are summarized in the middle part of Table 1.

#### 3.2.1 Rephrasing Analysis

When a customer repeats or rephrases an utterance, it usually indicates a problem with the agent’s understanding of the customer’s intent. This can be caused by different reasons as described in (Sano et al., 2017). To measure the similarity between subsequent customer turns to detect repetition or rephrasing, we used the same approach as described in Section 3.1.1. Turns with a high similarity value are considered as rephrases.

#### 3.2.2 Emotional Analysis

The customer’s emotional state during the conversation is known to correlate with the conversation’s quality (Oliver, 2014). In order to analyze the emotions that customers exhibit in each turn, we utilized the IBM Tone Analyzer service, available publicly online.

### 3.3 Similarity Analysis

This service was trained using customer care interactions, and infers emotions such as frustration, sadness, happiness. We focused on negative emotions (denoted as NEG EMO) to identify turns with a negative emotional peak (i.e., single utterances that carried high negative emotional state), as well as to estimate the aggregate negative emotion throughout the conversation (i.e., the averaged negative emotion intensity). In order to get a more robust representation of the customer’s negative emotional state, we summed the score of the negative emotions (such as frustration, sadness, anger, etc.) into a single negative sentiment score (denoted as NEG SENT). Note that we used the positive emotions as a filter for other customer features, such as the rephrasing analysis. Usually, high positive emotions capture different styles of “thanking the agent”, or indicate that the customer is somewhat satisfied (Rychalski and Hudson, 2017), thus, the conversation is less likely to become egregious.

#### 3.3.1 Similarity Features

In examining the conversation logs, we noticed that it is not unusual to find a customer asking to be transferred to a human agent. Such a request might indicate that the virtual agent is not providing a satisfactory service. Moreover, even if there are human agents, they might not be available at all times, and thus, a rejection of such a request is sometimes reasonable, but might still lead to customer frustration (Amsel, 1992).

#### 3.3.2 Unigram Input

In addition to the above analyses, we also detected customer turns that contain exactly one word. The assumption is that single word (unigram) sentences are probably short customer responses (e.g., no, yes, thanks, okay), which in most cases do not contribute to the egregiousness of the conversation. Hence, calculating the percentage of those turns out of the whole conversation gives us another measurable feature.

#### 3.3.3 Customer-Agent Interaction Features

We also looked at features across conversation utterance-response pairs in order to capture a more complete picture of the interaction between the customer and the virtual agent. Here, we considered a pair to be customer utterance followed by an agent response. For example, a pair may contain a turn in which the customer expressed negative emotions and received a response of “not trained” by the agent. In this case, we would leverage the two analyses: emotional and unsupported intent. Figure 1 gives an example of this in the customer’s penultimate turn. Such interactions may divert the conversation towards becoming egregious. These features are summarized in the last part of Table 1.

### 3.4 Conclusion

We also calculated the similarity between the customer’s turn and the virtual agent’s response in cases of customer rephrasing. This analysis aims to capture the virtual agent’s response to customer rephrasing. When a similarity score between the customer’s turn and the agent’s response is low, this may indicate a misclassified intent, as the agent’s responses are likely to share some textual similarity to the customer’s utterance. Thus, a low score may indicate a poor interaction, which might lead the conversation to become egregious. Another similarity feature is between two customer’s subsequent turns when the agent’s response was “not trained”.

### Table 1: Features sets description.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGENT RPT</td>
<td>Response is “not trained”</td>
</tr>
<tr>
<td>CUSTOMER SENT</td>
<td>Customer utterance is negative</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer did not understand the agent’s response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer kept repeating the same requests</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
<tr>
<td>AGENT RPT</td>
<td>Agent failed to understand the customer’s request</td>
</tr>
<tr>
<td>CUSTOMER RPT</td>
<td>Customer didn’t know what to say in response</td>
</tr>
</tbody>
</table>
3.4 Conversation Egregiousness Prediction Classifier

We trained a binary SVM classifier with a linear kernel. A feature vector for a sample in the training data is generated using the scores calculated for the described features, where each feature value is a number between [0,1].

After the model was trained, test conversations are classified by the model, after being transformed to a feature vector in the same way a training sample is transformed. The SVM classification model (denoted EGR) outputs a label “egregious” or “non-egregious” as a prediction for the conversation.

4 Experiments

4.1 Dataset

We extracted data from two commercial systems that provide customer support via conversational bots (hereafter denoted as company A and company B). Both agents are using similar underlying conversation engines, each embedded in a larger system with its own unique business logic. Company A’s system deals with sales support during an online purchase, while company B’s system deals with technical support for purchased software products. Each system logs conversations, and each conversation is a sequence of tuples, where each tuple consists of (conversation id, turn id, customer input, agent response). From each system, we randomly extracted 10000 conversations. We further removed conversations that contained fewer than 2 turns, as these are too short to be meaningful since the customer never replied or provided more details about the issue at hand. Figure 2 depicts the frequencies of conversation lengths which follow a power-law distribution and included 1100 and 200 conversations for conversations from our datasets. This sample in this purpose, we randomly sampled conversations, and each conversation is a sequence of tuples, where each tuple consists of (conversation id, turn id, customer input, agent response).

Figure 2: Frequency versus conversation length for company A and company B on a log-log scale.

The sampled conversations were tagged using an in-house tagging system designed to increase the consistency of human judgements. Each conversation was tagged by four different expert judges. Given the full conversation, each judge tagged whether the conversation was egregious or not following this guideline: “Conversations which are extraordinary bad in some way, those conversations where you’d like to see a human jump in and save the conversation”.

We generated true binary labels by considering each conversation to be egregious if at least three of the four judges agreed. The inter-rater reliability between all judges, measured by Cohen’s Kappa, was 0.72 which indicates high level agreement. This process generated the egregious class sizes of 95 (8.6%) and 16 (8%) for company A and company B, respectively. This verifies the unbalanced data expectation as previously discussed.

We also implemented two baseline models, rule-based and text-based, as follows:

- **Rule-based.** In this approach, we look for cases in which the virtual agent responded with a “not trained” reply, or occurrences of the customer requesting to talk to a human agent. As discussed earlier, these may be indicative of the customer’s dissatisfaction with the nature of the virtual agent’s responses.
- **Text-based.** A model that was trained to predict egregiousness given the conversation’s text (all customer and agent’s text durations). We also implemented two baseline models, rule-based and text-based, as follows:

4.2 Experimental Setup

The first step in building a classification model is to obtain ground truth data. For this purpose, we randomly sampled conversations from our datasets. This sample included 1100 and 200 conversations for company A and company B respectively.

Table 2: Cross-validation results for the baselines and EGR models.

<table>
<thead>
<tr>
<th></th>
<th>EGR</th>
<th>Non-EGR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model</td>
<td>F</td>
</tr>
<tr>
<td>Rule-based</td>
<td>0.28</td>
<td>0.45</td>
</tr>
<tr>
<td>Text-based</td>
<td>0.46</td>
<td>0.56</td>
</tr>
<tr>
<td>EGR</td>
<td>0.47</td>
<td>0.79</td>
</tr>
</tbody>
</table>

4.3 Classification Results

Table 2 depicts the classification results for both classes and the three models we explored. The EGR model significantly outperformed both baselines. Specifically, for the egregious class, the precision obtained by the text-based and EGR models were similar. This indicates that the text analyzed by both models encodes some information about egregiousness. On the other hand, for the recall and hence the F1-score, the EGR model relatively improved the text-based model by 41% and 18%, respectively. We will further analyze the models below.

We also studied how robust our features were: If our features generalize well, performance should not drop much when testing company B with the classifier trained exclusively on the data from company A. Although company A and company B share similar conversation engine platforms, they are completely different in terms of objectives, domain, terminology, etc. For this task, we utilized the 200 annotated conversations of company B as test data, and experimented with the different models, trained on company A’s data. The rule-based baseline does not require training, of course, and could be applied directly.

Table 3 summarizes the results showing that the performance of the EGR model is relatively stable (w.r.t the model’s performance when it was trained and tested on the same domain), with a degradation of only 9% in F1-score. In addition, the results also show that the text-based model performs poorly when applied to a different domain (F1-score of 0.11). This may occur since textual features are closely tied to the training domain.

Figure 3: Precision (P), Recall (R), and F1-score (F) for various group combinations.
Detecting Egregious Conversations between Customers and Virtual Agents

4.6 Models Analysis

4.6.1 Customer Rephrasing Analysis

Inspired by (Sitarkaya, 2017; Samo et al., 2017) we analyzed the customer rephrasing motivations for both the egregious and the non-egregious classes. First, we detected customer rephrasing as described in Section 3.2.1, and then assigned to each its motivation. Specifically, in our setting, the relevant motivations are: (1) Natural language understanding (NLU) error – the agent’s intent detection is wrong, and thus the agent’s response is semantically far from the customer’s turn; (2) Language generation (LG) limitation – the intent is detected correctly, but the customer is not satisfied by the response (for example, the response was too generic); (3) Unsupported intent error – the customer’s intent is not satisfied by the agent.

In order to detect NLU errors, we measured the similarity between the first customer turn (before the rephrasing) and the agent response. We followed the methodology presented in (Jovita et al., 2015) claiming that the best answer given by the system has the highest similarity value between the customer turn and the agent answer. Thus, if the similarity was < 0.8 we considered this as an erroneous detection. If the similarity was ≥ 0.8 we considered the detection as correct, and thus the rephrasing occurred due to LG limitation. To detect unsupported intent error we used the approach described in Section 3.1.2. As reported in table 4, rephrasing due to an unsupported intent is more common in egregious conversations (18% vs. 14%), whereas, rephrasing due to generation limitations (LG limitation) is more common in non-egregious conversations (57% vs. 33%). This indicates that customers are more tolerant of cases where the system understood their intent, but the response is not exactly what they expected, rather than cases where the system’s response was “not trained”. Finally, the percentage of rephrasing due to wrong intent detection (NLU error) is similar for both classes, which is somewhat expected as it is expected that similar underlying systems provided NLU support.

4.6.2 Recall Analysis

We further investigated why the EGR model was better at identifying egregious conversations (i.e., its recall was higher compared to the baseline models). We manually examined 26 egregious conversations that were identified correctly by the EGR model, but mis-classified by the other models. Those conversations were particularly prevalent with the agent agent response was not exactly the customer’s intent due to NLU errors or LG limitations. We did not encounter any unsupported intent errors leading to customer rephrasing, which affected the ability of the rule-based model to classify those conversations as egregious. In addition, the customer intents that appeared in those conversations were very diverse. While customer rephrasing was captured by the EGR model, for the text-based model some of the intents were new (did not appear in the training data) and thus were difficult for the model to capture.

5 Conclusions and Future Work

In this paper, we have shown how it is possible to detect egregious conversations using a combination of customer utterances, agent responses, and customer-agent interactional features. As explained, the goal of this work is to give developers of automated agents tools to detect and then solve problems created by exceptionally bad conversations. In this context, future work includes collecting more data and using neural approaches (e.g., RNN, CNN) for analysis, validating our models on a range of domains beyond the two explored here. We also plan to extend the work to detect egregious conversations in real time (e.g., for escalating to a human operators), and create log analysis tools to analyze the root causes of egregious conversations and suggest possible remedies.

References


In Section 5, Aude Rychalski and Sarah Hudson (2017) explored the asymmetric effects of customer emotions on satisfaction and loyalty in a utilitarian service context. Their work was published in the Journal of Business Research, 71, pages 84-91.

Tommy Sandbank, Michal Shmueli-Scheuer, Jonathan Herzig, David Konecnicky, and Rottem Shaul (2017) presented a tool for managing emotional hotspots for conversational agents. Their tool, EchoTool, was introduced in the Proceedings of the 22nd International Conference on Intelligent User Interfaces Companion. It was published by ACM, New York, NY, USA, IUI ’17 Companion, pages 125-128. The DOI for this publication is https://doi.org/10.1145/3030024.3038260.

Shumpei Sano, Nobuhiro Kaji, and Manabu Sasaki (2017) predicted causes of reformulation in intelligent assistants. Their work was published in the Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, pages 299-309.


Moller Sebastian, Klaus-Peter Engelbrecht, Christine Kalbol, Ina Wechsung, and Benjamin Weiss (2009) examined the taxonomy of quality of service and quality of experience of multimodal human-machine interaction. Their research was published in Quality of Multimedia Experience, 2009. QoMEx 2009. International Workshop on. IEEE.


Marilyn A. Walker, Rebecca Passonno, and Julie E. Roland (2001) conducted a quantitative and qualitative evaluation of DARPA Communicator spoken dialogue systems. Their work was published in the Proceedings of the 30th Annual Meeting on Association for Computational Linguistics. ACL ’01, pages 515-522.


Fighting Offensive Language on Social Media with Unsupervised Text Style Transfer

Cicero Nogueira dos Santos*
IBM Research
T.J. Watson Research Center
cicerons@us.ibm.com

Igor Melnyk*
IBM Research
T.J. Watson Research Center
igor.melnyk@ibm.com

Inkit Padhi*
IBM Watson
T.J. Watson Research Center
inkit.padhi@ibm.com

Abstract

We introduce a new approach to tackle the problem of offensive language in online social media. Our approach uses unsupervised text style transfer to translate offensive sentences into non-offensive ones. We propose a new method for training encoder-decoders using non-parallel data that combines a collaborative classifier, attention and the cycle consistency loss. Experimental results on data from Twitter and Reddit show that our method outperforms a state-of-the-art text style transfer system in two out of three quantitative metrics and produces reliable non-offensive transferred sentences.

1 Introduction

The use of offensive language is a common problem of abusive behavior on online social media networks. Various work in the past have attacked this problem by using different machine learning models to detect abusive behavior (Xiang et al., 2012; Warner and Hirschberg, 2012; Kwok and Wang, 2013; Wang et al., 2014; Nobata et al., 2016; Burnap and Williams, 2015; Davidson et al., 2017; Founta et al., 2018). Most of these work follow the assumption that it is enough to filter out the entire offensive post. However, a user that is consuming some online content may not want an entirely filtered out message but instead have it in a style that is non-offensive and still be able to comprehend it in a polite tone. On the other hand, for those users who plan to post an offensive message, if one could not only alert that a content is offensive and will be blocked, but also offer a polite version of the message that can be posted, this could encourage many users to change their mind and avoid the profanity.

In this work we introduce a new way to deal with the problem of offensive language on social media. Our approach consists on using style transfer techniques to translate offensive sentences into non-offensive ones. A simple encoder-decoder with attention (Bahdanau et al., 2014) would be enough to create a reasonable translator if a large parallel corpus is available. However, unlike machine translation, to the best of our knowledge, there exists no dataset of parallel data available for the case of offensive to non-offensive language. Moreover, it is important that the transferred text uses a vocabulary that is common in a particular application domain. Therefore, unsupervised methods that do not use parallel data are needed to perform this task.

We propose a method to perform text style transfer addressing two main challenges arising when using non-parallel data in the encoder-decoder framework: (a) there is no straightforward way to train the encoder-decoder because we cannot use maximum likelihood estimation on the transferred text due to lack of ground truth; (b) it is difficult to preserve content while transferring the input to a new style. We address (a) using a single collaborative classifier, as an alternative to commonly used adversarial discriminators, e.g., as in (Shen et al., 2017). We approach (b) by using the attention mechanism combined with a cycle consistency loss.

In this work we also introduce two benchmark datasets for the task of transferring offensive to non-offensive text that are based on data from two popular social media networks: Twitter and Reddit. We compare our method to the approach of Shen et al. (2017) using three quantitative metrics: classification accuracy, content preservation and perplexity. Additionally, some qualitative results are also presented with a brief error analysis.

*Equal contribution.
2 Method

We assume access to a text dataset consisting of two non-parallel corpora $X = X_0 \cup X_1$ with different style values $s_0$ and $s_1$ (offensive and non-offensive) of a total of $N = m + n$ sentences, where $|X_0| = m$ and $|X_1| = n$. We denote a randomly sampled sentence $i$ of style $s_i$ from $X$ as $x_i$, for $k \in \{1, \ldots, N\}$ and $i \in \{0, 1\}$. A natural approach to perform text style transfer is to use a regular encoder-decoder network. However, the training of such network would require parallel data. Since in this work we consider a problem of unsupervised style transfer on non-parallel data, we propose to extend the basic encoder-decoder by introducing a collaborative classifier and a set of specialized loss functions that enable the training on such data. Figure 1 shows an overview of the proposed style transfer approach. Note that for clarity, in Figure 1 we have used multiple boxes to show encoder, decoder and classifier, the actual model contains a single encoder and decoder, and one classifier.

