Adversarial and Parameter Generation Networks for Multi-Source Cross-Domain Dependency Parsing

author

Abstract

Thanks to the strong representation learning capability of deep learning, especially pretraining techniques with language model loss, dependency parsing has achieved great performance boost in the in-domain scenario with abundant labeled training data for targeted domains. However, the parsing community has to face the more realistic setting where labeled data only exists for several fixed domains, known as the domain adaptation problem. In this work, we propose a novel model for multi-source cross-domain dependency parsing. The model consists of two components, i.e., an adversarial network for learning domain-invariant representations, and a parameter generation network for distinguishing domain-specific features. Experiments on a recently released NLPCC-2019 dataset for multi-domain dependency parsing show that our model can consistently boost cross-domain parsing performance by about 1.6 LAS point in average over strong BERT-enhanced baselines. Detailed analysis is conducted to gain more insights on contributions of the two components.

1 Introduction

Dependency parsing, as one fundamental task in natural language processing, aims to derive syntactic and semantic tree structures over input sentential words [Kübler *et al.*, 2009; McDonald *et al.*, 2013]. Recently, supervised neural dependency parsing models have achieved great success, leading to impressive performance [Chen and Manning, 2014; Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017; Li *et al.*, 2019a; Zhang *et al.*, 2020]. Remarkably, the Bi-Affine dependency parsing model can obtain a UAS of 96.67 and a LAS of 95.03 on standard Penn Treebank benchmark for the English language.

In order to obtain competitive performing, supervised dependency parsing models rely on a sufficient amount of training dataset, which is inevitably dominated to several fixed domains. When the test dataset is sourced from similar domains, good performance could be achieved. However, the performance could be decreased significantly when the test data is from a different domain which has a large gap between the training domains. Thus domain adaptation for dependency parsing has been concerned by a number of studies [Yu *et al.*, 2013; Yu *et al.*, 2015; Sato *et al.*, 2017; Clark *et al.*, 2018]. These works mostly focus on single-source cross-domain dependency parsing, assuming the training data is from a single source domain [Yu *et al.*, 2013; Sato *et al.*, 2017]. In fact, multi-domain dependency parsing is a more practical setting, considering that several dependency parsing corpora from different domains have been developed [Peng *et al.*, 2019]. Intuitively, an effective exploration of all these corpora can give better performance for the target domain compared with the single-source domain adaption.

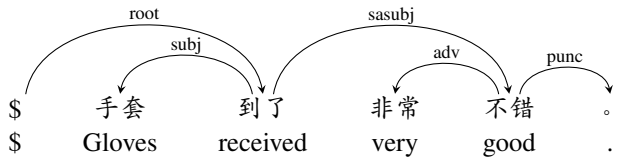


Figure 1: An example of dependency tree from the product comment domain.

为什么说是训练领域间有gap，因为训练集可能是一样的

Separating domain-invariant and domain-specific features is one popular way for domain adaptation to distinguish the similarity and discrepancy of different domains [Daumé III, 2007; Kim *et al.*, 2016; Sato *et al.*, 2017]. Domain-invariant features indicate the shared feature space across domains, which have been widely-adopted as knowledge transferring. Domain-specific features imply the differences between domains, which could be helpful if the domain gaps could be accurately measured and effectively modeled. The learning of domain invariant and specific features are actually complementary because of mutual exclusivity, especially for single-source domain adaption. For multi-domain dependency parsing, the feature separation process could be more challenging.

多源领域更难区分不同类别的特征

In this work, we investigate unsupervised multi-domain dependency parsing, assuming that there exists training corpora for multiple source domains, and no annotated corpus is available for the target domain. We propose a novel network architecture for effective separation of domain-invariant and domain-specific features, denoted as "APGN". Concretely, we exploit an adversarial neural network to detect domain-

invariant features by cheating the domain identification. At the same time, we leverage a parameter generation network (PGN) over a vanilla encoder to learn domain-specific representations by using distributional domain representations as PGN inputs.

We evaluate our approaches on a benchmark dataset containing annotated corpus of four domains. We take the state-of-the-art BiAffine dependency parser as the baseline, and then apply our method on the baseline parser. Experimental results show that our baseline parser can achieve highly-competitive performance only when the testing domain is the same as the training dataset, but its performance drops drastically when domain changes. Our proposed method can boost the parsing performance significantly, leading to averaged UAS and LAS improvements by 2.22 and 2.08, respectively. Detailed comparative experiments show that our method outperforms several other models with alternative domain representation strategies, and our designed distributed domain representation is extremely useful to extract more reliable domain knowledge that benefits for the parsing task. In addition, we conduct in-depth analysis to gain crucial insights on the influence and the effectiveness of our proposed framework. Meanwhile, we find that a proper scale of target-domain unlabeled data can further improve model performance by a large margin. We will release our codes at https://url for facilitating future researches.

其它模型指什么？

2 Baseline Model

In this work, we adopt BiAffine parser [Dozat and Manning, 2017] as our baseline model, which mainly contains four components, i.e., *Input layer*, *Encoder layer*, *MLP layer*, and *BiAffine layer*.

