NAIST-IS-MT1651053

Master's Thesis

Cross-Domain and Cross-Lingual Parsing with Adversarial Training

Motoki Sato

March 15, 2018

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Master's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Master of SCIENCE

Motoki Sato

Thesis Committee:

Professor Yuji Matsumoto	(Supervisor)
Professor Satoshi Nakamura	(Co-supervisor)
Associate Professor Masashi Shimbo	(Co-supervisor)
Assistant Professor Hiroyuki Shindo	(Co-supervisor)
Assistant Professor Hiroshi Noji	(Co -supervisor)

Cross-Domain and Cross-Lingual Parsing with Adversarial Training*

Motoki Sato

Abstract

Syntactic parsers are essential for many natural language processing applications, but training them requires expensive annotations of syntactic trees by linguistic experts. To ease the annotation effort, we focus in this thesis on supervised paresr domain adaptation, in which only a limited amount of annotated trees on the target domain is available. Many recent state-of-the-art parsers are neural parsers and utilize word representations conditioned on the entire sentence obtained with bi-directional LSTMs. We pursue a better domain adaptation technique for such modern neural parsers. As in the classical approach to domain adaptation for feature-based models, our adaptation approach tries to capture both domain-general and domain-specific representations with separate LSTMs, while the domain-general one is encouraged to be truly general with the mechanism of adversarial training. Our baseline parser is a recently proposed state-of-the-art arc-factored dependency parser on bi-directional LSTMs. On the experiments on Universal Dependencies 2.0, we find our technique consistently improves the performance for the target domain in a single language setting. We also find our technique is beneficial for learning low-resource language parser, by treating a high-resource language as a source *language*, especially when the source language is typologically similar to the target language.

Keywords:

Dependency Parsing, Domain Adaptation, Adversarial Training

*Master's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT1651053, March 15, 2018.

敵対学習を用いた分野と言語を横断する構文解析*

佐藤 元紀

内容梗概

構文解析は，多くの自然言語処理にとって重要なタスクである．しかし，構文解析器を訓練するには，言語専門家による構文木のアノテーションが必要である．本論文では，アノテーション作業を容易にするために，対象ドメイン上のアノテーションツリーの限定された量しか利用できない状況下における分野適応について考える．近年の構文解析器はニューラルネットワークをベースとなるものが多く使われており，双方向 LSTM で得られた文全体の文脈情報を利用する．我々は，このような最新の構文解析器を分野適応へ応用する．我々のアプローチは，個別の LSTM を用いてドメイン不変およびドメイン特有の表現を獲得することを目的としている．ベースラインに用いた構文解析器は，双方向 LSTM を用いた最高精度の構文解析器である．Universal Dependencies 2.0 の実験では，単一言語の設定において，ターゲットドメインのパフォーマンスを一貫して向上させることがわかった．我々の手法は，ソース言語がターゲット言語と類似している場合に，低リソース言語の学習に有益であることもわかった．

キーワード

構文解析, 分野適応, 敵対学習

*奈良先端科学技術大学院大学 情報科学研究科 情報処理学専攻 修士論文, NAIST-IS-MT1651053, 2018 年 3 月 15 日.

Acknowledgments

本修士論文を執筆にあたり，多くの方々のご支援を頂きました．この場を借りて，ご支援をいただいた皆様に，深く感謝申し上げます．

主指導教員の松本裕治教授には，奈良先端大の自然言語処理研究室に受け入れていただきました．自然言語処理を長年研究してきた研究者としての問題意識は，常に勉強になりました．休学中の私を受け入れていただき，入学後は様々な機会を与えていただき，有意義な 2 年間を送ることができました．

副指導教員の中村哲教授にはゼミナールにおいて有益なコメントを頂きました．

新保仁准教授には，勉強会やゼミにおいて的確な指摘やアドバイスを頂き，議論を行うことができました．

進藤裕之助教授には，系列ラベリングにおけるセグメントの研究で特に指導していただきました．全く研究の進め方が分からなかった私に対して，どのような方針にするべきかアドバイスを頂き，国際会議に参加することができました．

能地宏助教授には，本修士論文の指導や構文解析に関して知識のない私に対して指導していただきました．また，論文執筆の際に多大なる指導をしていただきました．論文の書き進め方に慣れていない私にとって，能地先生のご指導のおかげで論文投稿をすることができました．

さらに研究室外の方々にも大変お世話になりました．株式会社 Preferred Networks の海野裕也さんには，夏のインターンシップや，その後のアルバイトで大変お世話になりました．実装面や，自然言語処理に関する視点がとても勉強になりました．