As can be seen from Figure 1, the encoder (a GRU RNN, $E(x_i, s_i) = H_i$) takes as input a sentence $x_i$ together with its style label $s_i$, and outputs $H_i$, a sequence of hidden states. The decoder/classifier (also a GRU RNN, $C(H_i, s_i) = x'_i$ for $s_i, j \in \{0, 1\}$) takes as input the previously computed $H_i$ and a desired style label $s_j$ and outputs a sentence $x'_i$, which is the original sentence but transferred from style $s_j$ to style $s_i$. The hidden states $H_i$ are used by the decoder in the attention mechanism (Bahdanau et al., 2014), and in hidden states $t$ and $a$, which can be thought to be similar to an autoencoder loss in (1) but in the style domain. (forward-transfer step), due to the lack of parallel data, we cannot use the same approach. For this purpose, we propose to transfer $X$ back to $X$ (back-transfer step) and compute the reconstruction loss between $x'_i$ and $x_i$ (see Eq. (4)). Note also that we as transfer the text forward and backward, we also control the accuracy of style transfer using the classifier (see Eqs. (2), (3) and (5)). In what follows, we present the details of the loss functions employed in training.

2.1 Forward Transfer

Reconstruction Loss. Given the encoded input sentence $x_i$ and the decoded sentence $x'_i$, the reconstruction loss measures how well the decoder $G$ is able to reconstruct it:

$$L_{rec} = E_{x_i \sim X} \left[ \log \text{p}_G(x'_i | E(x_i, s_i), s_j) \right].$$ (1)

Classification Loss. Formulated as follows:

$$L_{class, od} = E_{x_i \sim X} \left[ \log \text{p}_G(x'_i | s_j) \right].$$ (2)

For the encoder-decoder this loss gives a feedback on the current generator’s effectiveness on transferring sentences to a new style. For the classifier, it provides an additional training signal from generated data, enabling the classifier to be trained in a semi-supervised regime.

Classification Loss - Original Data. In order to enforce a high classification accuracy, the classifier also uses a supervised classification loss, measuring the classifier predictions on the original (supervised) instances $x_i$ in $X$:

$$L_{class, od} = E_{x_i \sim X} \left[ \log \text{p}_G(x'_i | x_i) \right].$$ (3)

2.2 Backward Transfer

Reconstruction Loss. The back-transfer (or cycle consistency) loss (Zhu et al., 2017) is motivated by the difficulty of imposing constraints on the transferred sentences. Back-transfer transforms the transferred sentences $x'_i \rightarrow$ back to the original style $s_i$, i.e., $x'_i \rightarrow$ and compares them to $x_i$. This also implicitly imposes the constraints on the generated sentences and improves the content preservation. The loss is formulated as follows:

$$L_{back, rec} = E_{x_i \sim X} \left[ \log \text{p}_G(x_i | E(x'_i, s_i), s_j) \right].$$ (4)

which can be thought to be similar to an autoencoder loss in (1) but in the style domain.

Classification Loss. Finally, we ensure that the back-transferred sentences $x'_i$ have the correct style $s_i$:

$$L_{back, rec} = E_{x_i \sim X} \left[ \log \text{p}_G(x_i | E(x'_i, s_i), s_i) \right].$$ (5)

In summary, the training of the components of our architecture consists in optimizing the following loss function using SGD with back-propagation:

$$\mathcal{L}(G, E; H_i, C) = \min_{G, E} \mathcal{L}_{rec} + \mathcal{L}_{back, rec} + \mathcal{L}_{class, od} + \mathcal{L}_{class, od} + \mathcal{L}_{class, bkd}$$

3 Related Work

Most previous work that address the problem of offensive language on social media has focused on text classification using different machine learning methods (Xiang et al., 2012; Warner and Hirschberg, 2012; Kwok and Wang, 2013; Wang et al., 2014; Burnap and Williams, 2015; Nobata et al., 2016; Davidson et al., 2017; Founta et al., 2018). To the best of our knowledge, there is no previous work on approaching the offensive language problem using style transfer methods.

Different strategies for training encoder-decoders using non-parallel data have been proposed recently. Many of these methods borrow the idea of using an adversarial discriminator/classifier from the Generative Adversarial Networks (GANs) framework (Goodfellow et al., 2014) and/or use a cycle consistency loss. Zhu et al. (2017) proposed the pioneering use of the cycle consistency loss in GANs to perform image style transfer from non-parallel data. In the NLP area, some recent effort has been done on the use of non-parallel data for style/content transfer (Shen et al., 2017; Melnyk et al., 2017; Fu et al., 2018) and machine translation (Lample et al., 2018; Artetxe et al., 2018). Shen et al. (2017), Fu et al. (2018) and Lample et al. (2018) use adversarial classifiers as a way to force the decoder to transfer the encoded sentence from the original data, to a different style/language. Lample et al. (2018) and Artetxe et al. (2018) use the cycle consistency loss to enforce content preservation in the translated sentences. Our work differs from the previous mentioned work in different aspects: we propose a new relevant style transfer task that has not been previously explored; our proposed method combines a collaborative classifier with the cycle consistency loss, which gives more stable results. Note that a potential extension to a problem of multiple attributes transfer would still use a single classifier, while in (Shen et al., 2017; Fu et al., 2018) this may require as many discriminators as the number of attributes.

Another line of research connected to this work consists in the automatic text generation conditioned on stylistic attributes. (Hu et al., 2017) and (Tielker and Goldberg, 2017) are examples of this line of work which use labeled data during training.
4 Experiments

4.1 Datasets

We created datasets of offensive and non-offensive texts by leveraging Henderson et al. (2018)'s pre-processing of Twitter (Ritter et al., 2010) and Reddit Politics (Sorban et al., 2017) corpora, which contain a large number of social media posts. Henderson et al. (2018) have used Twitter and Reddit datasets to evaluate the impact of offensive language and hate speech in neutral dialogue systems. We classified each entry in the two datasets using the offensive language and hate speech classifier from (Davidson et al., 2017). For Reddit, since the posts are long, we performed the classification at the sentence level. We note that since ground truths (parallel data) is not available, it is important to use the same classifier for data generation and evaluation so as to have a fair comparison and avoid inconsistencies. Therefore, we use the classifier from (Davidson et al., 2017) to test the performance of the compared algorithms in Sec. 4.3.

For our experiments, we used sentences/tweets with size between 2 and 15 words and removed repeated entries, which were frequent in Reddit. The final datasets have the following number of instances: Twitter - train [58,642 / 1,962,224] (offensive / non-offensive), dev [7842] (offensive), test [7734]. Reddit - [122,319 / 7,109,473], dev [11,883], test [30,583]. In both training sets the number of non-offensive entries is much larger than of the offensive ones, which is a problem since the objective is to have the best possible transfer to the non-offensive domain. We limited the vocabulary size by using words with frequency equal or larger than 70 (20) in Reddit (Twitter) dataset. All the other words are replaced by a placeholder token.

4.2 Experimental Setup

In all the presented experiments, we have used the same model parameters and the same configuration: the encoder/decoder is a single layer GRU RNN with 200 hidden neurons; the classifier is a single layer CNN with a set of filters of width 1, 2, 3 and 4, and size 128 (the same configuration as in the discriminators of (Shen et al., 2017)). Following (Shen et al., 2017), we have also used randomly initialized word embeddings of size 100, and trained the model using Adam optimizer with the minibatch size of 64 and learning rate of 0.0005. The validation set has been used to select the best model by early stopping. Our model has a quite fast convergence rate and achieves good results within just 1 epoch for the Reddit dataset and 5 epochs for the Twitter dataset.

Our baseline is the model of Shen et al. (2017) and it has been used with the default hyperparameters setting proposed by the authors. We have trained the baseline neural net for three days using a K40 GPU machine, corresponding to about 13 epochs on the Twitter dataset and 5 epochs on the Reddit dataset. The validation set has also been used to select the best model by early stopping.

4.3 Results and Discussion

Although the method proposed in this paper can be used to transfer text in both directions, we are interested in transferring in the direction of offensive to non-offensive only. Therefore, all the results reported in this section correspond to this direction. In Table 1, we compare our method with the approach of Shen et al. (2017) using three quantitative metrics: (1) classification accuracy (Acc.), which we compute by applying Davidson et al. (2017)'s classifier to the transferred test sentences; (2) content preservation (CP), a metric recently proposed by Fu et al. (2018) which uses pre-trained word embeddings to compute the content similarity between transferred and original sentences. We use Glove embeddings of size 300 (Pennington et al., 2014); (3) perplexity (PPL), which is computed by a word-level LSTM language model trained using the non-offensive training sentences.

Table 1: Classification accuracy, content preservation and perplexity for two datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>System</th>
<th>Acc.</th>
<th>CP</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reddit</td>
<td>Shen17</td>
<td>87.66</td>
<td>0.894</td>
<td>93.59</td>
</tr>
<tr>
<td>Reddit</td>
<td>Ours</td>
<td>99.54</td>
<td>0.933</td>
<td>115.75</td>
</tr>
<tr>
<td>Twitter</td>
<td>Shen17</td>
<td>95.36</td>
<td>0.891</td>
<td>90.97</td>
</tr>
<tr>
<td>Twitter</td>
<td>Ours</td>
<td>99.63</td>
<td>0.947</td>
<td>162.75</td>
</tr>
</tbody>
</table>

As can be seen from the table, our proposed method achieves high accuracy on both datasets, which means that almost 100% of the time Davidson et al. (2017)'s classifier detects that the transferred sentences are non-offensive. In terms of the content preservation, for both datasets our method also produces better results (closer to the better) when compared to (Shen et al., 2017) . A reason for these good results can be found by checking the examples presented in Table 2. The use of the back transfer loss and the attention mechanism makes our model good at preserving the original sentence content while being precise at replacing offensive words by non-offensive ones. Also observe from Table 2 that, quite often, Shen et al. (2017)’s model changes many words in the original sentence significantly modifying the content.

On the other hand, our model produces worse results in terms of perplexity values. We believe this can be due to one type of mistake that is frequent among the transferred sentences and that is illustrated in Table 3. The model uses the same non-offensive word (e.g. big) to replace an offensive word (e.g. f**king) almost everywhere, which produces many unusual and unexpected sentences.

We have performed ablation experiments by removing some components of the proposed model. The results for the Twitter dataset are shown in Table 4. We can see that attention and back-transfer loss play important roles in the model. In particular, when both of them are removed (last row in Table 4), although the classification accuracy improves, the perplexity and the content preservation drop significantly. This behavior happens due to the trade off that the decoder has to balance when transferring a sentence from a style to another. The decoder must maintain a proper balance between transferring to the correct style and generating sentences of good quality. Each of these properties can be achieved on its own, e.g., copying the entire input sentence will give low perplexity and good content preservation but low accuracy, on the other hand, outputting a single keyword can give high accuracy but high perplexity and low content preservation. While the classification loss guides the decoder to generate sentences that belong to the target style, the back-transfer loss and the attention mechanism encourage the decoder to copy words from the input sentence. When both back-transfer loss and attention are removed, the model is encouraged to just meet the classification requirement in the transfer step.

Table 2: Example of offensive sentences from Reddit and Twitter and their respective transferred versions.

<table>
<thead>
<tr>
<th>System</th>
<th>Reddit</th>
<th>Twitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>‘I’m back bitches’.</td>
<td>‘I’m back bruh!’</td>
</tr>
<tr>
<td>Ours</td>
<td>‘I hope they pay me as a freelancer or no’.</td>
<td>‘club tomorrow’.</td>
</tr>
</tbody>
</table>

Table 3: Examples of common mistakes made by our proposed model.

Table 4: Ablations results for the Twitter dataset.

<table>
<thead>
<tr>
<th>System</th>
<th>Acc.</th>
<th>CP</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Attention</td>
<td>99.88</td>
<td>0.939</td>
<td>196.65</td>
</tr>
<tr>
<td>No Back Transfer</td>
<td>97.08</td>
<td>0.938</td>
<td>257.93</td>
</tr>
<tr>
<td>No Attn &amp; Back Trans</td>
<td>100.0</td>
<td>0.876</td>
<td>751.56</td>
</tr>
</tbody>
</table>

5 Conclusions

This work is a first step in the direction of a new promising approach for fighting abusive posts on social media. Although we focus on offensive language, we believe that further improvements on the proposed methods will allow us to cope with other types of abusive behaviors.

References

Mikel Artetxe, Goriko Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural ma-


Exemplar Encoder-Decoder for Neural Conversation Generation

Gaurav Pandey, Danish Contractor, Vineet Kumar and Sachindra Joshi
IBM Research AI
New Delhi, India
{gpandey1, dcontrac, vineeku6, jsachind}@in.ibm.com

Abstract

In this paper we present the Exemplar Encoder-Decoder network (EED), a novel conversation model that learns to utilize similar examples from training data to generate responses. Similar conversation examples (context-response pairs) from training data are retrieved using a traditional TF-IDF based retrieval model. The retrieved responses are used to create exemplar vectors that are used by the decoder to generate the response. The contribution of each retrieved response is weighed by the similarity of corresponding context with the input context. We present detailed experiments on two large data sets and find that our method outperforms state of the art sequence to sequence generative models on several recently proposed evaluation metrics. We also observe that the responses generated by the proposed EED model are more informative and diverse compared to existing state-of-the-art method.

1 Introduction

With the availability of large datasets and the recent progress made by neural methods, variants of sequence to sequence learning (seq2seq) (Sutskever et al., 2014) architectures have been successfully applied for building conversational systems (Serban et al., 2016, 2017b). However, despite these methods being the state-of-the-art frameworks for conversation generation, they suffer from problems such as lack of diversity in responses and generation of short, repetitive and uninteresting responses (Liu et al., 2016; Serban et al., 2016, 2017b). A large body of recent literature has focused on overcoming such challenges (Li et al., 2016a; Lowe et al., 2017).

In part, such problems arise as all information required to generate responses needs to be captured as part of the model parameters learnt from the training data. These model parameters alone may not be sufficient for generating natural conversations. Therefore, despite providing enormous amount of data, neural generative systems have been found to be ineffective for use in real world applications (Liu et al., 2016).

In this paper, we focus our attention on closed domain conversations. A characteristic feature of such conversations is that over a period of time, some conversation contexts are likely to have occurred previously (Lu et al., 2017b). For instance, Table 1 shows some contexts from the Ubuntu dialog corpus. Each row presents an input dialog context with its corresponding gold response followed by a similar context and response seen in training data – as can be seen, contexts for “installing dms”, “sharing files”, “blocking ufw ports” have all occurred in training data. We hypothesize that being able to refer to training responses for previously seen similar contexts could be a helpful signal to use while generating responses.

In order to exploit this aspect of closed domain conversations we build our neural encoder-decoder architecture called the Exemplar Encoder-Decoder (EED), that learns to generate a response for a given context by exploiting similar contexts from training conversations. Thus, instead of having the seq2seq model learn patterns of language only from aligned parallel corpora, we assist the model by providing it closely related (similar) samples from the training data that it can refer to while generating text.

Specifically, given a context c, we retrieve a set

\[ 359 \]
Table 1: Sample input contexts and corresponding gold responses from Ubuntu validation dataset along with similar contexts seen in training data and their corresponding responses. We refer to training data as training data for the Ubuntu corpus. The highlighted words are common between the gold response and the exemplar response.