Input layer. The input layer maps each word w_i into a dense vector representation $\mathbf{x}_i$. First, we apply a BiLSTM to encode the constituent characters of each word w_i into its character representation $\mathbf{rep}_i^{\texttt{char}}$. Then, we concatenate $\mathbf{rep}_i^{\texttt{char}}$ with $\mathbf{emb}_i^{\texttt{word}}$ as the input vector $\mathbf{x}_i$.

$$\mathbf{x}_i = \mathbf{emb}_i^{\texttt{word}} \oplus \mathbf{rep}_i^{\texttt{char}} \tag{1}$$

where $\mathbf{emb}_i^{\texttt{word}}$ is the pre-trained word embedding, and $\oplus$ indicates vectorial concatenation. In addition, we also use BERT representation to enhance our model, denoted as $\mathbf{rep}_i^{\texttt{BERT}}$, where $\mathbf{emb}_i^{\texttt{word}}$ is substituted by $\mathbf{rep}_i^{\texttt{BERT}}$ simply.

Encoder layer. Following Dozat and Manning [2017], we employ a three-layer BiLSTM to sequentially encode the inputs $\mathbf{x}_0 \dots \mathbf{x}_n$ and generate context-aware word representations $\mathbf{h}_0 \dots \mathbf{h}_n$. We omit the detailed computation of the BiLSTM due to space limitation, and denote it as follows:

$$\mathbf{h}_0 \dots \mathbf{h}_n = \mathbf{BiLSTM}(\mathbf{x}_0 \dots \mathbf{x}_n, \theta_{\text{BiLSTM}}) \tag{2}$$

where θ_{BiLSTM} represents all parameters of the BiLSTM.

MLP layer. The MLP layer uses two independent MLPs to get lower-dimensional vectors of each position $0 \le i \le n$.

$$\begin{aligned} \mathbf{r}_i^{\text{H}} &= \text{MLP}^{\text{H}}(\mathbf{h}_i) \\ \mathbf{r}_i^{\text{D}} &= \text{MLP}^{\text{D}}(\mathbf{h}_i) \end{aligned} \tag{3}$$

where $\mathbf{r}_i^{\text{H}}$ is the representation vector of w_i as a head word, and $\mathbf{r}_i^{\text{D}}$ as a dependent.

BiAffine layer. The score of a dependency $i \leftarrow j$ is computed by a BiAffine attention as Equation 4:

$$\texttt{score}(i \leftarrow j) = \begin{bmatrix} \mathbf{r}_i^{\text{D}} \\ 1 \end{bmatrix}^{\text{T}} \mathbf{W}^b \mathbf{r}_j^{\text{H}} \tag{4}$$

where the weight matrix $\mathbf{W}^b$ determines the strength of a link from w_j to w_i.

Parsing loss. Assuming the gold-standard head of w_i is w_j, the parsing loss for each position i is computed as follows:

$$L_{par} = -\log \frac{e^{\texttt{score}(i \leftarrow j)}}{\sum_{0 \le k \le n, k \ne i} e^{\texttt{score}(i \leftarrow k)}} \tag{5}$$

The classification of dependency labels can be regarded as a separate task after finding the best dependency tree, and the detailed illustration can be seen in Dozat and Manning [2017].

3 Approach

The multi-source cross-domain setting assumes that there are several source-domain labeled datasets $\text{S} = \{\text{S}_i\}_{i=1}^m$, and one target-domain unlabeled dataset T. The goal is to train a parser that achieves the best performance on the target-domain dev/test datasets, by making full use of all available labeled and unlabeled data.

接下来要总的讲方法的motivation，工作原理的big picture。

下面这一段，直入细节的把方法讲出来，也许能懂怎么做，但是很难理解为什么这么做。 As shown in Figure 2, our model consists of three key components, i.e., *word-level distributed domain representation learning*, *BiAffine parser*, and *word-level adversarial learning*. The key idea is to maximize the utilization of the discrepancy and similarity across domains by effective feature separation. Concretely, an improved parameter generation and a standard adversarial neural networks are exploited to learn domain-specific features $\mathbf{h}_i^{\texttt{spe}}$ and domain-invariant features $\mathbf{h}_i^{\texttt{inv}}$, respectively. Then, the two type of features are combined to train the BiAffine parser.

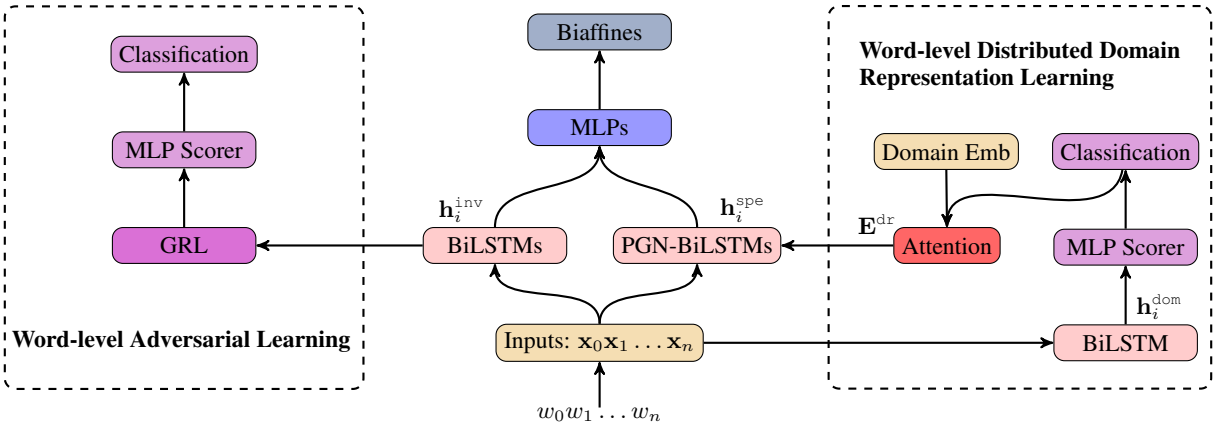
3.1 Learning Domain-Specific Features via Parameter Generation Network