株式会社 Preferred Networks の坪井祐太さん（元 日本 IBM 株式会社 所属）には共同研究として，多言語対話に関する研究でお世話になりました．定期的に遠隔でミーティングをさせていただき，論文の執筆や分野適応に関することで特に勉強になりました．ありがとうございました．

NTT コミュニケーション科学基礎研究所の鈴木潤さんには，自然言語処理における解釈可能な敵対学習についての共同研究でお世話になりました．鈴木さんの研究に対する姿勢や考え方がとても勉強になりました．とてもフランクに接していただき，技術的にも研究的にも充実した日々を送ることができました．今後とも学んだことを活かして研究をしていきたいと思います．

また北川裕子秘書をはじめ，同期，先輩，後輩にも深く感謝いたします．常に議論の機会がある松本研で２年間を過ごすことができたことは，自分の人生の中でも特に充実した２年間であったと思います．心から感謝いたします．

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In natural language processing, syntactic parsing is an important first step for various natural language understanding applications [32, 31, 26]. However, developing a high-quality parser is costly, since it requires a large amount of annotated syntactic trees by linguistic experts.

A promising direction to reduce the annotation effort is domain adaptation. It is well known that the parser performance largely degrades with a domain shift. For many languages it is typical that we have abundant annotations, i.e., a treebank, in a particular domain, commonly in the newswire domain. Using such a high-resource treebank along with a small size of annotated target domain data, domain adaptation aims at improving the performance of the target domain. For parsing, one can also assume that parser transfer across languages is a kind of domain adaptation; that is, when we want to develop a parser for a new (target) language, we may utilize the existing high-resource treebank for some language, possibly typologically similar to the target one. The recent effort of Universal Dependencies [25] opens up the opportunity for such study, as all languages are annotated in a consistent way with the same annotation scheme. Thus, parser domain adaptation is an attractive way for creating a new parser in many scenarios.

Most recent state-of-the-art parsers employ neural networks, in particular bi-directional LSTMs (Bi-LSTMs) [17, 10, 8, 28]. Contrary to the ordinary feature-based systems that can only see local contexts [22], such modern parsers are able to exploit the entire sentence for score calculation. For example, the current state-of-the-art dependency parser [10] is a simple arc-factored model, in which each dependency arc score is obtained with an attention between the head and dependent word representations, both come from the output of Bi-LSTMs.

In this thesis, we present an empirical study to explore a better domain adaptation technique for such modern neural parsers. Inspired by the traditional domain adaptation technique for feature-based models [9, 12], in which augmented domain-general features absorb the domain differences, we try to capture the domain-general in addition to the domain-specific word representation via separate LSTMs, which are then combined with a context-dependent gate. One concern with this basic architecture is that there is no explicit mechanism to encourage the domain-general LSTMs to learn the generalizable features across domains. We thus explore the mechanism of adversarial training on the domain-general LSTMs which, with an adversarial domain classifier, makes it generalizable as much as possible. We focus in our experiments on the current state-of-the-art neural dependency parser [10] that we introduce next, though we expect the lessons from our study would be useful for the other and future parsers with the similar architectures.

Our attention in this thesis is limited on *supervised* domain adaptation, in which a small amount of annotated trees on a target domain is available. Fully unsupervised adaptation that relies on a large set of unlabeled text is also popular [20, 37] but in practice with such approaches the improvements from the source-only model are rather modest. The supervised, or semi-supervised setting is thus important for parser adaptation, for which a parser needs to deal with both lexical and grammatical changes. This setting has previously been enjoyed in a shared task [27]. Utilizing additional unlabeled text is nevertheless important, which we leave as our future work.

We evaluate our technique on Universal Dependencies (UD) 2.0 [25], a recently released large collection of treebanks. This allows us to evaluate our domain adaptation technique cross-linguistically, since for several languages it contains a variety of treebanks with varying data sizes and properties. For example, there are three variants of English treebanks, *en*, *en_lines*, and *en_partut*; notably the training data sizes are quite divergent: *en* contains 12,543 sentences while *en_lines* and *en_parunt* contain only 2,738 and 1,090 sentences, respectively. We regard the differences between these treebanks as domain changes, and consider the dominant treebank (e.g., *en*) as a source domain while the others as the target domains.[1]