<table>
<thead>
<tr>
<th>Input Context</th>
<th>Gold Response</th>
<th>Similar Context in training data</th>
<th>Associated Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1 if you want everything a lot do you need...</td>
<td>Lighten, gnome, lubm, gnome, lubm</td>
<td>if you're running a lot, it will probably want...</td>
<td>e.g. gnome, lubm, lubm, gnome, lubm</td>
</tr>
<tr>
<td>U2 if it's possible to share it in one...</td>
<td>white, show, is there...</td>
<td>is there a...</td>
<td>e.g. white, show, is there...</td>
</tr>
<tr>
<td>U2 if you can't get permissions (for...</td>
<td>white, show, is there...</td>
<td>is there a...</td>
<td>e.g. white, show, is there...</td>
</tr>
</tbody>
</table>

Contributions: We present a novel Exemplar Encoder-Decoder (EED) architecture that makes use of similar conversations, fetched from an index of training data. The retrieved context-response pairs are used to create exemplar vectors which are used by the decoder in the EED model, to learn the importance of training context-response pairs, while generating responses. We present detailed experiments on the publicly benchmarked Ubuntu dialog corpus data set (Lowe et al., 2015) as well a large collection of more than 127,000 technical support conversations. We compare the performance of the EED model with the existing state of the art generative models such as VHRED (Serban et al., 2016) and VHRED (Serban et al., 2017b). We find that our model out-performs these models on a wide variety of metrics such as the recently proposed Activity Entropy metrics (Serban et al., 2017a) as well as Embedding-based metrics (Lowe et al., 2015). In addition, we present qualitative insights into our results and we find that exemplar based responses are more informative and diverse. The rest of the paper is organized as follows. Section 2 briefly describes the recent works in neural dialogue generation. The details of the proposed EED model for dialogue generation are described in detail in Section 3. In Section 4, we describe the datasets as well as the details of the models used during training. We present quantitative and qualitative results of EED model in Section 5.

2 Related Work

In this section, we compare our work against other data-driven end-to-end conversation models. End-to-end conversation models can be further classified into two broad categories — generation based models and retrieval based models. Generation based models cast the problem of dialogue generation as a sequence to sequence learning problem. Initial works treat the entire context as a single long sentence and learn an encoder-decoder framework to generate response word by word (Shang et al., 2015; Vinyals and Le, 2015). This was followed by work that models context better by breaking it into conversation history and last utterance (Sordoni et al., 2015b). Context was further modeled effectively by using a hierarchical encoder decoder (HRED) model which first learns a vector representation of each utterance and then combines these representations to learn vector representation of context (Serban et al., 2016). Later, an alternative hierarchical model called VHRED (Serban et al., 2017b) was proposed, where generated responses were conditioned on latent variables. This leads to more informative responses and adds diversity to response generation. Models that explicitly incorporate diversity in response generation have also been studied in literature (Li et al., 2016b; Vijayakumar et al., 2016; Cao and Clark, 2017; Zhao et al., 2017).

Our work differs from the above as none of these above approaches utilize similar conversation contexts observed in the training data explicitly.

Retrieval based models on the other hand treat the conversation context as a query and obtain a set of responses using information retrieval (IR) techniques from the conversation logs (Ji et al., 2014). There has been further work where the responses are further ranked using a deep learning based model (Yan et al., 2016a,b; Qiu et al., 2017). On the other hand of the spectrum, end-to-end deep learning based rankers have also been employed to generate responses (Wu et al., 2017; Henderson et al., 2017). Recently a framework has also been proposed that uses a discriminative dialog network that ranks the candidate responses received from a response generator network and trains both the networks in an end to end manner (Lu et al., 2017a).

In contrast to the above models, we use the input contexts as well as the retrieved responses for generating the final responses. Contemporaneous to our work, a generative model for machine translation that employs retrieved translation pairs has also been proposed (Gu et al., 2017). We note that while the underlying premise of both the papers remains the same, the difference lies in the mechanism of incorporating the retrieved data.

3 Exemplar Encoder Decoder

3.1 Overview

A conversation consists of a sequence of utterances. At a given point in the conversation, the utterances expressed prior to it are jointly referred to as the context. The utterance that immediately follows the context is referred to as the response. As discussed in Section 1, given a conversational context, we wish to generate a response by utilizing similar context-response pairs from the training data. We retrieve a set of K exemplar context-response pairs from an inverted index creating the training data in an off-line manner. The input and the retrieved context-response pairs are then fed to the Exemplar Encoder Decoder (EED) network. A schematic illustration of the EED network is presented in Figure 1. The EED encoder combines the input context and the retrieved responses to create a set of exemplar vectors. The EED decoder then uses the exemplar vectors based on the similarity between the input context and retrieved contexts to generate a response. We now provide details of each of these modules.

3.2 Retrieval of Similar Context-Response Pairs

Given a large collection of conversations as (context, response) pairs, we index each response and its corresponding context in an $f - idf$ vector space. We further extract the last turn of a conversation and index it as an additional attribute of the context-response document pairs so as to allow directed queries based on it.

Given an input context, we construct a query that weighs the last utterance in the context twice as much as the rest of the context and use it to retrieve the top K similar context-response pairs from the index based on a BM25 (Robertson et al., 2009) retrieval model. These retrieved pairs form our exemplar context-response pairs $(e^{(1)}, r^{(1)}), \ldots, (e^{(K)}, r^{(K)})$, $1 \leq k \leq K$.
formulation about similar responses along with the encoded input context representation.

\[ e^{(k)} = [w_1; r_1^{(k)}], 1 \leq k \leq K \]  

(2)

The exemplar vectors \( e^{(k)} \), \( 1 \leq k \leq K \) are further used by the decoder for generating the ground truth response as described in the next section.

### 3.4 Exemplar Decoder Network

Recall that we want the exemplar responses to help generate the responses based on how similar the corresponding contexts are with the input context. More similar an exemplar context is to the input context, higher should be its effect in generating the response. To this end, we compute the similarity scores \( s^{(k)} \), \( 1 \leq k \leq K \) using the encodings computed in Section 3.3 as shown below:

\[ s^{(k)} = \exp\left(\epsilon^{(k)}\right) \sum_{k=1}^{K} \exp(\epsilon^{(k)}) \]  

(3)

Next, each exemplar vector \( e^{(k)} \) computed in Section 3.3, is fed to a decoder, where the decoder is responsible for predicting the ground truth response given the exemplar embedding. The objective function to be maximized, is expressed as a function of the scores \( s^{(k)} \), the decoding distribution \( p^{\text{dec}} \) and the exemplar vectors \( e^{(k)} \) as shown below:

\[ \mathcal{L} = \sum_{k=1}^{K} s^{(k)} \log p^{\text{dec}}(r | e^{(k)}) \]  

(4)

Note that we weigh the contribution of each exemplar vector to the final objective based on how similar the corresponding context is to the input context. Moreover, the similarities are differentiable function of the input and hence, trainable by back propagation. The model should learn to assign higher similarities to the exemplar contexts, whose responses are helpful for generating the correct response.

The model description uses encoder and decoder networks that can be implemented using any differentiable parametrized architecture. We discuss our choices for the encoders and decoder in the next section.

### 3.5 The Encoders and Decoder

In this section, we discuss the various encoders and the decoder used by our model. The context representation consists of an ordered sequence of utterances and each utterance can be further viewed as a sequence of words. Thus, context can be viewed as having multiple levels of hierarchies—at the word level and then at the utterance (sentence) level. We use a hierarchical recurrent encoder—popularly employed as part of the HRED framework for generating responses and query suggestions (Sordoni et al., 2015a; Serban et al., 2016, 2017a). The word-level encoder encodes the vectors representing words of an utterance to an utterance vector. Finally, the utterance-level encoder encodes the utterance vectors to a context vector.

Let \((u_1, \ldots, u_K)\) be the utterances present in the context. Furthermore, let \((w_{1}, \ldots, w_{M})\) be the words present in the \(n^{th}\) utterance for \(1 \leq n \leq N\). For each word in the utterance, we retrieve its corresponding embedding from an embedding matrix. The word embedding for \(w_m\) will be denoted as \(e_{w_m}\). The encoding of the \(n^{th}\) utterance can be computed iteratively as follows:

\[ h_{w_m} = f(r_{w_m}, u_{w_m}), 1 \leq m \leq M \]  

(5)

We use an LSTM (Hochreiter and Schmidhuber, 1997) to model the above equation. The last hidden state \(h_{u_m}\), referred to as the utterance encoding and will be denoted as \(h_u\).

The utterance-level encoder takes the utterance encodings \(h_1, \ldots, h_K\) as input and generates the encoding for the context as follows:

\[ c_m = f(r_{c}^{(m-1)}, h_{u}), 1 \leq n \leq N \]  

(6)

Again, we use an LSTM to model the above equation. The last hidden state \(c_{u}\) is referred to as the context embedding and is denoted as \(c_c\).

A single level LSTM is used for embedding the response. In particular, let \((w_1, \ldots, w_M)\) be the sequence of words present in the response. For each word \(w_i\), we retrieve the corresponding word embedding \(e_{w_i}\) from a word embedding matrix. The response embedding is computed from the word embeddings iteratively as follows:

\[ r_{w_i} = f(r_{w_{i-1}}, e_{w_i}), 1 \leq i \leq M \]  

(7)

Again, we use an LSTM to model the above equation. The last hidden state \(r_{w_M}\) is referred to as the response embedding and is denoted as \(r_r\).
create training context-response pairs. Table 3 depicts some statistics for this dataset:

<table>
<thead>
<tr>
<th>Size</th>
<th>Conversations</th>
<th>127,466</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Pairs</td>
<td>204,808</td>
<td></td>
</tr>
<tr>
<td>Validation Pairs</td>
<td>8,738</td>
<td></td>
</tr>
<tr>
<td>Test Pairs</td>
<td>8,756</td>
<td></td>
</tr>
<tr>
<td>(</td>
<td>V</td>
<td>)</td>
</tr>
</tbody>
</table>

Table 3: Dataset statistics for Tech Support dataset.

### 4.2 Model and Training Details

The EED and HRED models were implemented using the PyTorch framework (Paszke et al., 2017). We initialize the word embedding matrix as well as the weights of context and response encoders from the standard normal distribution with mean 0 and variance 0.1. The biases of the encoders and decoder are initialized with 0. The word embedding matrix is shared by the context and response encoders. For Ubuntu dataset, we use a word embedding size of 600, whereas the size of the hidden layers of the LSTMs in context and response encoders and the decoder is fixed at 1200. For Tech support dataset, we use a word embedding size of 128. Furthermore, the size of the hidden layers of the multiple LSTMs in context and response encoders and the decoder is fixed at 256. A smaller embedding size was chosen for the Tech Support dataset since we observed much less diversity in the responses of the Tech Support dataset as compared to Ubuntu dataset.

Two different encoders are used for encoding the input context (not shown in Figure 1 for simplicity). The output of the first context encoder is concatenated with the exemplar response vectors to generate exemplar vectors as detailed in Section 3.2. The output of the second context encoder is used to compute the scoring function as detailed in Section 3.4. For each input context, we retrieve 5 similar context-response pairs for Ubuntu dataset and 3 context-response pairs for Tech support dataset using the tf-idf mechanism discussed in Section 3.2.

We use the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 1e− 4 for training the model. A batch size of 20 samples was used during training. In order to prevent overfitting, we use early stopping with log-likelihood on validation set as the stopping criteria. In order to generate the samples using the proposed EED model, we identify the exemplar context that is most similar to the input context based on the learned scoring function discussed in Section 3.4. The corresponding exemplar vector is fed to the decoder to generate the response. The samples are generated using a beam search with width 5. The average per-word log-likelihood is used to score the beams.

### 5 Results & Evaluation

#### 5.1 Quantitative Evaluation

##### 5.1.1 Activity and Entity Metrics

A traditional and popular metric used for comparing a generated sentence with a ground truth sentence is BLEU (Papineni et al., 2002) and is frequently used to evaluate machine translation. The metric has also been applied to compute scores for predicted responses in conversations, but it has been found to be less indicative of actual performance (Liu et al., 2016; Sordoni et al., 2015a; Serban et al., 2017a), as it is extremely sensitive to the exact words in the ground truth response, and gives equal importance to stop words/phrases and informative words.

Serban et al. (2017a) recently proposed a new set of metrics for evaluating dialogue responses for the Ubuntu corpus. It is important to highlight that these metrics have been specifically designed for the Ubuntu corpus and evaluate a generated response with the ground truth response by comparing the coarse level representation of an utterance (such as entities, activities, Ubuntu OS commands). Here is a brief description of each metric:

- **Activity**: Activity metric compares the activities present in a predicted response with the ground truth response. Activity can be thought of as a verb. Thus, all the verbs in a response are mapped to a set of manually identified list of 192 verbs.

- **Entity**: This compares the technical entities that overlap with the ground truth response. A total of 3115 technical entities is identified using public resources such as Debian package manager apt.

Table 3: Activity & Entity metrics for the Ubuntu corpus. LSTM*, HRED* & VHRED* as reported by Serban et al. (2017a).

<table>
<thead>
<tr>
<th>Model</th>
<th>P</th>
<th>R</th>
<th>F1</th>
<th>P</th>
<th>R</th>
<th>F1</th>
<th>Acc.</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM</td>
<td>1.7</td>
<td>1.03</td>
<td>1.18</td>
<td>1.18</td>
<td>0.81</td>
<td>0.87</td>
<td>14.57</td>
<td>94.79</td>
</tr>
<tr>
<td>VHRED</td>
<td>6.43</td>
<td>4.31</td>
<td>4.63</td>
<td>3.28</td>
<td>2.41</td>
<td>2.53</td>
<td>20.2</td>
<td>92.02</td>
</tr>
<tr>
<td>HRED</td>
<td>5.93</td>
<td>4.05</td>
<td>4.34</td>
<td>2.81</td>
<td>2.16</td>
<td>2.22</td>
<td>22.2</td>
<td>92.58</td>
</tr>
<tr>
<td>EED</td>
<td>6.42</td>
<td>4.77</td>
<td>4.87</td>
<td>3.8</td>
<td>2.91</td>
<td>2.99</td>
<td>31.73</td>
<td>95.06</td>
</tr>
</tbody>
</table>

Table 4: Activity & Entity metrics for the Ubuntu corpus. LSTM*, HRED* & VHRED* as reported by Serban et al. (2017a).

- **Cmd**: This measure compares the time tense of ground truth with predicted response.

- **Tense**: This metric computes accuracy by comparing commands identified in ground truth utterance with a predicted response.

Table prepared using the publicly available script5 for all our computations. As the test outputs for HRED are not available for Technical Support dataset, we use our tool to compute the metrics.

#### 5.1.2 Embedding Metrics

Embedding metrics (Lowe et al., 2017) were proposed as an alternative to word by word comparison metrics such as BLEU. We use pre-trained Google news word embeddings6 similar to Serban et al. (2017b), for easy reproducibility as these metrics are sensitive to the word embeddings used. The three metrics of interest utilize the word vectors in ground truth response and a predicted response and are discussed below:

- **Average**: Average word embedding vectors are computed for the candidate response and ground truth. The cosine similarity is computed between these averaged embeddings. High similarity gives as indication that ground truth and predicted response have similar words.

- **Greedy**: Greedy matching score finds the most similar word in predicted response to ground truth response using cosine similarity.

- **Extrema**: Vector extrema score computes the maximum or minimum value of each dimension of word vectors in candidate response and ground truth.

Of these, the embedding average metric is the most reflective of performance for our setup. The extrema representation, for instance, is very sensitive to text length and becomes ineffective beyond single length sentences (Forgues et al., 2014). We use the publicly available script7 for all our computations. As the test outputs for HRED are not available for Technical Support dataset, we use our tool to compute the metrics.

---

6: https://google-news-embeddings-vectors-negsmall300d.bin from https://code.google.com/archive/p/word2vec/

7: https://github.com/julianser/bed-dlp-truncated/blob/master/Evaluation/embedding_metrics.py
We note that the improvement achieved by the proposed method on a large collection of technical support conversations. Finally, we evaluate the diversity of the generated responses for HRED, VHRED and the proposed EED for a few selected contexts along with the corresponding similar exemplar responses. As can be observed from the table, the responses generated by EED tend to be more specific to the input context as compared to the responses of HRED and VHRED. For example, in conversations 1 and 2 we find that both HRED and VHRED generate simple generic responses whereas EED generates responses with additional information such as the type of disk partition used or a command not working. This is also confirmed by the quantitative results obtained using activity and entity metrics in the previous section. We further observe that the exemplar responses contain informative words that are utilized by the EED model for generating the responses as highlighted in Table 7.