Considering each input word has own domain distributions, encoding all words with the same model parameters may lead to potential domain conflicts. Hence, we design a parameter generation network to deal with this problem, shown as the right part of Figure 2. First, a domain classification is employed to obtain distributed domain representations. Then, the parameter generation network takes the distributed representations as inputs and dynamically generates PGN-BiLSTM parameters. Finally, the word from different domains may have its specific BiLSTM parameters.

怎么做到的？

是单词还是句子？

Distributed domain representation learning. Now we give a detailed illustration of the distributed domain representation learning. First, a one-layer BiLSTM is applied to



Ldom的loss是分类器的loss吗

Figure 2: Framework of our proposed model.

encode the input sentence. Second, we employ MLP and softmax operations to obtain the domain distribution probabilities $\mathbf{z}_i$,

$$\mathbf{z}_i = \text{softmax}\left(\text{MLP}\left(\mathbf{h}_i^{\text{dom}}\right)\right) \tag{6}$$

where $\mathbf{h}_i^{\text{dom}}$ is the BiLSTM output. Third, we directly employ an attention mechanism between $\mathbf{z}_i$ and the domain embedding vectors to obtain word-level distributed domain representation $\mathbf{rep}_i^{\text{dom}}$ that can more accurately represent the domain information for word w_i,

这个领域嵌入是什么

$$\mathbf{rep}_i^{\text{dom}} = \sum_{j=1}^{m} z_i^j \mathbf{emb}_i^{\text{dom}}(j) \tag{7}$$

没有说领域嵌入是什么来的吧？每个单词都有一个领域嵌入吗，还是共有m个

where m is the number of domains, $\mathbf{emb}^{\text{dom}}(j)$ is the fine-tuned domain embedding vector of the word w_i belonging to domain j, and z_i^j is the probability value produced by the softmax operation. Finally, we utilize an average pooling to generate the sentence-level distributed domain representation $\mathbf{E}^{\text{dr}}$ that will be used as the input of PGN-BiLSTM.

句子级的领域表示

$$\mathbf{E}^{\text{dr}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{rep}_i^{\text{dom}} \tag{8}$$

PGN-BiLSTM encoder. To better capture domain-specific features, we exploit the PGN-BiLSTM instead of a standard BiLSTM encoder. For convenience, we directly formalize the vallina BiLSTM encoder as follows:

为什么用V？不用theta？为什么不说是PGN-BiLSTM？

$$\mathbf{h}_0 \ldots \mathbf{h}_n = \text{BiLSTM}\left(\mathbf{x}_0 \ldots \mathbf{x}_n, \mathbf{V}\right) \tag{9}$$

where $\mathbf{V}$ can be regarded as a flattened vector which contains all the BiLSTM parameters. Compared with the vanilla BiLSTM, PGN-BiLSTM enables to dynamically generate the domain-related parameters for BiLSTM, which is formalized as follows:

$$\begin{aligned}\mathbf{h}_0^{\text{spe}} \ldots \mathbf{h}_n^{\text{spe}} &= \text{PGN-BiLSTM}\left(\mathbf{x}_0 \ldots \mathbf{x}_n, \mathbf{E}^{\text{dr}}\right) \\ &= \text{BiLSTM}\left(\mathbf{x}_0 \ldots \mathbf{x}_n, \mathbf{V}^{\text{dr}}\right) \\ &= \text{BiLSTM}\left(\mathbf{x}_0 \ldots \mathbf{x}_n, \mathbf{W}\mathbf{E}^{\text{dr}}\right)\end{aligned} \tag{10}$$

这个W是什么？为什么这么做？

where $\mathbf{V}^{\text{dr}}$ is dynamically produced via $\mathbf{W}\mathbf{E}^{\text{dr}}$ operation, $\mathbf{W}$ is a meta parameter of PGN-BiLSTM, and $\mathbf{E}^{\text{dr}}$ is computed as Equation 8.

Different from Jia et al. [2019], our PGN-BiLSTM parameters are generated based on a distributed domain embedding $\mathbf{E}^{\text{dr}}$ rather than a fixed one. Intensively, the distributional representation is more useful than the fixed one to integrate multi-domain information and reduce potential domain conflicts. In addition, we will conduct detailed comparative experiments to verify the effectiveness of the distributional domain representation.

3.2 Learning Domain-Invariant Features via Adversarial Network

标题中就定性了：adversarial就是学domain-invariant特征，可能有点过于肯定。adversarial的其他论文都是这么明确的规定其作用吗？（如果不肯定，先放着，等最后改。摘要中的说法也需要斟酌）我感觉应该先讲PGN，两个小的改进（贡献）都在PGN。对抗只是使用了一下，没有什么改进。 The architecture of the adversarial network is shown in the left part of Figure 2. First, the input words from different domains are parameterized by a shared BiLSTM. Then, the output $\mathbf{h}_i^{\text{inv}}$ is fed into a Gradient Reversal Layer (GRL). Finally, the domain classifier receives $\mathbf{h}_i^{\text{inv}}$ and attempts to identify the domain of the input word. During training, we train the BiLSTM to make it difficult for the domain classifier to correctly distinguish domain category. Thus the shared BiLSTM is encouraged to find the domain-invariant features that are not specific to a particular domain as much as possible. As a result, we expect the parser is able to utilize the shared knowledge from multiple domains effectively.

We now give a more detailed illustration about the main components of adversarial network. Following Ganin and Lempitsky [2015], the forward and backward propagations for the GRL are defined as follows:

为什么梯度是个负值？理论上是1啊

$$\begin{aligned} GRL_\lambda(\mathbf{h}_i^{\text{inv}}) &= \mathbf{h}_i^{\text{inv}} \\ \frac{\mathrm{d}GRL_\lambda(\mathbf{h}_i^{\text{inv}})}{\mathrm{d}(\mathbf{h}_i^{\text{inv}})} &= -\lambda I \end{aligned} \tag{11}$$

为什么要在GRL层进行平衡？

where λ is a hyper-parameter that is used to balance the domain classification and dependency parsing tasks, and I is an

I为什么是个单位阵？而不是向量？

Algorithm 1 Joint Training Procedure

Input: labeled multi-source domain data $\mathrm{S} = \{\mathrm{S}_i\}_{i=1}^{m}$, unlabeled target domain data T.
Hyper-parameters: loss weights: α and β. joint training iteration k.
Output: Target model.

1: **Repeat**
2: **if** $iter < k$ **do**
3: Sample a mini-batch $x^s \in$ S
4: Accumulate loss $L = L_{par} + \alpha L_{adv} + \beta L_{dom}$
5: Sample a mini-batch $x^t \in$ T
6: Accumulate loss $L = \alpha L_{adv} + \beta L_{dom}$
7: **else**
8: Sample a mini-batch $x^s \in$ S
9: Accumulate loss $L = L_{par}$
10: $iter += 1$
11: **until** convergence

identity matrix. Over the GRL, we apply an MLP to compute the domain distribution scores and a softmax to select the domain category.

$$\mathbf{z}_i = \text{softmax}\left(\text{MLP}\left(\mathbf{h}_i^{\text{inv}}\right)\right) \tag{12}$$

where $\mathbf{h}_i^{\text{inv}}$ is the shared BiLSTM output. Finally, the cross-entropy is used to compute the adversarial loss.

这里是个负值吧，那么是最大化还是最小化？ 没有体现出对抗过程

$$L_{adv} = \sum_{i=0}^{n}\sum_{j=1}^{m} \hat{z}_i \log\left(z_i^j\right) \tag{13}$$

where $\hat{z}_i$ is the gold domain of word w_i defined according to which domain the word comes from, z_i^j represents the probability of word w_i belonging to domain j, m is the number of domains, and n is the word number of one sentence.

3.3 Joint Training

In this work, we design a joint training strategy to make full use of all available training data, as shown in Algorithm 1. In the first k iterations, mini-batches of source-domain and target-domain take turns to train (lines 3-4 and 5-6, respectively). If the mini-batch comes from the source-domain labeled data, we jointly train the model with the parsing, adversarial, and domain classification losses. Otherwise, the model is trained with the adversarial and domain classification losses. In the first stage, all data is used to select domain-invariant and domain-specific features via the adversarial and parameter generation networks. To deal with the overfitting problem of the domain classifications, the model is updated with only parsing loss until convergence after k iterations.

目标领域判断为哪个领域呢？

没说什么时候更新参数

4 Experiments

4.1 Settings

Data. We use the Chinese multi-domain dependency parsing datasets released at the NLPCC-2019 shared task[1], containing four domains: one source domain which is a balanced corpus (BC) from news-wire, three target domains which are the product comments (PC) data from Taobao, the product blog (PB) data from Taobao headline, and a web fiction data named "ZhuXian" (ZX). Table 1 shows the detailed illustration of the data statistics. In this work, we pick one target domain as the meta-target and the rest domains as the meta-source. For example, if the meta-target domain is PC, meta-source domains are BC, PB, and ZX.

	BC	PC	PB	ZX
train	16,339	6,885	5,129	1,645
dev	997	1,300	1,300	500
test	1,992	2,600	2,600	1,100
unlabeled	-	349,922	291,481	33,792

Table 1: Data statistics in sentence number

Evaluation. We use unlabeled attachment score (UAS) and labeled attachment score (LAS) to evaluate the dependence parsing accuracy. Each model is trained for at most $1,000$ iterations, and the performance is evaluated on the dev data after each iteration for model selection. We stop the training if the peak performance does not increase in 100 consecutive iterations.