Table 7: Contexts, exemplar responses and responses generated by HRED, VHRED and the proposed EED model. We use the published responses for HRED and VHRED, GT indicates the ground truth response. The change of turn is indicated by →. The highlighted words in bold are common between the exemplar response and the response predicted by EED.

### References


Minghui Qiu, Feng-Lin Li, Siyu Wang, Xing Gao, Yan Chen, Wei Peng Zhao, Xiaojing Chen, Jun Huang, and Wei Chu. 2017. Alime chat: A sequence to sequence model. In ACL.


Rui Yan, Yiping Song, and Hua Wu. 2016a. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In SIGIR.

Rui Yan, Yiping Song, Xiangyang Zhou, and Hua Wu. 2016b. “shall i be your chat companion?”. Towards an online human-computer conversation system. In CIKM.

Neural Cross-Lingual Coreference Resolution And Its Application To Entity Linking

Gourab Kundu and Avirup Sil and Radu Florian and Wael Hamza
IBM Research
1101 Kitchawan Road
Yorktown Heights, NY 10598
{gkundu, avi, raduf, whamza}@us.ibm.com

Abstract

We propose an entity-centric neural cross-lingual coreference model that builds on multi-lingual embeddings and language-independent features. We perform both intrinsic and extrinsic evaluations of our model. In the intrinsic evaluation, we show that our model, when trained on English and tested on Chinese and Spanish, achieves competitive results to the models trained directly on Chinese and Spanish respectively. In the extrinsic evaluation, we show that our English model helps achieve superior entity linking accuracy on Chinese and Spanish test sets than the top 2015 TAC system without using any annotated data from Chinese or Spanish.

1 Introduction

Cross-lingual models for NLP tasks are important since they can be used on data from a new language without requiring annotation from the new language (Ji et al., 2014, 2015). This paper investigates the use of multi-lingual embeddings (Faruqui and Dyer, 2014; Upadhyay et al., 2016) for building cross-lingual models for the task of coreference resolution (Ng and Cardie, 2002; Pradhan et al., 2012). Consider the following text from a Spanish news article:

"Tormenta de nieve afecta a 100 millones de personas en EEUU. Unos 100 millones de personas enfrentaban el sábado nuevas dificultades tras la enorme tormenta de nieve de hace días en la costa este de Estados Unidos."

The mentions “EEUU” (“US” in English) and “Estados Unidos” (“United States” in English) are coreferent. A coreference model trained on English data is unlikely to coreference these two mentions in Spanish since these mentions did not appear in English data and a regular English style abbreviation of “Estados Unidos” will be “EU” instead of “EEUU”. But in the bilingual English-Spanish word embedding space, the word embedding of “EEUU” sits close to the word embedding of “US” and the sum of word embeddings of “Estados Unidos” sit close to the sum of word embeddings of “United States”. Therefore, a coreference model trained using English-Spanish bilingual word embeddings on English data has the potential to make the correct coreference decision between “EEUU” and “Estados Unidos” without ever encountering these mentions in training data.

The contributions of this paper are two-fold. Firstly, we propose an entity-centric neural cross-lingual coreference model. This model, when trained on English and tested on Chinese and Spanish from the TAC 2015 Trilingual Entity Discovery and Linking (EDL) Task (Ji et al., 2015), achieves competitive results to models trained directly on Chinese and Spanish respectively. Secondly, a pipeline consisting of this coreference model and an Entity Linking (henceforth EL) model can achieve superior linking accuracy than the official top ranking system in 2015 on Chinese and Spanish test sets, without using any supervision in Chinese or Spanish.

Although most of the active coreference research is on solving the problem of noun phrase coreference resolution in the Ontonotes data set, invigorated by the 2011 and 2012 CoNLL shared task (Pradhan et al., 2011, 2012), there are many important applications/end tasks where the mentions of interest are not noun phrases. Consider the sentence,

"(U.S. president Barack Obama who started ((his) political career) in (Illinois)), was born in (Hawaii)."

The bracketing represents the Ontonotes style...
noun phrases and underlines represent the phrases that should be linked to Wikipedia by an EL system. Note that mentions like “U.S.” and “Barack Obama” do not align with any noun phrase. Therefore, in this work, we focus on coreference on mentions that arise in our end task of entity linking and conduct experiments on TAC TriLingual 2015 data sets consisting of English, Chinese and Spanish.

2 Coreference Model

Each mention has a mention type (m-type) of either name or nominal and an entity type (e-type) of Person (PER) / Location (LOC) / GPE / Facility (FAC) / organization (ORG) (following standard TAC; Ji et al., 2015) notation.

The objective of our model is to compute a function that can decide whether two partially constructed entities should be co-referenced or not. We gradually merge the mentions in the given document to form entities. Mentions are considered in the order of names and then nominals and within each group, mentions are arranged in the order they appear in the document. Suppose, the sorted order of mentions are \( m_1, \ldots, m_n, m_{N_1}, \ldots, m_{N_r}, \ldots, m_{N_2}, N_1 \) where \( N_1 \) and \( N_r \) are respectively the number of the named and nominal mentions. A singleton entity is created from each mention. Let the order of entities be \( e_1, \ldots, e_{N_1}, e_{N_1+1}, \ldots, e_{N_1+N_2}, N_1+N_r \).

We merge the named entities with other named entities, then nominal entities with named entities in the same sentence and finally we merge nominal entities across sentences as follows:

Step 1: For each named entity \( e_1 (1 \leq i \leq N_1) \), antecedents are all entities \( e_j (1 \leq j \leq i-1) \) such that \( e_j \) and \( e_i \) have same e-type. Training examples are triplets of the form \( (e_j, e_i, y_{ij}) \). If \( e_i \) and \( e_j \) are co-referent (meaning, \( y_{ij} = 1 \)), they are merged.

Step 2: For each nominal entity \( e_i (N_1+1 \leq i \leq N_1+N_r) \), we consider antecedents \( e_j \) such that \( e_i \) and \( e_j \) have the same e-type and \( e_j \) has some mention mentionants in the same sentence as some mention in \( e_i \). Training examples are generated and entities are merged as in the previous step.

Step 3: This is similar to previous step, except \( e_i \) and \( e_j \) have no sentence restriction.

Features: For each training triplet \((e_1, e_2, y_{12})\), the network takes the entity pair \((e_1, e_2)\) as input and tries to predict \( y_{12} \) as output. Since each entity represents a set of mentions, the entity-pair embedding is obtained from the embeddings of mention pairs generated from the cross product of the entity pair. Let \( M(e_1, e_2) \) be the set \( \{ (m_1, m_2) \mid \exists (m_1, m_2) \in e_1 \times e_2 \} \). For each \((m_1, m_2) \in M(e_1, e_2) \), a feature vector \( \phi_{m_1, m_2} \) is computed. Then, every feature in \( \phi_{m_1, m_2} \) is embedded as a vector in the real space. Let \( \phi_{m_1, m_2} \) denote the concatenation of embeddings of all features in \( \phi_{m_1, m_2} \). Embeddings of all features except the words are learned in the training process. Word embeddings are pre-trained. \( \phi_{m_1, m_2} \) includes the following language independent features:

String match: whether \( m_1 \) is a substring or exact match of \( m_1 \) and vice versa (e.g. \( m_1 = \text{Barack Obama} \) and \( m_2 = \text{Obama} \)).

Distance: word distance and sentence distance between \( m_1 \) and \( m_2 \) discretized into bins m-type: concatenation of m-types for \( m_1 \) and \( m_2 \); e-type: concatenation of e-types for \( m_1 \) and \( m_2 \). Acronym: whether \( m_1 \) is an acronym of \( m_2 \) or vice versa (e.g. \( m_1 = \text{United States} \) and \( m_2 = \text{US} \)).

First name mismatch: whether \( m_1 \) and \( m_2 \) belong to e-type PERSON with the same last name but different first name (e.g. \( m_1 = \text{Barack Obama} \) and \( m_2 = \text{Michelle Obama} \)).

Speaker detection: whether \( m_1 \) and \( m_2 \) both occur in the context of words indicating speech e.g. “say”, “said”. In addition, \( \phi_{m_1, m_2} \) includes the average of the word embeddings of \( m_1 \) and average of the word embeddings of \( m_2 \).

2.1 Network Architecture

The network architecture from the input to the output is shown in figure 1.

Embedding Layer: For each training triplet \((e_1, e_2, y_{12})\), a sequence of vectors \( \phi_{m_1, m_2} \) for each \((m_1, m_2) \in M(e_1, e_2) \) is given as input to the network.

Relu Layer: \( \phi_{m_1, m_2} = \text{max}(\text{ReLU}(\phi_{m_1, m_2})) \)

Attention Layer: To generate the entity-pair embedding, we need to combine the embeddings of mention pairs generated from the entity-pair. Consider two entities \( e_1 \) (President) and \( e_2 \) (Obama). There are superscripts used to indicate different two mentions with the same surface form. Since the named mention pair (Obama, Clinton) has no string overlap, \( e_1 \) and \( e_2 \) should not be co-referenced even though the nominal mention pair (President1, President2) has full string overlap. So, while combining the embeddings for the mention pairs, mention pairs with m-type (name, name) should get higher weight than mention pairs with m-type (nominal, nominal). The entity pair embedding is the weighted sum of the mention-pair embeddings. We introduce 4 parameters \( \phi_{\text{mention, name}}, \phi_{\text{mention, nominal}}, \phi_{\text{nominal, name}}, \phi_{\text{nominal, nominal}} \) as weights for mention pair embeddings with m-types of (name, name), (name, nominal), (nominal, nominal), (nominal, name) respectively. The entity pair embedding is computed as follows:

\[
P(y_{12} = 1 | e_1, e_2) = \frac{1}{1 + e^{-\alpha y_{12}}} \]

The training objective is to maximize \( L \).

\[
L = \prod_{d \in D} \prod_{e_1, e_2 \in \mathcal{E}(d)} P(y_{12} = 1 | e_1, e_2, W^{(1)}, W^{(2)}, a, w)\]

Here \( D \) is the corpus and \( \mathcal{E} \) is the training triplets generated from document \( d \).

Decoding proceeds similarly to training algorithm, except at each of the three steps, for each entity \( e_i \), the highest scoring antecedent \( e_j \) is selected and if the score is above a threshold, \( e_i \) and \( e_j \) are merged.

3 A Zero-shot Entity Linking model

We use our recently proposed cross-lingual EL model, described in (Sil et al., 2018), where our target is to perform zero-shot learning (Socher et al., 2013; Palatucci et al., 2009). We train an EL model on English and use it to decode on any other language, provided that we have access to multi-lingual embeddings from English and the target language. We briefly describe our techniques here and direct the interested readers to the paper. The EL model computes several similarity/coherence scores \( S \) in a “feature abstraction layer” which computes several measures of similarity between the context of the mention \( m \) in the query document and the context of the candidate link’s Wikipedia page which are fed to a
feed-forward neural layer which acts as a binary classifier to predict the correct link for \( m \). Specifically, the feature abstraction layer computes cosine similarities (Sil and Florian, 2016) between the representations of the source query document and the target Wikipedia pages over various granularities. These representations are computed by performing CNNs and LSTMs over the context of the entities. Then these similarities are fed into a Multi-perspective Binning layer which maps each similarity into a higher dimensional vector. We also train fine-grained similarities and dissimilarities between the query and candidate document from multiple perspectives, combined with convolution and tensor networks.

The model achieves state-of-the-art (SOTA) results on English benchmark EL datasets and also performs surprisingly well on Spanish and Chinese. However, although the EL model is “zero-shot”, the within-document coreference resolution in the system is a language-dependent SOTA coreference system that has won multiple TAC KBP (Ji et al., 2015; Sil et al., 2015) evaluations but is trained on the target language. Hence, our aim is to apply our proposed coreference model to the EL system to perform an extrinsic evaluation of our proposed algorithm.

### 4 Experiments

We evaluate cross-lingual transfer of coreference models on the TAC 2015 Tri-Lingual EL datasets. Our system takes mentions annotated with their grounded Freebase 1 links (if such links exist) or corpus-wide clustering information for 3 languages: English (henceforth, En), Chinese (henceforth, Zh) and Spanish (henceforth, Es). Table 1 shows the size of the training and test sets for the three languages. The documents come from two genres of newswire and discussion forums. The mentions in this dataset are either named entities or nominals that belong to five types: PER, ORG, GPE, LOC and FAC.

#### Hyperparameters

Every feature is embedded in a 50 dimensional space except the words which reside in a 300 dimensional space. The Relu and Sigmoid layers have 100 and 500 neurons respectively. We use SGD for optimization with an initial learning rate of 0.05 which is linearly reduced to 0.0001. Our mini batch size is 32 and we train for 50 epochs and keep the best model based on dev set.

#### Coreference Results

For each language, we followed the official train-test splits made in the TAC 2015 competition. Except, a small portion of the training set is held out as development set for tuning the models. All experimental results on all languages reported in this paper are obtained on the official test sets. We used the official CoNLL 2012 evaluation script and report MUC, B1 and CEAF scores and their average (CoNLL score). See Pradhan et al. (2011, 2012).

To test the competitiveness of our model with other SOTA models, we train the publicly available system of Clark and Manning (2016) (henceforth, C&M16) on the TAC 15 En training set and test on the TAC 15 En test set. The C&M16 system normally outputs both noun phrase mentions and their coreference and is trained on Ontonotes. To ensure a fair comparison, we changed the configuration of the system to accept gold mention boundaries both during training and testing. Since the system was unable to deal with partially overlapping mentions, we excluded such mentions in the evaluation. Table 2 shows that our model outperforms C&M16 by 8 points.

#### For cross-lingual experiments, we build monolingual embeddings for En, Zh and Es. However, most of the recent neural coreference models (Wiseeman et al., 2015, 2016; Clark and Manning, 2015, 2016; Lee et al., 2017) have focused on training and testing on the same language. In contrast, our model performs cross-lingual coreference. There have been some recent promising results regarding such cross-lingual models for other tasks, most notably mention detection (Ni et al., 2017) and EL (Tsai and Roth, 2016; Sil and Florian, 2016). In this work, we show that such promise exists for coreference also.

The tasks of EL and coreference are intrinsically related, prompting joint models (Durrett and Klein, 2014; Hajihasrati et al., 2013). However, the recent SOTA was obtained using pipeline models of coreference and EL (Sil et al., 2018). Compared to a joint model, pipeline models are easier to implement, improve and adapt to a new domain.

### 6 Conclusion

The proposed cross-lingual coreference model was found to be empirically strong in both intrinsic and extrinsic evaluations in the context of an entity linking task.

---

1 TAC uses BaseKB, which is a snapshot of Freebase. Sti18 links entities to Wikipedia and in-turn links them to BaseKB.

Table 1: No of documents for the TAC 2015 Tri-Lingual EL Dataset

<table>
<thead>
<tr>
<th>Language</th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>En</td>
<td>168</td>
<td>129</td>
</tr>
<tr>
<td>En</td>
<td>167</td>
<td>167</td>
</tr>
<tr>
<td>Zh</td>
<td>147</td>
<td>166</td>
</tr>
</tbody>
</table>

Table 2: Coreference results on the En test set of TAC 15 competition. Our model significantly outperforms C&M16.

<table>
<thead>
<tr>
<th>Model</th>
<th>MUC</th>
<th>B1</th>
<th>CEAF</th>
<th>CoNLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>En Model</td>
<td>95.9</td>
<td>92.1</td>
<td>87.2</td>
<td>89.3</td>
</tr>
<tr>
<td>En Model</td>
<td>91.6</td>
<td>89.1</td>
<td>84.8</td>
<td>89.8</td>
</tr>
<tr>
<td>Zh Test Set</td>
<td>90.1</td>
<td>85.3</td>
<td>81.2</td>
<td>85.3</td>
</tr>
<tr>
<td>Zh Test Set</td>
<td>90.1</td>
<td>85.3</td>
<td>81.2</td>
<td>85.3</td>
</tr>
</tbody>
</table>

Table 3: Coreference results on the En and Zh test sets of TAC 15. En model performs competitively to the models trained on target language data.