Hyper-parameters. We mostly follow the hyper-parameter settings of Dozat and Manning [2017], such as learning rate, dropout ratios, and so on. The Chinese character embeddings are randomly initialized, and the dimension is 100. The loss weights α and β are set as 0.01. For pre-trained word embeddings, we train word2vec [Mikolov *et al.*, 2013] embeddings on Chinese Gigaword Third Edition, consisting of about 1.2 million sentences. For BERT, we use the released Chinese BERT-Based model to obtain the fixed BERT representations for each word.[2] Following Li et al. [2019a], we utilize the averaged sum of the top-4 layer outputs as the final BERT representation $\mathbf{rep}_i^{\text{BERT}}$.

Baseline models. To verify the effectiveness and advantage of our proposed model, we select the following approaches as our strong baselines.

- **Concatenation (Concat).** We directly train the BiAffine parser [Dozat and Manning, 2017] with the concatenation of all meta-source training data. The main drawback is that the model shares all parameters across different domains and ignores the domain differences, thus making it difficult to build the relationship between different domains.
- **Domain embedding (DE).** The vallina DE method has been proven more effective than the Concat approach on semi-supervised dependency parsing [Li *et al.*, 2019c]. The key idea is to train the BiAffine parser with an extra fixed domain embedding to indicate which domain the input sentence comes from. However, when directly applies the DE method to our task, the fixed representation is trained inadequately due to the lack of target-domain labeled data.
- **Adversarial domain embedding (ADE).** Li et al. [2020b] propose to apply adversarial learning on the DE method, which utilizes the domain-aware embedding and adversarial learning to extract the

[1] http://hlt.suda.edu.cn/index.php/Nlpcc-2019-shared-task

[2] https://github.com/google-research/bert

	PC		PB		ZX		Avg.	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
BC-train	42.60	29.14	68.33	61.91	73.55	66.05	61.49	52.37
PC-train	**67.28**	**59.33**	62.00	54.93	45.73	38.40	58.34	50.89
PB-train	43.05	31.54	**75.19**	**70.38**	57.44	47.90	58.56	49.94
ZX-train	32.19	19.55	53.87	46.23	**73.82**	**67.93**	53.29	44.57
All-train	48.97	37.32	73.36	67.61	73.30	65.53	**65.21**	**56.82**

Table 2: Results of BiAffine parser on dev data with different training data setting. "All-train" means training a parser with the concatenation of multi-source data.

	PC		PB		ZX		Avg.	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
Comparison with Our Implemented Baseline Models								
Concat	47.30	35.63	72.81	67.24	71.00	62.91	63.70	55.26
DE	47.49	35.56	72.61	67.08	70.98	62.68	63.69	55.11
ADE	48.18	35.85	72.80	67.25	71.46	63.59	64.15	55.56
PGN	49.53	36.87	72.71	66.93	70.65	63.16	64.30	55.66
APGN	**51.48**	**39.12**	**73.86**	**68.10**	**72.43**	**64.80**	**65.92**	**57.34**
Comparison with BERT-Enhanced Baseline Models								
Concat	60.62	49.52	81.59	77.07	80.60	74.53	74.27	67.04
DE	60.45	49.49	82.08	77.15	79.85	73.65	74.13	66.76
ADE	60.76	50.22	82.54	78.04	81.43	75.70	74.91	67.99
PGN								
APGN	**62.98**	**51.90**	**82.92**	**78.21**	**82.00**	**76.08**	**75.97**	**68.73**

Table 3: Final results on test data.

domain-specific and domain-invariant features. But it also exists the inadequate training problem as vallina DE method.

- **Parameter generation network (PGN).** Jia et al. [2019] use a parameter generation network to dynamically generate BiLSTM parameters based on task and domain representations. Different from the vallina PGN, we also exploit the PGN with a distributed domain embedding to generate domain-related BiLSTM parameters as our strong baseline.

4.2 Effect of Source Domain Data Size

In practical applications, the scale of training data has a significant effect on the parsing performance. Table 2 reports the parsing accuracy of BiAffine parser, which is trained with different training data. On the one hand, we can see that although PC-train is much smaller than the concatenation of all multi-source domains, the PC-trained parser still obviously outperforms the All-trained one. The experimental results demonstrate that BiAffine parser can achieve good performance when the training and testing datasets are from the same domain, but its performance drops drastically when the domain changes. Therefore, it is urgent to solve the domain transfer problem, specially without target-domain labeled data. On the other hand, All-trained parser achieves better performances than the one trained with a single-source domain on PC-dev and PB-dev, but it seems no obvious improvement on ZX-dev. Considering large parameters sharing may lead to negative transfer, we design a feature separation architecture to control the transformation of domain feature distributions, and all multi-source domains labeled data will be used for the model training in the following experiments.

指提升不明显，而非没提升

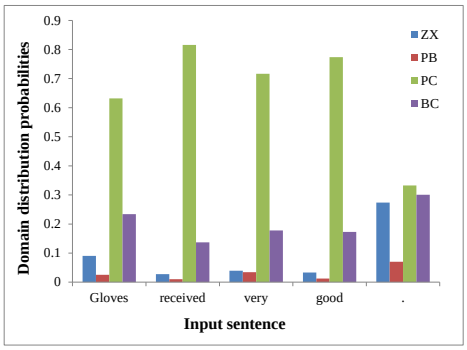


Figure 3: Domain distributional probabilities of different words.