Table 4: Performance comparison on the TAC 2015 En and Zh datasets. EL + En Coref outperforms the best 2015 TAC system (Rank 1) without requiring any Es or Zh coreference data.

<table>
<thead>
<tr>
<th>System</th>
<th>Train on</th>
<th>Test on</th>
<th>Acc. on</th>
<th>Acc. on</th>
</tr>
</thead>
<tbody>
<tr>
<td>EL + Coref</td>
<td>No</td>
<td>No</td>
<td>78.1</td>
<td>81.3</td>
</tr>
<tr>
<td>EL + En Coref</td>
<td>No</td>
<td>No</td>
<td>88.1</td>
<td>88.1</td>
</tr>
<tr>
<td>TAC Rank 1</td>
<td>Yes</td>
<td>Yes</td>
<td>80.4</td>
<td>83.1</td>
</tr>
<tr>
<td>Sti18</td>
<td>Yes</td>
<td>Yes</td>
<td>82.3</td>
<td>84.4</td>
</tr>
</tbody>
</table>
References


Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In EMNLP.

Anders Björkelund and Jonas Kahn. 2014. Learning structured perceptions for coreference resolution with latent antecedents and non-local features. In ACL.

Kevin Clark and Christopher D Manning. 2015. Entity-centric coreference resolution with model stacking. In ACL.

Kevin Clark and Christopher D Manning. 2016. Improving coreference resolution by learning entity-level distributed representations. In ACL.

Greg Durrett, David Leo Wright Hall, and Dan Klein. 2013. Decentralized entity-level modeling for coreference resolution. In ACL.


Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In EACL.

Enzaldo Rezende Fernandes, Cezar Nogueira Dos Santos, and Ruy Luiz Milididi. 2012. Latent structure perception with feature induction for unrestricted coreference resolution. In EMNLP-CoNLL.


Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of tac-kbp2015 tri-lingual entity discovery and linking. In TAC.


Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In ACL.


Karthik Raghunathan, Heyoung Lee, Sudarshan Ranganathan, Nathaniel Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In EMNLP.

Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In EMNLP.

Avirup Sil, Georgiana Dinu, and Radu Florian. 2015. The ibm systems for trilingual entity discovery and linking at tac 2015. In TAC.

Avirup Sil and Radu Florian. 2016. One for all: Towards language independent named entity linking. In ACL.

Avirup Sil, Gourab Kundu, Radu Florian, and Wael Hamza. 2018. Neural cross-lingual entity linking. In AAAI.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In NIPS.

Chen-Tse Tsai and Dan Roth. 2016. Cross-lingual wiki-linkification using multilingual embeddings. In ICLR. NAACL.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. 2016. Cross-lingual models of word embeddings: An empirical comparison. In ACL.


Sam Joshua Wiseman, Alexander Matthew Rush, Stuart Merrill Shieber, and Jason Weston. 2015. Learning anaphoricty and antecedent ranking features for coreference resolution. In ACL.
Exploiting Structure in Representation of Named Entities using Active Learning

Nikita Bhutani
University of Michigan, Ann Arbor IBM Research - Almaden
nbhutani@umich.edu

Kun Qian
IBM Research - Almaden
qian.kun@ibm.com

Yunyao Li
IBM Research - Almaden
yunyaoli@us.ibm.com

H. V. Jagadish
University of Michigan, Ann Arbor IBM Research - Almaden
jag@umich.edu

Mauricio A. Hernandez
IBM Research - Almaden
mahernan@us.ibm.com

Mitesh Vasa
IBM Research - Almaden
mitesh.vasa@us.ibm.com

Abstract

Fundamental to several knowledge-centric applications is the need to identify named entities from their textual mentions. However, entities lack a unique representation and their mentions can differ greatly. These variations arise in complex ways that cannot be captured using textual similarity metrics. However, entities have underlying structures, typically shared by entities of the same entity type, that can help reason over their name variations. Discovering, learning and manipulating these structures typically requires high manual effort in the form of large amounts of labeled training data and handwritten transformation programs. In this work, we propose an active-learning based framework that drastically reduces the labeled data required to learn the structures of entities. We show that programs for mapping entity mentions to their structures can be automatically generated using human-comprehensible labels. Our experiments show that our framework consistently outperforms both handwritten programs and supervised learning models. We also demonstrate the utility of our framework in relation extraction and entity resolution tasks.

1 Introduction

Named entities are atomic objects of reference and reasoning in many cognitive applications and knowledge-centric services like deep question answering, text summarization and analytics. A real-world entity may have a great variety of representations (Galarraga et al., 2014; Nakashole et al., 2011). For example, University of California, Santa Cruz could have different string representations or name variations: UCSC, UC Santa Cruz, UC–Santa Cruz. Determining if two representations refer to the same entity is an important primitive in entity resolution and entity linking algorithms that drive these knowledge-centric applications (Shen et al., 2015; Arasu and Kaushik, 2009).

Unifying entity representations has been widely studied in record linkage (Christen, 2012), deduplication (Elmagarmid et al., 2007) and reference matching (McCallum et al., 2000). Typically, duplicates are identified using the attributes and/or the contextual information of entities (Zhang et al., 2010; Han et al., 2011; Shen et al., 2015). The string representation or mention of an entity forms key evidence: similar mentions likely refer to the same entity. Although deemed as crucial (Dredze et al., 2010), variations in mentions are typically handled using textual similarity like edit distance and cosine similarity (Zheng et al., 2010; Lehmann et al., 2010; Liu et al., 2013), which can be misleading.

Example. Consider the mentions: (a) General Electric Corporation, (b) General Electric China Corporation, (c) GE Corp. Mentions (a) and (b) are textually similar, differing in just one token. However, they refer to different entities, owing to the location detail ‘China’. Conversely, textually dissimilar mentions (a) and (c) refer to the same entity.

An entity mention is not merely a sequence of characters (Arasu and Kaushik, 2009). It instead has an internal structure, specific to the type of entity. For example in Table 1, the company mentions have a %name%, optionally followed by %loc% and %suffix%. Such structural interpretation can help design similarity...
functions to capture the nuances in name variations of an entity that string similarity functions cannot. For instance, GE can be explained as a mention of General Electric Corporation using transformations like abbreviate \( \langle \text{name} \rangle \) and drop \( \langle \text{suffix} \rangle \). These transformations coupled with string similarity functions can augment existing entity linking algorithms (Qian et al., 2017).

The name variations and, consequently, the transformations are highly domain-dependent (Arasu et al., 2008). Designing similarity functions for an entity type that can reason over the structure of entities requires: (a) a comprehensive list of the structured representations (e.g. column 2 in Table 1), and (b) programs that can map mentions to these structures (e.g. column 3 in Table 1). Traditionally, a domain expert would scan a list of mentions of a target entity type to identify the different structured representations and handwrite programs. This requires specialized skills, and is error-prone and expensive, taking up to several person months to tune the programs for a single application (Campos et al., 2015).

To alleviate the high manual effort, some of the prior works (Arasu and Kaushik, 2009) use declarative, programmable frameworks that allow an expert to directly manipulate the mentions. The expert can provide a program as a set of grammar rules to generate the structured representations. While this equips the expert to manipulate the structure of a mention, it is only a partial solution. The rules are not generic, requiring the expert to specify how each mention is parsed to its structure. This is wasteful because a structure, shared by several mentions, can be captured with a generic rule. Another limitation of previous approaches is that they do not handle structural ambiguities. For instance, there can be multiple structures of Apple Inc. such as \( \langle \text{name} \rangle \text{loc} \) and \( \langle \text{name} \rangle \text{suffix} \), only one of which is correct.

Since several mentions of an entity type tend to have similar structures, it is possible to learn the various structures from a subset of mentions. Given a good query strategy, active learning offers a promising approach to efficiently select a small set of such mentions. Moreover, the expert need not provide programs that parse the selected mentions and generalize to unseen mentions. These programs can be induced from the labels for the structures of the mentions. Embodying these ideas, we propose LUSTRE, an entity learning based framework that learns structured representations for an entity type from human-comprehensible labels for a small set of mentions. It automatically synthesizes generalizable programs from the labels to map new mentions of the entity type to the learned structured representations. In addition, it allows the expert to incorporate domain knowledge and additional feedback to handle structural ambiguities. Our framework significantly reduces the manual effort in labeling mentions and writing programs for the structured representations. We also demonstrate how these structured representations help define similarity functions that benefit entity resolution, and string transformation functions that benefit relation extraction. The intellectual contributions of this work are as follows:

- **Structured Representations**: We present a framework to reason about name variations of entities based on their structured representations.
- **Active-learning**: We present an active-learning approach and a unified query strategy to learn structured representations for a target entity type with minimal human input.
- **Program Synthesis**: We propose to automatically synthesize generalizable programs from human-understandable labels to map mentions to their structured representations.
- **Experimental Evaluation**: Our experiments show that our framework achieves an average of 92% precision and 86% recall in predicting structured representations for several entity types, outperforming competing approaches that require high manual effort. We demonstrate the usefulness of the structured representations in two important tasks: entity resolution and relation extraction.

---

### Table 1: Example Structured Representations of Company and University Mentions

<table>
<thead>
<tr>
<th>Unlabeled Mention</th>
<th>Labeled Mention</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM United Kingdom Ltd.</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle \langle \text{suffix} \rangle )</td>
</tr>
<tr>
<td>Alibaba USA</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle \langle \text{suffix} \rangle )</td>
</tr>
<tr>
<td>Barclays</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle )</td>
</tr>
<tr>
<td>Hewlett-Packard Co.</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle \langle \text{suffix} \rangle )</td>
</tr>
<tr>
<td>University of California, Irvine</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle \langle \text{suffix} \rangle )</td>
</tr>
<tr>
<td>UM-Ann Arbor</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle \langle \text{suffix} \rangle )</td>
</tr>
<tr>
<td>Stanford University</td>
<td>( \langle \text{name} \rangle \langle \text{loc} \rangle \langle \text{suffix} \rangle )</td>
</tr>
</tbody>
</table>

---

### 2 Problem Formalization and Notations

Given a set of unlabeled mentions (i.e., raw strings) of a target entity type \( \epsilon \), our goal is to actively learn a high-quality model \( \mathcal{M} \) (with a small number of user labels) that can map a new mention of type \( \epsilon \) to its structured representation. Key technical challenges in learning such a model include: (1) designing an effective query strategy to representative mentions that are structurally similar to several other mentions and to find mentions with diverse structures; (2) automatically inferring mapping programs from the user labels to reduce the human effort, as in (Campos et al., 2015) and (Arasu and Kaushik, 2009); (3) handling ambiguities when an unlabeled mention has multiple candidate structured representations.

Mentions of a target entity type have internal structured representations consisting of atomic semantic units. Given the labels for the semantic units for a small set of mentions, we want to learn a model \( \mathcal{M} \) of mapping rules. Each mapping rule converts a raw string mention into a structured representation.

**Definition 1 (Structured Representation)**: A structured representation \( \mathcal{S} \) of an entity mention is a sequence of atomic semantic units that compose its structure.

**Definition 2 (Semantic Unit)**: A semantic unit is a tuple \( \langle l, p \rangle \) where \( l \) is a label for the unit and \( p \) is a pattern matching function (e.g., a regular expression), referred to as a matcher.

A matcher describes how a substring in a mention matches a semantic unit. For instance, a matcher \( \langle [A-Z][a-z]*[0-9]? \rangle \) captures \( \langle \text{name} \rangle \) in mentions Apple Inc and GE Corp. Name variations of an entity can be explained as transformations on its semantic units (e.g. dropping suffix in Apple Inc. generates Apple).

**Definition 3 (Mapping Rule)**: The mapping rule \( r_{\epsilon} \) for a structured representation \( \mathcal{S} \) consists of matches of identifiers which decide how an unlabeled mention is mapped to \( \mathcal{S} \), resulting in a labeled entity mention.

Specifically, a mapping rule \( r_{\epsilon} \) is a sequence of matches in the semantic units in \( \mathcal{S} \), denoted by \( r_{\epsilon} = \{ p_{i} \mid (p_{i}, p_{i+1}) \in \mathcal{S} \} \). This mapping rule constitutes a program that maps unseen mentions to \( \mathcal{S} \). Together the various mapping rules constitute a model \( \mathcal{M} \) for the entity type \( \epsilon \).

### 3 The LUSTRE System

We propose LUSTRE that addresses the aforementioned challenges in learning a model of mapping rules for an entity type by:

- adopting a unified query strategy that combines uncertainty sampling (i.e., selecting a mention whose current structured representation is unknown or uncertain) and density-weighted sampling (i.e., selecting a mention whose structured representation is representative of many unlabeled mentions);
- seeking human-comprehensible labels for the semantic units in the structure of an unlabeled mention, and deducing a mapping rule by combining the matches for the semantic units.
- handling structural ambiguities of an unlabeled mention by ranking the candidate mapping rules based on their reliabilities and additional user feedback.

Figure 1 depicts the workflow of LUSTRE and Algorithm 1 shows our learning algorithm. It takes as input unlabeled mentions \( U_{\epsilon} \) and optionally dictionaries \( D_{\epsilon} \) of a target entity type \( \epsilon \). These dictionaries and a set of pre-defined matches form the building blocks for the mapping rules. Before training, LUSTRE evaluates all the matches against \( U_{\epsilon} \) to inform the query strategy and rule generation (Sec 3.1).

---

### Figure 1: LUSTRE Architecture
Algorithm 1 LUSTRE learning algorithm

input: \(U\) = a pool of unlabeled mentions \(m\)
output: \(M\) = a model of mapping rules \(r\)

1. function \(\text{train}(U)\)
2. \(L\) = a set of labeled mentions \(\{(m, l)\}\)
3. \(\mathcal{F} = \text{feedback}([m, l, f] | m \in U, f \in [0, 1])\)
4. \(M = \emptyset\)
5. for \(t = 1, 2, \ldots\) do
6. \(M = \text{UPDATE}(M, L, \mathcal{F}\)
7. select \(m^*\) = \(L\) mention with highest utility
8. select \(\mathcal{F}\) with least confident predictions
9. query label \(l^*\) for mention \(m^*\)
10. query binary feedback on \(\mathcal{F}\)
11. \(L = L \cup \{m^*, l^*\}\)
12. return \(M\)
13. function \(\text{UPDATE}(M, L, \mathcal{F}\)
14. for \((m, l) \in L\) do
15. \(r_{M} = \text{generate rule}(l)\)
16. if \(r_{M} \notin M\) then
17. \(P_{ao} = \text{reliability}(r_{M})\)
18. \(M = M \cup \{r_{M}, P_{ao}\}\)
19. for \((r_{M}, P_{ao}) \in M\) do
20. \(P_{ao} = \text{estimate}(P_{ao})\)
21. return \(M\)

3.1 Indexing
User-provided dictionaries and pre-defined regular expressions constitute the vocabulary of matchers for the mapping rules. The dictionaries help capture key domain-specific terminology but need not correspond to semantic units.


Additionally, we use type-independent regex matchers shown in Table 2. In a pre-processing step (Line 1 of Algorithm 1), we evaluate the unlabeled mentions against the matchers to save computational overhead. We refer to these enriched

<table>
<thead>
<tr>
<th>Name</th>
<th>Regex</th>
</tr>
</thead>
<tbody>
<tr>
<td>caps</td>
<td>([A-Z][A-Za-z]+)</td>
</tr>
<tr>
<td>num</td>
<td>([0-9]+)</td>
</tr>
<tr>
<td>special</td>
<td>([A-Za-z0-9]+)</td>
</tr>
<tr>
<td>wild</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 2: Predefined Matchers

3.2 Candidate Selection
The query strategy to determine what constitutes an informative mention is the central challenge in our active-learning setting (Line 7 of Algorithm 1). We consider a mention informative if its representations consist of several other unlabeled mentions and its current structure is unknown or uncertain. We adopt a unified approach, combining density-weighted sampling (Settles and Craven, 2008) and uncertainty sampling (Cutlotta and McCallum, 2005) as it is robust to outliers and input distribution.

For each unlabeled mention \(m_u\), we compute a correlation score \(c_i\) (based on its structural similarity to other mentions) and an uncertainty score \(f_i\) (based on its predicted structure). We then compute a utility score \(u_i\) combining its correlation and uncertainty scores, and select a mention \(m^*\) with the highest utility score for labeling.

To compute correlation score \(c_i\), we need a reliable metric to measure the similarity of the structures of two mentions. Since surface-string similarity metrics won’t suffice, we estimate structural similarity of two mentions as a function of the matchers that constitute their structures. Specifically, we compute structural similarity \(c_i(l, j)\) of a pair of mentions \((l, j)\) as the edit distance of their structures, \(S_l\) and \(S_j\).

Example. ‘IBM Ltd.’ and ‘Apple Inc.’ have the same structure \(\langle\text{caps}\rangle\langle\text{suffix}\rangle\), and therefore, have an edit distance of 0. Each has edit distance 1 to ‘Microsoft Ltd. Inc.’ with structure \(\langle\text{caps}\rangle\langle\text{num}\rangle\).

Given the pair-wise structural similarity metric, the correlation score \(c_i\) of a mention \(m\) is its average structural similarity to other unlabeled mentions.

\[c_i = \frac{1}{|L|} \sum_{j \in L} c_i(l, j)\]

if no rule maps \(m\)

\[f_i = \frac{1}{|L^*|} \sum_{j \in L^*} f_i(l, j)\]

otherwise

To estimate the uncertainty score \(f_i\) of a mention, we use \(P_{ao}\), the reliability of the mapping rules that can parse the mention. If no mapping rule can parse a mention \(m\), its structure is unknown i.e. \(f_i\) is simply 1. Otherwise, it is the uncertainty of the most reliable rule \(r_{M}\) known to parse the mention. We discuss how the reliability of \(P_{ao}\) of mapping rules are estimated in Sec. 3.4.

3.3 Rule Generation
Once the user labels a selected mention, LUSTRE next has to synthesize a generic program i.e. a mapping rule for the structure of the mention (Line 15 of Algorithm 1). Deriving a mapping rule is non-trivial as semantic units in the structure can potentially span multiple tokens and matchers. As a result, there are many ways to combine matchers and derive the rule.

Example. A user can label ‘General Motors’ as semantic unit (\(\langle\text{name}\rangle\) in ‘General Motors Co.’. Tokens ‘General’ and ‘Motors’ map to matchers \(\text{caps}\) and \(\alpha\text{Num}\). The most reliable interpretation for \(\langle\text{name}\rangle\) is \(\langle\alpha\text{Num}\text{spec}\{1, 3\}\rangle\), as a combination of two adjacent matchers \(\langle\text{caps}\rangle\langle\alpha\text{Num}\rangle\).

LUSTRE derives a reliable rule as the sequence of most selective matchers, where selectivity is the expected number of matches of a matcher over the set of unlabeled mentions \(U\) (Li et al., 2008).

\[s(e[p]) = \{m | m \in U\}\]

with \(match(p, m)\) being number of matches of \(p\) over mention \(m\).

3.4 Parsing
When a new mapping rule is learned, we want to estimate its reliability in predicting structures of mentions, and update the model \(M\) (Line 17-18 of Algorithm 1). These reliability scores are used for estimating utility scores at candidate selection and for resolving ambiguities when multiple rules can parse a mention (with preference given to the most reliable rule). Following the intuition that generic rules are less reliable, we estimate reliability of a rule based on its expected numbers of matches in the unlabeled mentions. Specifically, reliability \(P_{ao}\) in a function of the selectivity of the matchers in \(r_{M}\).

\[P_{ao} = \sum_{p \in s(e[p])} \{m | m \in U\}\]

Table 2: Predefined Matchers

1. The order of preference can be inferred in a pre-processing step in which the system counts the selectivity of each matcher against the input mentions.
with $\alpha$ being the decay constant. We found such estimation, though simple, was effective in estimating the reliability of mapping rules. Training more powerful measures such as a discriminative model would require larger user feedback than is available in this setting.

4 Experiments

We assess the effectiveness of our learning algorithm, and the quality and usefulness of learned structures.\(^2\)

4.1 Experimental Setup

Datasets. We conduct experiments on four entity types of varied complexity and ambiguity.
- Person: Focusing on subtype Individual, we randomly select 200 unique mentions for training and another 200 mentions for testing from the ACE 2005 dataset (Walker et al., 2006). For out-of-domain tests, we randomly select 200 person mentions from Freebase (Bollacker et al., 2008).
- Company: Focusing on subtype Commercial, we randomly select 200 mentions for training and use the remaining 100 mentions for testing from the ACE dataset. For out-of-domain test, we randomly select 200 company mentions from Freebase.
- Tournament: We use a 50/50 split for the 100 unique mentions of tournaments in Freebase.
- Academic Title: We use a 50/50 split for the 350 unique mentions of academic titles in Freebase.

For each mention, we ask two experts to manually annotate every token with a semantic label to produce the ground truth. The average inter-annotator agreement was 0.89 for Cohen’s $\kappa$.

Baselines. We compare LUSTRE with the following methods.
- STG: A commercial system (Campos et al., 2015) which requires an expert (with domain knowledge and programming skills) to analyze the structures of an entity type and handwrite mapping programs.
- Linear-chain CRF\(^3\): A linear-chain CRF model that predicts a sequence of labels for the tokens in a mention. We use matches from the indexing stage as features. We use two different training settings: CRF trained on the entire training set (to compare with best model), and CRF\(^4\) trained on the subset of training set selected via the query strategy in LUSTRE (to compare with same user effort).
- LUSTRE\(^5\): LUSTRE with a native tf-idf based query strategy.

Evaluation Metric. We consider a prediction correct if the model and expert agree on the semantic labels for each token in the mention. We measure precision as the fraction of predictions that are correct and recall as the fraction of correct structures that are predicted. We define a new metric, which we call the $\alpha$ value, to estimate the role of manual effort on performance of various methods (definition is given below). Intuitively, higher the $\alpha$ value, higher the effectiveness of a method in learning from a user label.

\[ \alpha(X, t) = \frac{F\text{-score of method } X \text{ on entity type } t}{\text{number of user labels requested by } X} \]

\(^5\)We will release the code and data from this work. Code is proprietary but can be licensed.

4.2 Quality Analysis

The performance results of the different methods are summarized in Table 3. For Person mentions, all methods achieve reasonably good results ($F$-scores are all above 0.85). This is not surprising as person mentions typically have simple structures with few semantic units. STG achieves lowest performance, indicating that manually-crafted programs are not as robust as learned models. CRF\(^5\) models, especially CRF\(^4\) trained on mentions selected using LUSTRE, show evident improvement over STG. LUSTRE achieves comparable performance. However, it outperforms all other methods on out-of-domain test data, suggesting it can capture the structures in mentions regardless of their data source.

Company mentions are more complex than Person mentions. They can have several semantic units such as core name, location, suffix, and subsidiary, which can appear in different orders, be separated by special symbols etc. Consequently, capturing structures in company mentions is more difficult, as is reflected in the performance across methods. Learning a reliable model/program would require higher manual effort in STG and more training data in CRF. In contrast, LUSTRE, with small human input, achieves high precision and recall for both in-domain and out-of-domain data.

Tournament and Academic Title mentions have even more complex structures that exhibit more variations. Even for these complex types, LUSTRE outperforms CRF. We do not report results of STG for these two entity types because these were not considered when STG was developed. We do not provide an out-of-domain evaluation of the two entity types because only the Freebase data included mentions of these entity types.

In summary, LUSTRE outperforms STG and CRF-based methods in terms of overall $F$-score for the exception of Person where CRF\(^5\) has a small improvement over LUSTRE. Furthermore, its unified query strategy is more effective than a tf-idf based strategy, as is reflected in the performance of LUSTRE\(^5\).

We found two main sources of errors made by LUSTRE. First, akin to all learning methods (such as CRF), it has low recall when the training data is not representative. Second, ranking mapping rules sometimes cannot effectively resolve structural ambiguities. For example, ‘The Stanford University Professor in Nephrology’ can be interpreted as ‘[honorific prefix] [title] [specialty’ or ‘[institute] [title] [specialty’]. The former is incorrect, but being more commonly observed in the training data, is ranked higher.

4.3 Effectiveness

To assess how the quality of structured representations evolves in the learning process, we examine the precision and recall of LUSTRE after each iteration (as shown in Figure 2). We found that the precision remains nearly constant. There are a limited number of structures for the same types of entities. Every time LUSTRE learns a precise rule that improves coverage. There are slight drops in the precision due to the long tail of non-representative cases. The recall generally increases depending on whether the system prefers to gather additional evidence or to discover new structured representations.

We also examine the number of rules learned, number of incorrect intermediate predictions, and percentage of input training data covered after each learning iteration in LUSTRE (as shown in Figure 3). We found that it took 8-13 iterations to learn almost all different structures of an entity type. The fraction of training data that could be parsed using the mapping rules also generally increased, indicating that our query strategy selected structurally diverse mentions for labeling. Only a few (<5) predictions were

---

\(^2\)http://mallet.cs.umass.edu/sequences.php

---

Table 3: Performances of LUSTRE, STG and CRF

<table>
<thead>
<tr>
<th>Type</th>
<th>In-Domain</th>
<th>Out-of-Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STG</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>CRF(^5)</td>
<td>0.97</td>
<td>0.97</td>
</tr>
<tr>
<td>CRF(^4)</td>
<td>0.99</td>
<td>0.99</td>
</tr>
<tr>
<td>LUSTRE(^5)</td>
<td>0.98</td>
<td>0.96</td>
</tr>
<tr>
<td>LUSTRE</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>Company</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STG</td>
<td>0.83</td>
<td>0.83</td>
</tr>
<tr>
<td>CRF(^5)</td>
<td>0.87</td>
<td>0.87</td>
</tr>
<tr>
<td>CRF(^4)</td>
<td>0.81</td>
<td>0.81</td>
</tr>
<tr>
<td>LUSTRE(^5)</td>
<td>0.84</td>
<td>0.78</td>
</tr>
<tr>
<td>LUSTRE</td>
<td>0.95</td>
<td>0.90</td>
</tr>
<tr>
<td>Tournament</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRF(^5)</td>
<td>0.70</td>
<td>0.70</td>
</tr>
<tr>
<td>LUSTRE(^5)</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>LUSTRE</td>
<td>0.96</td>
<td>0.90</td>
</tr>
<tr>
<td>Title</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRF(^5)</td>
<td>0.69</td>
<td>0.69</td>
</tr>
<tr>
<td>LUSTRE(^5)</td>
<td>0.67</td>
<td>0.67</td>
</tr>
<tr>
<td>LUSTRE</td>
<td>0.36</td>
<td>0.23</td>
</tr>
<tr>
<td>LUSTRE</td>
<td>0.79</td>
<td>0.65</td>
</tr>
</tbody>
</table>

---

Figure 2: Performance of LUSTRE over iterations

---

Figure 3: Performance of LUSTRE over iterations
Emp-Social given the extreme low matching ratio for this dataset. ERLearn-CIKM already identifies a significant subset of the true links in such a sparse space that it is non-trivial for ERLearn-LUSTRE to have found additional 45 true links. In contrast, the ER task is more challenging for the Crystal scenario, with more matching functions (158 vs. 68) (Qian et al., 2017). ERLearn-LUSTRE could still identify additional 1544 true links, which is a significant improvement over 130k true links already identified by ERLearn-CIKM.5

Relation Extraction
We use MULTIR (Hoffmann et al., 2011), a state-of-the-art relation extractor trained on NY Times text (Riedel et al., 2010) with weak supervision from Freebase (Bollacker et al., 2008). The weak supervision data is generated by exactly matching the textual mentions to canonicalized entities in Freebase. We instead match to the variations of entities of types Person and Company, generated using the configurations in Table 6. We follow the approach of (Hoffmann et al., 2011) to generate supervision data, compute features and evaluate aggregate extraction.

For the 2 million entities, we generate 5.3 million variations. There were 24,882 sentences where textual mentions exactly matched a canonical Freebase entity and one of the specified relations existed between the entities. This increased to 34,197 sentences when named variations of entities were included, suggesting name variations are useful for entity recognition. By including the variations for only two entity types, we could generate a training data that improved the overall extractor performance (F-1 score increased by 3% from 0.485 to 0.499). The extractor could further benefit from variations of entities of other types.

5 Related Work
Exploiting the compositional structure of entities and attributes especially from query streams and text has received much attention in databases and NLP literature. It has mostly been used for understanding NL questions (Berant et al., 2013), noun-phrase queries (Li, 2010) or normalizing time expressions (Lee et al., 2014; Bethard, 2013). Consequently, structuring and linking information on the web (Bollacker et al., 2008; Auer et al., 2007) about entities and their attributes, has seen some interest. With this information being automatically extracted from textual data (Fader et al., 2011; Carlson et al., 2010), reconciling variations in entities and attribute names has become an integral part of the effort. Some recent work (Halevy et al., 2016) has attempted to organize attribute names by learning their compositional structure. On the other hand, some have proposed complex normalization frameworks (D’Souza and Ng, 2015) for specific domains. However, we need methods that can learn structured representations for the large scale of entity types found on the web.

Named entities are not atomic units and often contain other entities (Finkel and Manning, 2009). However, entity resolution has relied largely on surface-level match of entity mentions (Riedel et al., 2010; Hoffmann et al., 2011; Xu et al., 2013). Variations are typically handled using similarity functions such as edit distance, jaccard similarity, which have limited customizability. While learning string transformation rules (Araujo et al., 2009; Singh and Gulwani, 2012) to reconcile variations has been studied in different contexts, it typically relies on a set of input-output examples. It is difficult to obtain such data for entities and their variations. We instead propose a different approach to first learn the internal structure of an entity and then enable configurable transformations on its structure to generate its variations.

We focus on the problem of reducing manual effort in selecting a set of representative examples for learning the regular expression patterns, which is different from the problem of synthesizing regular expressions.
expressions from examples (Bartoli et al., 2012; Li et al., 2008).

6 Conclusion
This paper identifies a novel problem of understanding structured representations of entities for handling their name variations. We propose an active-learning based approach, LUSTRE, to iteratively learn the structured representations of an entity type from a few labeled mentions and a large set of unlabeled mentions. With small manual effort, it can learn these structured representations and automatically generate programs to map mentions to their structured representations. Reasoning over such structured representations is useful for entity resolution and relation extraction that require reasoning over name variations of entities. In the future, we plan to extend our approach to learn structures of nested entities, and use sophisticated variant generation algorithms that could rank the variations based on their reliability.

Acknowledgements
This work was supported in part by IBM under contract 4915012629 and a graduate fellowship, and by a grant from the UM Office of Research.

References


Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In COLING.


Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.


### 7 Appendix A. Experimental Setup of Entity Resolution

Our entity resolution experiments use ERLearn (Qian et al., 2017) on two scenarios: Emp-Social and Crystal. We aim to provide it supplementary matching functions that reason over structured representations of entity names. To learn structured representations for the Emp-Social scenario, we randomly sample 1000 person names from the Emp records and 1000 person names from the online social network data. Using LUSTRE over the 2000 names, we learn the structures for Person with five different semantic units: \( \text{first} \), \( \text{last} \), \( \text{middle} \), \( \text{suffix} \), and \( \text{location} \). For the Crystal scenario, we learn structures for Company names using a sample of 2000 names. These structures have semantic units \( \text{name} \), \( \text{industry} \), \( \text{suffix} \), and \( \text{location} \).