4.3 Final Results

Table 3 shows the final results and makes a comparison with multiple baselines on test data. First, we can see that our proposed APGN model achieves the best results on all domains, demonstrating that the APGN is extremely useful for unsupervised dependency parsing. Second, the ADE model improves the parsing accuracy indicates that feature separation is an efficient way to construct the domain discrimination and reduce the difficulty for domain transformation. Specially, our APGN outperforms the ADE model by about 2 points on the averaged LAS, showing that the parameter generation network based on distributed domain representation is meaningful to fuse domain knowledge and boost the parsing performance. Finally, although the performance of different models is obviously improved by utilizing BERT representation, our model still achieves consistently higher accuracy than other baselines, further demonstrating the effectiveness of our proposed method.

4.4 Comparisons on Alternative Domain Representation Strategies

Most previous works use a fixed domain embedding to indicate which domain the input word comes from [Jia *et al.*, 2019; Li *et al.*, 2019c]. However, the fixed representation may lead to potential domain conflicts when a word belongs to multiple domains. As shown in Figure 3, we can see that each word has its unique domain distribution and it is difficult to define all word with an explicit fixed representation. Hence, it is necessary to design a more accurate representation, named as distributed domain embedding, which can be regarded as weighted sum of the fixed domain embeddings and its distributional probabilities.

Detailed comparative experiments are conducted to verify the effectiveness of two domain representation strategies on various models, and the results are shown in Table 4. First, we find that the APGN with fixed domain representation like Jia et al. [2019] achieves lower performance than other models. The main reason may be that without the language model as the bridge between domains, it is difficult for the parameter generation network to construct the relationship of different domains with the fixed representation. Second, the APGN with distributed domain representation achieves best performance among all the models, indicating that the distributed

这里和PGN有什么区别吗？

为什么表4和表3的数据不一致

	PC		PB		ZX		Avg.	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
Models with the fixed domain representations								
DE	48.23	36.40	73.25	67.39	73.27	66.49	64.92	56.76
ADE	**49.16**	**36.68**	**73.49**	**67.89**	**73.91**	**67.01**	**65.52**	**57.19**
APGN	44.20	30.89	71.28	65.35	71.50	63.85	62.33	53.36
Models with the distributed domain representations								
DE	50.37	38.13	73.96	67.88	73.71	66.61	66.01	57.54
ADE	50.63	38.50	73.90	68.08	73.72	67.79	66.08	58.12
APGN	**52.22**	**40.58**	**74.60**	**68.90**	**75.19**	**68.26**	**67.34**	**59.25**

Table 4: Results of different models with fixed or distributed domain representations.

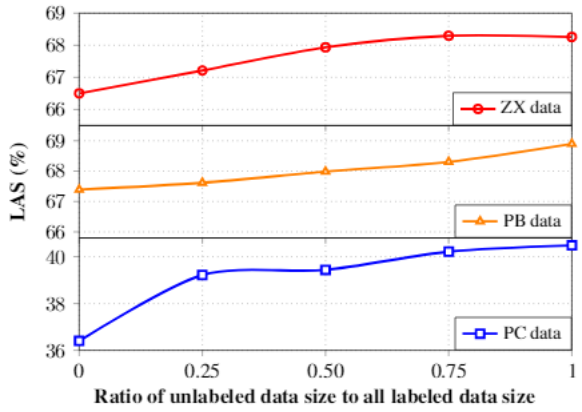


Figure 4: Influence of utilizing different amount of unlabeled data on APGN model. The x-axis is the ratio of target-domain unlabeled data size to source-domain labeled data.

domain embedding has a strong representational capacity and significantly improves our model performance. Finally, we can see that all models with the distributed domain representation outperform the ones with the fixed representation by a large margin, demonstrating that the distributed domain representation is helpful to reduce potential domain conflicts and extracts more reliable domain knowledge that benefits for the parsing task.

4.5 Analysis on Unlabeled Data Size.

Figure 4 illustrates the influence of target-domain unlabeled data size on dev data. In each curve, we fix the size of labeled data and incrementally add a random subset of unlabeled data. Considering a large-scale unlabeled data may lead to the sample unbalance problem, we randomly sample unlabeled data with the ratios less than 1. On the one hand, we can see that using unlabeled data leads to consistently higher performance for all three domains, indicating that the unlabeled data is an important source that benefits for the target-domain dependency parsing. On the other hand, we find that the improvement of parsing accuracy is obviously steady when the ratio is set as 0.75, showing that the APGN model can achieve best performances with a suitable amount of unlabeled data.

强行得出结果？

5 Related Work

Domain adaptation has been extensively studied in many research areas, including machine learning [Zoph *et al.*, 2016; Britz *et al.*, 2017; Wang *et al.*, 2017; Zeng *et al.*, 2019; Li *et al.*, 2020a], computer vision [Ganin and Lempitsky, 2015; Jang *et al.*, 2019; Li and Hoiem, 2018; Rozantsev *et al.*, 2019] and natural language processing [Kim *et al.*, 2016; Liu *et al.*, 2017; Yu *et al.*, 2018; Li *et al.*, 2019b; Sun *et al.*, 2020]. We first give a detailed illustration of single-source domain adaptation on unsupervised and semi-supervised two aspects where only uses a source domain training data. Then, we summarize the approaches on multi-source domain adaptation, where the training data comes from more than one source domain.