To design matching functions that reason over structured representations to identify duplicates, we refer to the name variations of mentions generated using their representations. Intuitively, duplicate mentions are likely to share many name variations. Given a mention, we first generate its variations using the configurations in Table 7. We keep both the original string and the transformed string for a semantic unit modified using an operator from the configuration. For instance, dropping the title in a person name would generate many name variations. Given a mention, we first generate its variations using the configurations in Table 7. We keep both the original string and the transformed string for a semantic unit modified using an operator from the configuration. For instance, dropping the title in a person name would generate

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration</th>
<th>Matching Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>( \text{first} ), ( \text{last} ), ( \text{middle} ), ( \text{suffix} ), ( \text{location} )</td>
<td></td>
</tr>
<tr>
<td>Company</td>
<td>( \text{name} ), ( \text{industry} ), ( \text{suffix} ), ( \text{location} )</td>
<td></td>
</tr>
</tbody>
</table>

| Person     | \( \text{first} \), \( \text{last} \), \( \text{middle} \), \( \text{suffix} \), \( \text{location} \) | 
| Company    | \( \text{name} \), \( \text{industry} \), \( \text{suffix} \), \( \text{location} \) |  

Table 7: Configurations with examples

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Matching Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>( \text{first} ), ( \text{last} ), ( \text{middle} ), ( \text{suffix} ), ( \text{location} )</td>
</tr>
<tr>
<td>Company</td>
<td>( \text{name} ), ( \text{industry} ), ( \text{suffix} ), ( \text{location} )</td>
</tr>
</tbody>
</table>

There are to key differences: (1) the matching function for first names \( \text{first} \) uses the same matching function \( \text{matchPerson} \), (2) the threshold value used in \( \text{lastNameFreqFilter} \) decreased from 60% to 50%. In contrast to the original rules, the lower threshold value for last names in the new rule makes it less conservative, potentially increasing the risk of identifying incorrect links. However, the matching function \( \text{matchPerson} \) makes the rule less susceptible to over-generalization. We found that by including matching functions that exploit the structured representations of entities, ERLearn could learn an ER model with appropriate generalization.

#### Rules from ERLearn-CIKM

- \( \text{matchEmp(i, Social s By R):} \)
  - \( \text{matchPerson(i.Name, s.name, r)} \)
  - \( \text{matchCompany(c1.name, cp1.name, r)} \)
  - \( \text{matchCompany(c1.parent.name, cp2.parent.name, r)} \)

where \( r = 1, 2, \ldots, 20 \). These matching functions complement the matching functions used in the original study (Qian et al., 2017). We compare the ER rules learned by ERLearn in two different configurations, ERLearn-LUSTRE and ERLearn-CIKM, with and without the new matching functions respectively. The rules learned by ERLearn-LUSTRE had similar consistent matching functions as rules from ERLearn-CIKM. However, we found they additionally included the new matching functions and had relaxed some of the matching functions. For illustration, consider the following rules:

<table>
<thead>
<tr>
<th>Company</th>
<th>( \text{name} ), ( \text{industry} ), ( \text{suffix} ), ( \text{location} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>( \text{first} ), ( \text{last} ), ( \text{middle} ), ( \text{suffix} ), ( \text{location} )</td>
</tr>
</tbody>
</table>

(a) ERLearn-LUSTRE

(b) ERLearn-CIKM

These are to key differences: (1) the matching function for first names \( \text{first} \) uses the same matching function \( \text{matchPerson} \), (2) the threshold value used in \( \text{lastNameFreqFilter} \) decreased from 60% to 50%. In contrast to the original rules, the lower threshold value for last names in the new rule makes it less conservative, potentially increasing the risk of identifying incorrect links. However, the matching function \( \text{matchPerson} \) makes the rule less susceptible to over-generalization. We found that by including matching functions that exploit the structured representations of entities, ERLearn could learn an ER model with appropriate generalization.
Will it Blend? Blending Weak and Strong Labeled Data in a Neural Network for Argumentation Mining

Eyal Shnarch, Carlos Alzate, Lena Dankin, Martin Gleize, Yufang Hou, Leshem Choshen, Ranit Aharonov, Noam Slonim
IBM Research
{eyals, lenad, leshem.choshen, ranita, noams}@il.ibm.com
{carlos.alzate, martin.gleize, yhou}@ie.ibm.com

Abstract
The process of obtaining high quality labeled data for natural language understanding tasks is often slow, error-prone, complicated and expensive. With the vast usage of neural networks, this issue becomes more notorious since these networks require a large amount of labeled data to produce satisfactory results. We propose a methodology to blend high quality but scarce labeled data with noisy but abundant weak labeled data during the training of neural networks. Experiments in the context of topic-dependent evidence detection with two forms of weak labeled data show the advantages of the blending scheme. In addition, we provide a manually annotated data set for the task of topic-dependent evidence detection.

1 Introduction
In recent years, neural networks have been widely used for natural language understanding tasks. Such networks demand a considerable amount of labeled data for each specific task. However, for many tasks, the process of obtaining high quality labeled data is slow, expensive, and complicated (Habernal et al., 2018). In this work, we propose a method for improving network training when a small amount of labeled data is available.

Several works have suggested methods for generating weak labeled data (WLD) whose quality for the task of interest is low, but that can be easily obtained. One approach for gathering WLD is to apply heuristics to a large corpus. For example, Hearst (1992) considered a noun to be the hypernym of another noun if they are connected by the is a pattern in a sentence.

Distant supervision is another form of WLD used in various tasks such as relation extraction (Mintz et al., 2009; Surdeanu et al., 2012) and sentiment analysis (Go et al., 2009). Other works use emojis or hashtags as weak labels describing the texts in which they appear (e.g., Davidov et al. (2010) in the context of sarcasm detection).

WLD can be freely obtained, however it comes with a price: it is often very noisy. Therefore, systems trained only on WLD are at a serious disadvantage compared to systems trained on high quality labeled data, which we term henceforth strong labeled data (SLD). However, we suggest that the easily accessible WLD is still useful when used alongside SLD, which is naturally limited in size.

In this work we propose a method for blending WLD and SLD in the training of neural networks. Focusing on the argumentation mining field, we create and release a data set for the task of topic-dependent evidence detection. Our evaluation shows that such blending improves the accuracy of the network compared to not using WLD or not blending it. This improvement is even more evident when SLD is not abundantly available.

We believe that blending WLD and SLD is a general notion that may be applicable to many language understanding tasks, and can especially assist researchers who wish to train a network but have a small amount of SLD for their task of interest.

2 Background
2.1 WLD and networks
In the field of neural networks, WLD has mainly been employed for pre-training networks. This was done in related fields such as information retrieval (Dehghani et al., 2017b) and sentiment analysis (Severyn and Moschitti, 2015; Deriu et al., 2017). Contrary to those works, we ex-
plote a way to utilize WLD together with SLD and throughout the training process.

Most similar to our work, Delghani et al. (2017a) use WLD and SLD together, for sentiment classification. They train two separate networks, one with WLD only, and another with SLD only. They control the magnitude of the gradient updates to the network trained on WLD, using the scores provided by the network trained on SLD. Differently, we blend the two types of labeled data in a single network.

2.2 Argumentation mining

Argumentation mining is attracting a lot of attention (Lippi and Torroni, 2016). One line of research focuses on identifying arguments (claims and evidence/premises) within a text (Stab and Gurevych, 2014; Habernal and Gurevych, 2015; Persing and Ng, 2016; Eger et al., 2017). Another line of work seeks to mine arguments relevant for a given topic or claim, either from a pre-built argument repository where arguments are collected from online debate portals (Wachsmuth et al., 2017), or from unstructured large scale corpora (Levy et al., 2014; Rinott et al., 2015; Levy et al., 2017). Our work falls into the latter category of corpus wide topic-dependent argument mining.

Previous work by Rinott et al. (2015) presented the task of detecting evidence texts that are relevant for claims of a given topic. They search in a preselected set of articles, in which the likelihood to find an evidence is considerably higher than in an arbitrary article from the corpus. In this work, we detect evidence directly supporting or contesting the topic (without an intermediate claim), and we search in the entire corpus, with no need for pre-selecting a small set of relevant articles.

2.3 SLD and WLD in argumentation mining

Publicly available strong labeled data (SLD) for argument mining is usually only a couple of thousand instances in size (e.g., Stab and Gurevych (2017) present one of the largest, with around 6,000 annotated positive instances). Recently, Habernal et al. (2018) have commented about the difficulty to collect valuable SLD from crowd sourcing for such tasks.

Several works utilize WLD for argumentation mining, Webis-Debate-16 (Al-Khatib et al., 2016) uses the structure of online debates as distant supervision for the task of argument classification. Sentences from the first paragraph are considered as non-argumentative and the rest of the sentences are considered as argumentative.

For the topic-dependent claim detection task, Levy et al. (2017) showed that retrieving sentences with the word that followed by the concept representing the topic, yields candidates that are more likely to contain a claim for that topic than arbitrary sentences which contain the topic concept.

3 BlendNet

We present BlendNet, a neural network that is trained on a blend of WLD and SLD.

3.1 Network description

Our network is a bi-directional LSTM (Graves and Schmidhuber, 2005) with an additional attention layer (Yang et al., 2016). The models are all trained with a dropout of 0.85, using a single dropout mask across all time-steps as proposed by Gal and Ghahramani (2016). The cell size in the LSTM layers is 128, and the attention layer is of size 100. We use the Adam method as an optimizer (Kingma and Ba, 2015) with a learning rate of 0.001, and apply gradient clipping with a maximum global norm of 1.0. Words are represented using the 300 dimensional GloVe embeddings learned on 840B Common Crawl tokens and are left untouched during training (Pennington et al., 2014).

We note that even though we chose this network architecture, there is nothing in the blending method we propose which is restricted to it, and blending can be easily applied to other networks.

3.2 WLD blending

WLD is a pair of disjoint sets, WLDpos and WLDneg. The two sets are constructed such that the probability of finding positive instances in WLDpos is significantly higher than that of finding them in WLDneg. This difference in probabilities is the source of the signal WLD provides. Importantly, the probability in WLDpos can still be rather low.

As mentioned in Section 2.1, using WLD to pre-train neural networks has been proven to be effective. We extend this idea by allowing the use of WLD alongside SLD during the entire training process of the network. Our intuition is that even though WLD signal is noisy, there is potential in its additional massive amount, and integrating it can improve training when SLD is limited in size.

In every epoch (a pass through the entire SLD), the training data is enriched with WLD. However, since WLD is noisy, an exponentially decreasing fraction of it is blended into the network at each epoch.

Formally, we have $n_{\text{initialization}}$ epochs using the entire WLD with no SLD. After this pre-training phase, we continue with $n$ blending epochs, in each using all the available SLD, and a fraction of the WLD which is determined by a blend factor $\alpha \in [0,1]$. In the $k$th blending epoch ($k \in [0,n_{\text{init}}]$) we blend $\alpha^k$ of the WLD with the SLD, and feed the data in a random order to the network.

Consequently, the first blending epoch uses full SLD and full WLD, and in every subsequent epoch the amount of WLD decays by a factor of $\alpha$. The stopping point $n$ will typically be empirically determined. We set it to a number that will guarantee that the last couple of epochs will be composed of mainly SLD, since eventually, this is the better signal for training.

One can come up with different methods for blending WLD and SLD. For instance, start training with all available SLD and gradually blend more and more WLD, or use all available WLD and SLD during the entire training. In Section 5 we refer to some other alternatives and show that they do not achieve better results than the one presented above. However, we do not claim that our blending method is the only option or even the best one. The goal of this work is to suggest one method which works.

4 Data sets

We created a data set of 5,785 sentences with manual annotations for the task of topic-dependent evidence detection (this will serve as our SLD). It is available on the IBM Debater Datasets webpage.¹ We use it for training and for evaluation and describe it next. In Section 4.2 we describe two methods for freely obtaining weak labeled data for our task.

4.1 SLD annotation

Our strong labeled data (SLD) consists of pairs of a topic and a sentence. Topics were extracted from several sources, such as Debatepedia, an online encyclopedia dedicated to debates and argumentation. The data set includes 118 diverse topics, from domains such as politics, science and education. The topics generally deal with one clearly identifiable concept.

The sentences were extracted from Wikipedia and were annotated by crowd-sourcing. We used 10 annotators for each pair of topic and sentence, each annotator either confirms or denies evidence as evidence for the topic. We combine the annotators’ votes into a binary label by majority. Ties are resolved as non-evidence.

The guidelines for the task present three criteria which all have to be met for a positive label. The sentence must clearly support or contest the topic, and not simply be neutral. It has to be coherent and stand mostly on its own. Finally it has to be convincing, something you could use to sway someone’s stance on the topic: a claim is not enough, it has to be backed up.

The annotators agreement is 0.45 by Fleiss’ kappa. This is a typical value in such challenging labeling tasks, comparable to previous reports in the literature, e.g., (Aharoni et al., 2014; Rinott et al., 2015). In addition, for 85% of the labeled instances, the majority vote included at least 70% of the annotators, further supporting the quality of the released data.

The 118 topics were randomly split into two sets: 83 topics for training (8866 sentences), and 35 topics for testing (1,719 sentences). No sentences of the same topic appear in both sets. The prior for positive, i.e., an evidence instance, is about 40% for both sets. In addition, every occurrence of the topic concept in the candidate is replaced with a common token, to keep the training topic-independent. The topic concept is detected by an in-housewikification tool, similar to TagMe (Ferragina and Sciarrino, 2010), and README, provided with this paper, includes additional information about the data set and the pre-processing.

4.2 WLD generation

Next we describe two sources of WLD we use in our experiments. For the first source, we use the method described by Levy et al. (2017) for unsupervised topic dependent claim detection. Following them, we construct the set of $\text{WLD}_{\text{Levy}}$ by retrieving sentences from Wikipedia which match the query “that + topic concept”, i.e. sentences which contain the word “that” followed by the
concept of the topic (not necessarily adjacent). The WLD\textsubscript{neg} set is constructed by retrieving sentences that contain the topic concept and are not part of WLD\textsubscript{pos}. Levy et al. (2017) showed that the likelihood of claims in WLD\textsubscript{pos} is double the likelihood in WLD\textsubscript{neg}

We believe that the query “that + topic concept” is indicative of argumentative content in general, and not just of claims. It is therefore a good fit for constructing WLD for the topic-dependent evidence detection task. Indeed, in the data set, described in Section 4.1, the prior for positive in the entire training set is close to 40%, but among the candidates that match the query, it is much higher - 52%. Applying this WLD method we were able to extract 253, 352 sentences from Wikipedia which contain the topic concept. 25% of them also contain “that” before the topic concept, and they are our WLD\textsubscript{pos}.

For the second source of WLD, we use the Webis-Debate-16 corpus (Al-Khatib et al., 2016), using their argumentative vs. non-argumentative division. This division was automatically created by mapping the specific structure of idebate.org pages – introduction, points for/against, points/counterpoints – to the two classes. The sentences of the introduction are labeled by them as non-argumentative, under the assumption that they neutrally present the topic. We use them as our WLD\textsubscript{neg}. The other sentences are labeled in Webis-Debate-16 as argumentative, thus we use them as our WLD\textsubscript{pos}. Out of 16, 402 total instances, 66% are in WLD\textsubscript{pos}. This data set doubly deserves the status of WLD in our task because the labels do not exactly match the evidence/non-evidence classification, and in addition it is produced automatically based on a coarse-grained mapping that is bound to introduce noise.

5 Experimental setup and results

We use the data set described in Section 4, training the network on the train set and evaluating its accuracy on the test set. We empirically explore several blending configurations and evaluate their impact on the accuracy of the network. To validate our assumption that WLD contribution would be more prominent when SLD is limited, we test each configuration with varying sizes of SLD between 500 and 4,000.

Following some preliminary exploration, on a different data set, we noticed that the parameter \(\alpha\), the number of initialization epochs, does not make a significant difference, and we set it to be 1 (trying \(m > 1\) resulted in slightly worse accuracy). As mentioned in Section 3.2, our stopping criterion was set to zero in any configuration, we have four blending epochs in which the input for the network is mostly SLD, i.e. it is at least 95% of the data seen by the network.

For the blending factor we tried \(\alpha \in \{0, 0.05, 0.2\}\), and quickly learned that choosing a blending factor value larger than 0.5 is typically ineffective. Since the blending factor determinem the numbers of epochs in which the WLD is significant, and since it is reasonable to limit this number due to the noisy nature of the WLD, it is not surprising that a small value of \(\alpha\) is preferable. We note that setting \(\alpha = 0\) means WLD is only used in the initialization epochs.

Finally, to keep results reliable, as SLD size can get quite small, we repeat each configuration run five times with different SLD slices to reduce variance. For each run we record the best accuracy out of all its epochs and report the micro average of the best accuracies of the five runs.

Figure 1 depicts our results. Blending WLD throughout several epochs of training (the thick green curve with round dots), improves performance over using it only for initialization, as most previous works do (the dashed red curve), and over not using WLD at all (the blue curve with triangles). This effect is significantly more notable as we use less SLD. For example, in the left plot, which presents the usage of Webis-Debate-16 as WLD, we see that using 1,000 instances of SLD with WLD yields results comparable to using 2,500 SLD instances. Similarly, 2,000 SLD instances plus WLD, are comparable to using 3,000 SLD instances. The effect is smaller when the WLD is based on the “that + topic concept” query, but the trend is the same.

One may claim that the signal in WLD is stronger than we hypothesized and therefore the performance improves simply because we are adding labeled data for training. To test this claim we train the network with all available WLD and only it. The single triangles on the Y-axis of each plot show that the accuracy of the network with WLD is much lower than the entire SLD, reflecting the inferior quality of the WLD. In addition, we note that the accuracy on the test set of the “that + topic concept” query, which was used to collect one of our WLD types, is only 17%.

Another claim may be that just by utilizing WLD in addition to SLD the accuracy improves, and that there is no need for any blending method. To answer that, we unify the WLD and the SLD, without applying any blending method (single squares on the right border of each plot). For the WLD constructed by the “that + topic concept” query the accuracy is well below the accuracy achieved when using SLD alone, as can be seen in the right plot. On the left plot, we see that unifying the WLD with the SLD does not help nor harm compared to using the SLD alone.

We conclude that even though WLD is not nearly as accurate as SLD, it has the potential to improve performance, if blended correctly.

We also tried gradually increasing the amount of WLD in each blending epoch, instead of decreasing it. We tested several increasing factors on both types of WLD. Results were similar to the proposed blending method.

6 Conclusions

Neural networks have become widely useful in natural language understanding tasks. It is often the case that there is not enough high quality labeled data for the target task, leading to significant drops in network performance. On the other hand, for many tasks, weak labeled data can be easily obtained but is usually noisy.

In this work we explore a way to enable a network to take advantage of the large size of WLD without overriding the high quality of SLD.

In the method we present, training starts with initialization epochs in which only the WLD is used. It continues with blending epochs in which the data fed to the network is a mixture of WLD and SLD. The blending method we presented, assigns higher importance to the vast amount of WLD at the beginning of the training and decreases its impact as training progresses.

We evaluate our blending method on the task of topic-dependent evidence detection, leveraging two WLD sources, and show that it improves performance for each source. The impact of blending increases as the amount of SLD decreases.

Additionally, we release a data set of 5,785 manually labeled sentences to encourage reproducibility and further work on evidence detection. SLD is a large corpus, and the two WLD we used is evidently different: the Webis corpus seems to help more than the “that + topic concept” query. This calls for future work of understanding what makes a good fit between WLD and SLD. The amount of WLD does not seem to be an important factor, as we see that blending the smaller WLD of the two achieves better performance. It is probably highly related to the quality of the WLD. Sentences retrieved from Wikipedia are of many forms and domains, while the Webis corpus is composed of sentences from debates, which might explain why the network is able to leverage it better.

For future work we intend to examine ways to find better WLD and to make better use of it. For example, instead of choosing one type of WLD, we can combine several WLD types together.
References


Abstract

This paper presents a task for machine listening comprehension in the argumentation domain and a corresponding dataset in English. We recorded 200 spontaneous speeches arguing for or against 50 controversial topics. For each speech, we formulated a question, aimed at confirming or rejecting the occurrence of potential arguments in the speech. Labels were collected by listening to the speech and marking which arguments were mentioned by the speaker. We applied baseline methods addressing the task, to be used as a benchmark for future work over this dataset. All data used in this work is freely available for research.

1 Introduction

Machine reading comprehension (MRC) is the NLP task equivalent to reading comprehension tests that assess the understanding of written texts by humans. MRC is usually realized as a question answering (QA) task through multiple-choice questions or as a cloze test (Richardson et al., 2013; Hermann et al., 2015; Hill et al., 2015). With the abundance of multimedia content nowadays, this line of work has been extended to speech, by applying QA methods to speech transcripts, i.e. the output of automatic speech recognition (ASR). In such works, the task is consequently referred to as ‘spoken question answering’ (Li et al., 2018), ‘question answering over speech transcripts’ (Turmo et al., 2007; Lamel et al., 2008) or machine listening comprehension (MLC) (Chung and Glass, 2018).

We continue this line of work, and present a listening comprehension task and associated benchmark data over argumentative content. In the argumentation domain, such as political debates, people are often exposed directly to the audio (or the video), without access to a written version. Human comprehension is then done in real-time through listening. We simulate this scenario in our dataset. Namely, annotation is carried out by listening to debate speeches rather than by reading transcripts as done in prior work. The auditory modality is richer than written text in terms of the signal available to listeners, e.g., prosody. Similarly, machine comprehension can make use of the extra-lexical signal. The dataset we construct and release enables utilizing such signals, as done for instance in (Lippi and Torroni, 2016) for detecting claims in debates.

Most often, in both reading and listening comprehension tasks, the answer is explicitly mentioned in the text; frequently, the answer is even an actual segment of the text, as in SQuAD (Rajpurkar et al., 2016), one of the most popular MRC datasets. Conversely, in argumentation, presuppositions are fundamental (Habernal et al., 2018), inferences are more subtle and the answer may rely on common knowledge. Going beyond the factoid level, Tseng et al. (2016) presented a listening comprehension task over TOEFL listening tests. In comparison, our data consists of spontaneous speech and is not adapted for non-native speakers.

We use data from iDebate2, a high-quality, curated database of controversial topics – referred to as “motions”, as in formal parliamentary proposals – with a list of arguments for and against each motion. We selected 50 motions, and recorded experienced debaters making four speeches for each of them (two for and two against the motion). We then asked annotators to listen to a speech and present them with a list of arguments that were proposed independently in iDebate for the motion. The annotators had to mark which of these arguments were mentioned in the speech. All data used in this work is freely available for research.

2 Listening Comprehension over Argumentative Content

Shachar Mirkin1, Guy Moshkowich2, Matan Orbach2, Lili Kotlerman2, Yoav Kantor2, Tamar Lavee3, Michal Jacovi2, Yonatan Bilu2, Ranit Aharonov2 and Noam Slonim2

1Digimind, 2IBM Research, 3IBM Watson

1This work was done at IBM within Project Debater; the first 3 authors equally contributed to this work.
2https://www.ets.org/toefl
3https://idebate.org/debatabase
ments were mentioned in the speech (see Section 2 for further details). Example 1 shows one such argument alongside a speech snippet, demonstrating the unique nature of this domain and data. Specifically, the argument against the motion is implied from the speech, but is not explicitly mentioned in it.

Example 1 (Positively labeled argument) Motion: We should introduce goal line technology in sports

Argument: Controversy and debate are a part of the game

Speech stance: opposing (“con”)... people also like to see some extent when officials make mistakes, because it adds to some of the like drama, the, oh, what if this happened? ... And I think that one of the biggest things that fans enjoy bonding over is when refs make mistakes that are blatantly wrong.

iDebate was chosen since its arguments are good candidates to construct comprehension questions: more than half of the assessed arguments are mentioned in our recorded speeches, and the large majority of the speeches contain at least one of the arguments suggested by iDebate. Furthermore, each argument in iDebate is coupled with a counter-argument. Those, in turn, may be used to rebut each argument that was detected through MLC, suggesting intriguing future directions to explore the released data. In a task related to ours, Bolić and Šnajder (2014) have searched iDebate arguments in user argumentative comments. Their work, though, consisted of only two motions and included written, rather than audio data.

We release our annotated data and the results of baseline methods applied to it as a benchmark dataset for the MLC task. The dataset includes 200 speeches for 50 motions, in English. For each speech we include the following: (i) the audio version of the speech; (ii) manual and automatic transcripts; (iii) a labeled listening comprehension question, consisting of a set of arguments from iDebate that potentially appear in the speech.

The main contributions of this work are: (i) proposal of the new task of listening comprehension over argumentative content, a domain very different from those previously used for reading or listening comprehension tasks; (ii) a comprehensive and rich labeled dataset of 200 speeches covering 50 topics, transcribed both automatically and manually, and labeled for the listening comprehension task; (iii) establishment of baselines over the dataset.

All the recordings, their transcripts and labels are available for research at https://www.research.ibm.com/halfa/dept/vst/debating_data.shtml. For comparison, the WER of the ASR transcripts: 8.03% on average. For comparison, the WER of the ASR transcripts in (Li et al., 2018) is 22.73%.

Labeling Given the recorded speeches, we carried out a labeling task employing experienced annotators, all of whom are highly proficient English-speakers. Given a motion and a speech, the annotators were instructed to listen to it once, preferably without pausing, and select which ones of iDebate argument titles were mentioned in it, or None if none of them was. Specifically, they were instructed to answer positively if it would be correct to say that “the speaker argued that arg...” where arg is the argument’s title. A single question contained all the iDebate titles for the motion, which have the suitable stance for the given speech. Each of the 200 questions was answered by five annotators.

On average, the labeled data contains 4.4 argument titles per speech, where a title contains 10.5 words and an argument text includes 6 sentences and 150 words.

Labeling results
In 173 (86.5%) of the speeches, at least one iDebate argument was found, and 248 (~56%) of the iDebate arguments were labeled as positive at least once.

In order to analyze agreement between annotators, we transformed each multiple-choice question to a set of binary questions containing a speech and a single argument title. This amounted to 878 annotated speech+argument pairs, of which 354 (40.3%) are labeled as positive (i.e. an average of ~1.8 positive arguments per speech). The average pairwise Cohen’s Kappa score over the labels is 0.59 and the Fleiss’ Kappa is 0.60. While these may indicate that the reading task is somewhat easier (e.g. because the annotator can read the text multiple times), it was encouraging to discover that audio-based labeling achieves similar results to text-based labeling: we compared the labels obtained via reading and via listening and found that 87% of them were identical. Labeling by listening is closer to the task of listening comprehension than labeling via reading. Another advantage is that it removes the need to manually transcribe the speeches (in our experience, ASR transcripts are not ideal for labeling). As mentioned, in our work we accessed iDebate on Jan. 28, 2018.

Note: We use manual and not ASR transcripts for this analysis, under the assumption that when listening, the annotators are also receiving error-free content.
3.1 Assessed methods
All-yes baseline As a reference point, we compute the accuracy obtained when all arguments are predicted to be mentioned in the speech, resulting in 39.8% accuracy.

table2vec (w2v) We create a w2v (Mikolov et al., 2013) vector representation for each text, removing stopwords; each word is represented by a 200-dimensional word embedding learned over Wikipedia. A tf-idf-weighted average of the word embeddings represents each text, where tf/idf values are counted when considering each Wikipedia sentence as a document. Given a pair of texts, their score is the cosine similarity between their vector representations.

skip-thought (ST) Kiros et al. (2015) presented a general sentence encoder, that has been applied successfully to a variety of tasks such as semantic relatedness and paraphrase detection, often obtaining state of the art results. We use its available implementation to encode the texts as vectors, and compute the cosine similarity between them.

3.2 Results
Table 2 shows the accuracy of all w2v configurations. Representing an argument using more verbose several-sentences-long content outperforms using its short single-sentence title. On the speech side, considering each sentence separately is preferable to using the entire speech. We compare the results of the best w2v-based configuration (arg-sentence), to the performance of the skip-thought auto-encoder. In this setting, encoding individual speech sentences and an argument, the accuracy of skip-thought was 60.2%.

4 Conclusions
Machine listening comprehension is a challenging task, whose complexity stems, among other things, from the difficulty to handle spoken language and from errors due to automatic transcription. The argumentation domain, often with complex and elaborate reasoning, relying on presuppositions and world knowledge, adds another dimension to this complexity. In this work, we suggest a task and a corresponding benchmark dataset to assess comprehension in this domain. We focused on the task of confirming the occurrence of arguments in a speech, in which – as shown in this work – can be handled to some degree with standard textual inference methods. Other types of questions can be formulated over this data in following work. We release a rich dataset, accompanied with benchmarks, that can drive various studies in listening comprehension and argumentation mining.

5 Acknowledgments
We are thankful to the debaters and annotators who took part in the creation of this dataset.

References
Yi-An Chung and James R. Glass. 2018. Speech2Vec: A Sequence-to-Sequence Framework for Learning...


Lori Lamel, Sophie Rosset, Christelle Ayache, Djamel Mostefa, Jordi Turmo, and Pere Comas. 2008. Question Answering on Speech Transcriptions: the QAST evaluation in CLEF. In Proceedings of LREC.


Section 1: Augmenting Models

1. Co-regularized Alignment for Unsupervised Domain Adaptation

2. Domain Adaptation by Using Causal Inference to Predict Invariant Conditional Distributions
   Magliacane S, Claassen T, Bongers S, Versteeg P, Mooij JM. NIPS. 2018

3. Jointly Optimize Data Augmentation and Network Training: Adversarial Data Augmentation in Human Pose Estimation
   Peng X, Tang Z, Yang F, Feris RS, Metaxas D. CVPR. 2018

4. ∆-encoder: an effective sample synthesis method for few-shot object recognition
   Schwartz E, Karlinsky L, Shtok J, Harary S, Marder M, Kumar A, et al. NIPS. 2018

5. Routing Networks: Adaptive Selection of Non-Linear Functions for Multi-Task Learning
   Rosenbaum C, Klinger T, Riemer M. ICLR. 2018

6. Learning Deep Network Representations with Adversarially Regularized Autoencoders

7. Sobolev LGAN
   Mroueh Y, Li C-L, Szuc T, Raj A, Chen Y. ICLR. 2018

8. Non-parametric adaptation of Domain-Generalization in Generative Adversarial Network training
   Sim X, Rusu A. AISTATS. 2018

Section 2: Learning, Knowledge and Reasoning

9. Discovering Implicit Knowledge with Unary Relations
   Glass M, Gliozzo A. ACL. 2018

10. Semi-Supervised Concept Discovery over Event Databases
    Hossain-Ahmed D, Trewin S, Glass A. ESWC. 2018

11. A Systematic Classification of Knowledge, Reasoning, and Context within the ARK Dataset

12. Neural-Symbolic VQA: Quantifying Reasoning from Vision and Language Understanding
    Vahid N, Qian C, Tominka A. Deszendr PK, Tawerbaum JB. NIPS. 2018

Section 3: Optimal Actions

13. Exemplar Option Discovery through Temporal Abstraction and Representation Learning

14. Learning by noticing
    Sim X, Rusu A. AISTATS. 2018

15. Novel Object Learning in Deep Reinforcement Learning
    Glass M, Gliozzo A. ACL. 2018

Section 4: Perception


17. Dialog-based Image Retrieval
    So X, Wu H, Zhang Y, et al. CVPR. 2018

18. A Prior-Less Method for Multi-Face Tracking in Unconstrained Videos
    Lin C-C, Hang Y. CVPR. 2018

Section 5: Mastering Language

19. Detecting Egregious Conversations between Customers and Virtual Agents

20. Fighting Offensive Language on Social Media with Unsupervised Text Style Transfer
    dos Santos C, Molina 1, Padhi S. ACL. 2018

21. Exemplar Encoder-Decoder for Neural Conversation Generation
    Pandey G, Contractor D, Kumar V, Joshi S. ACL. 2018

22. Neural Cross-Lingual Coreference Resolution and Its Application To Entity Linking
    Kundu G, Shi A, Flanair R, Hermes W. ACL. 2018

23. Exploiting Structure in Representations of Named Entities using Active Learning
    Shokhn M, Qin M, Lu Y, Jagadeesh H V, Hernandez MA, Voss M. COLING. 2018

24. Will it Blend? Blending Weak and Strong Labeled Data in a Neural Network for Argumentation Mining
    Shnarch E, Alzate C, Dankin L, Gleize M, Hou Y, Choshen L, et al. ACL. 2018

25. Listening Comprehension over Argumentative Content
Credits

Design and Illustration: Carl De Torres, Francesco Muzzi, Story

Copyright 2018: IBM

Printed in Canada: Hemlock Printers Ltd.