Single-source domain adaptation. Due to the lack of the labeled data, previous researches mainly investigate the *unsupervised* domain adaptation. The common methods attempt to extend the training data with the new annotated data by the self-training [Charniak, 1997; Steedman *et al.*, 2003; Yu *et al.*, 2015], co-training [Sarkar, 2001], or tri-training [Li *et al.*, 2019d]. Although most works report that the automatic annotation data can boost the model performance, there are still many failed works on applying self-training and co-training [Steedman *et al.*, 2003].

Thanks to large annotation data released by a variety of communities, recent existing works pay more attention to the *semi-supervised* scenario, where both source and target domains have annotation data. Yu et al. [2013] give detailed error analysis on cross-domain dependency parsing and solve the ambiguous features problem. Sato et al. [2017] propose to separate domain-specific and domain-invariant features by feature argumentation and adversarial learning, finding that there is little gains and even damage the performance, specially when the scale of target-domain training data is small. Most recently, Li et al. [2019c] propose to leverage an extra domain embedding to indicate domain source and prove it outperforms traditional semi-supervised methods. In this work, we adjust the domain embedding method as our another strong baseline.

Multi-source domain adaptation. Many approaches of multi-source domain adaptation focus on leveraging domain knowledge to extract domain-related features, thus boosting the target domain parsing performance [Daumé III, 2007; Kim *et al.*, 2017; Chen and Cardie, 2018; Guo *et al.*, 2018; Li *et al.*, 2018; Li *et al.*, 2020a]. Most recently, Kim et al. [2017] train multiple experts with respective training data, and then use attention weights to make a prediction of the new domain, which performs well on unsupervised spoken language understanding task. Zeng et al. [2018] design a domain classifier and an adversarial network to construct domain-specific and domain-invariant features, achieving good performances on machine translation. Guo et al. [2018] apply meta-training and adversarial learning to compute the point-to-set distance as the weights of multi-task learning network, leading to improvement on classification tasks. As another interesting direction, Jia et al. [2019] propose to generate domain-related and task-related BiLSTM parameters based on task and domain representation vectors, leading to very

promising performances on cross-domain NER task.

Due to the limitation of annotation corpus and the essential difficulty of multi-source domain adaptation, there still lacks such studies on dependency parsing. Therefore, we propose a novel framework to separate domain-invariant and domain-specific features by the utilization of adversarial and parameter generation networks.

6 Conclusion

In this paper, we propose a simple yet effective approach for feature separation where an adversarial network is used to detect domain-invariant features and a parameter generation network is exploited to capture domain-specific features by using distributed domain representation as input. Experimental results show that our proposed model significantly outperforms the BiAffine parser, even when the model is enhanced with BERT. Detailed comparative experiments demonstrate that our distributed domain representation is extremely useful to reduce domain conflicts, thus extracting more reliable domain-specific features that benefit for dependency parsing. In-depth analysis indicates that suitable unlabeled data enables to boost parsing performance.

7 Comments

李老师意见**2021.1.14**

intro第一段话有点过时了，2020年也有一些工作了，要引用。第二段和第一段有些嗦，IJCAI短一些，要紧凑 这篇文章用了unlabeled data，而且作用还比较大，Intro中应该要体现出来才好，实验中已经有对应结果了

Section2对baseline的介绍，要根据不同的工作，进行相应的调整，并且尽量不要和之前的文章太像。(已经调整好了) 3部分，我今天再仔细读一遍，有意见我再提。

References

[Britz *et al.*, 2017] Denny Britz, Quoc V. Le, and Reid Pryzant. Effective domain mixing for neural machine translation. In *Proceedings of WMT*, pages 118–126, 2017.

[Charniak, 1997] Eugene Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of AAAI*, pages 598–603, 1997.

[Chen and Cardie, 2018] Xilun Chen and Claire Cardie. Multinomial adversarial networks for multi-domain text classification. In *Proceedings of NAACL-HLT*, pages 1226–1240, 2018.

[Chen and Manning, 2014] Danqi Chen and Christopher D. Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP*, pages 740–750, 2014.

[Clark *et al.*, 2018] Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. Semi-supervised sequence modeling with cross-view training. In *Proceedings of EMNLP*, pages 1914–1925, 2018.

[Daumé III, 2007] Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of ACL*, pages 256–263, 2007.

[Dozat and Manning, 2017] Timothy Dozat and Christopher Manning. Deep biaffine attention for neural dependency parsing. abs/1611.01734, 2017.

[Ganin and Lempitsky, 2015] Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of ICML*, pages 1180–1189, 2015.

[Guo *et al.*, 2018] Jiang Guo, Darsh J. Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. In *Proceedings of EMNLP*, pages 4694–4703, 2018.

[Jang *et al.*, 2019] Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Learning what and where to transfer. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 3030–3039, 2019.

[Jia *et al.*, 2019] Chen Jia, Xiaobo Liang, and Yue Zhang. Cross-domain NER using cross-domain language modeling. In *Proceedings of ACL*, pages 2464–2474, 2019.

[Kim *et al.*, 2016] Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. Frustratingly easy neural domain adaptation. In *Proceedings of COLING, Osaka, Japan*, pages 387–396, 2016.

[Kim *et al.*, 2017] Young-Bum Kim, Karl Stratos, and Dongchan Kim. Domain attention with an ensemble of experts. In *Proceedings of ACL*, pages 643–653, 2017.

[Kiperwasser and Goldberg, 2016] Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 4:313–327, 2016.

[Kübler *et al.*, 2009] Sandra Kübler, Ryan T. McDonald, and Joakim Nivre. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2009.

[Li and Hoiem, 2018] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12):2935–2947, 2018.

[Li *et al.*, 2018] Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. Hierarchical attention transfer network for cross-domain sentiment classification. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of AAAI*, pages 5852–5859, 2018.

[Li *et al.*, 2019a] Ying Li, Zhenghua Li, Min Zhang, Rui Wang, Sheng Li, and Luo Si. Self-attentive biaffine dependency parsing. In *Proceedings of IJCAI*, pages 5067–5073, 2019.

[Li *et al.*, 2019b] Yitong Li, Timothy Baldwin, and Trevor Cohn. Semi-supervised stochastic multi-domain learning using variational inference. In *Proceedings of ACL*, pages 1923–1934, 2019.

[Li *et al.*, 2019c] Zhenghua Li, Xue Peng, Min Zhang, Rui Wang, and Luo Si. Semi-supervised domain adaptation for dependency parsing. In *Proceedings of ACL*, pages 2386–2395, 2019.

[Li *et al.*, 2019d] Zuchao Li, Junru Zhou, Hai Zhao, and Rui Wang. Cross-domain transfer learning for dependency parsing. In *Proceedings of NLPCC*, pages 835–844, 2019.

[Li *et al.*, 2020a] Rumeng Li, Xun Wang, and Hong Yu. Metamt, a meta learning method leveraging multiple domain data for low resource machine translation. In *Proceedings of AAAI*, pages 8245–8252, 2020.

[Li *et al.*, 2020b] Ying Li, Zhenghua Li, and Min Zhang. Semi-supervised domain adaptation for dependency parsing via improved contextualized word representations. In *Proceedings of COLING*, pages 3806–3817, 2020.

[Liu *et al.*, 2017] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Adversarial multi-task learning for text classification. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of ACL*, pages 1–10, 2017.

[McDonald *et al.*, 2013] Ryan T. McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B. Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. Universal dependency annotation for multilingual parsing. In *Proceedings of ACL*, pages 92–97, 2013.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119, 2013.

[Peng *et al.*, 2019] Xue Peng, Zhenghua Li, Min Zhang, Rui Wang, Yue Zhang, and Luo Si. Overview of the NLPCC 2019 shared task: Cross-domain dependency parsing. In *Proceedings of NLPCC*, pages 760–771, 2019.

[Rozantsev *et al.*, 2019] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(4):801–814, 2019.

[Sarkar, 2001] Anoop Sarkar. Applying co-training methods to statistical parsing. In *Proceedings of NAACL*, 2001.

[Sato *et al.*, 2017] Motoki Sato, Hitoshi Manabe, Hiroshi Noji, and Yuji Matsumoto. Adversarial training for cross-domain universal dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Vancouver*, pages 71–79, 2017.

[Steedman *et al.*, 2003] Mark Steedman, Anoop Sarkar, Miles Osborne, Rebecca Hwa, Stephen Clark, Julia Hockenmaier, Paul Ruhlen, Steven Baker, and Jeremiah Crim. Bootstrapping statistical parsers from small datasets. In *Proceedings of EACL*, pages 331–338, 2003.

[Sun *et al.*, 2020] Tianxiang Sun, Yunfan Shao, Xiaonan Li, Pengfei Liu, Hang Yan, Xipeng Qiu, and Xuanjing Huang. Learning sparse sharing architectures for multiple tasks. In *Proceedings of AAAI*, pages 8936–8943, 2020.

[Wang *et al.*, 2017] Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. Instance weighting for neural machine translation domain adaptation. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of EMNLP*, pages 1482–1488, 2017.

[Yu *et al.*, 2013] Mo Yu, Tiejun Zhao, and Yalong Bai. Learning domain differences automatically for dependency parsing adaptation. In *IJCAI*, pages 1876–1882, 2013.

[Yu *et al.*, 2015] Juntao Yu, Mohab Elkaref, and Bernd Bohnet. Domain adaptation for dependency parsing via self-training. In *Proceedings of IWPT*, pages 1–10, 2015.

[Yu *et al.*, 2018] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. Diverse few-shot text classification with multiple metrics. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of NAACL-HLT*, pages 1206–1215, 2018.

[Zeng *et al.*, 2018] Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin, and Jianqiang Zhao. Multi-domain neural machine translation with word-level domain context discrimination. In *Proceedings of EMNLP*, pages 447–457, 2018.

[Zeng *et al.*, 2019] Jiali Zeng, Yang Liu, Jinsong Su, Yubin Ge, Yaojie Lu, Yongjing Yin, and Jiebo Luo. Iterative dual domain adaptation for neural machine translation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of EMNLP-IJCNLP*, pages 845–855, 2019.

[Zhang *et al.*, 2020] Yu Zhang, Zhenghua Li, and Min Zhang. Efficient second-order treecrf for neural dependency parsing. In *Proceedings of ACL*, pages 3295–3305, 2020.

[Zoph *et al.*, 2016] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of EMNLP*, pages 1568–1575, 2016.