[1]The differences between treebanks are not the domain differences in the standard sense, in that in UD the main source of distinction between treebanks is the variety of annotation projects, rather than the domains of the texts. For example the original English treebank (*en*) is the Web treebank [33], which itself comprises of several subdomains such as blog posts and e-mails. English-LinES (*en_lines*), on the other hand, is based on an English-Swedish parallel treebank project [1], which also comprises of several subdomains. Though these observations suggest that our adaptation across treebanks slightly differs from the ordinary domain adaptation we can also expect there are some notable differences across

For cross-lingual adaptation, our main interest is which language pair we should select for training jointly. In other words, for improving the parser of a low resource language, does our typological knowledge help to obtain a better model? UD is again the best platform for pursuing this question thanks to its consistent annotation format across languages and typological variety. For a number of languages with small data sizes, we pick up some source language treebanks, one from English, and others from its typologically similar languages, and find that the typologically close languages consistently lead to better performances.

Our main contribution is empirical, rather than theoretical. From the machine learning perspective, our network architecture itself is not entirely novel. In the context of multi-task learning, recently conceptually similar approaches to us with adversarial training have been proposed [18, 6]. However, an extensive evaluation of the architecture for parser domain adaptation with a comparison to other strong baselines, such as fine-tuning [19], has not been reported yet. We also investigate a linguistically interesting question of whether our typological knowledge helps for better adaptation on the presented architecture, and we find the answer is yes.

data; for example some genres such as literary texts are only found in *en_lines* and the translation texts are known to have some own characteristics [29]. We thus simply follow the distinction in UD and assume that each treebank amounts to a particular domain in this work.

Chapter 2

Related Work

2.1 Dependency Parsing

The Universal Dependencies (UD) project [38] provides us the dependency parsing data across languages. This allows us to evaluate our domain adaptation technique cross-linguistically. There is a cross-lingual syntactic transfer research [30]. They consider the problem of cross-lingual syntactic transfer with limited resources.

Recent years have seen exciting developments in cross-lingual linguistic structure prediction based on transfer or projection of POS and dependencies [21]. These works mainly use supervised learning and domain adaptation techniques for the target language. Some studies focus on unsupervised dependency parsing [2].

In our work, we focus on supervised dependency parsing and discuss the relevance of domain adaptation works in next section.

2.2 Domain Adaptation

Domain Adaptation is a method to learn classifiers for which the training and test samples are from different distributions. Some approaches perform this by selecting samples from the source domain[15]. In the other hand, other studies try to compute explicit feature space transformation to convert source distribution to target. An important aspect of the distribution matching method is the way to measure the similarity between different distributions.

Adversarial Domain Adaptation[13] is a promising domain adaptation method. By modifying the feature representation, this approach learns domain-invariant between

different distributions. This method uses a way to measure similarity between different distributions by using discriminatively-trained classifier. This method can be implemented by introducing the gradient reversal layer (GRL). More details will be described in Section 4.2.

Inspired by the success of adversarial training in the machine learning and vision community [14, 13, 3], recently the technique has started to be explored on natural language processing (NLP) applications as well [18, 6]. In cross-lingual sentiment classification task, Adversarial domain adaptation method is applied to learn language-invariant features in neural networks[7].

In our work, we propose network architecture for cross-lingual dependency parsing task. To our best knowledge, this is first work to applly adversarial domain adaptation technique for cross-lingual dependency parsing.

Chapter 3

Biaffine Attention Model

We first introduce the *biaffine* attention model [10], on which we build our domain adaptation techniques in the next section. This model is an extension to the recently proposed arc-factored dependency parser calculating the score of each arc independently from the representations of two tokens obtained by bi-directional LSTMs (Bi-LSTMs) [17]. For labeled dependency parsing, this model first predicts the best *unlabeled* dependency structure, and then assigns a label to each predicted arc with another classifier. For the first step, receiving the word and POS tag sequences as an input, the model calculates the score of every possible dependency arc. To obtain a well-formed dependency tree, these scores are fed into the Chu-Liu/Edmonds algorithm [11], which finds the possibly non-projective tree with the highest total score of all arcs. The overview of the biaffine model is shown in Figure 3.1.

Let w_t be the t-th token in the sentence. As an input the model receives the word embedding $\mathbf{w}_t \in \mathbb{R}^{d_{word}}$ and POS embedding $\mathbf{p}_t \in \mathbb{R}^{d_{pos}}$ for each w_t, which are concatenated to a vector $\mathbf{x}_t$. This input is mapped by Bi-LSTMs to a hidden vector $\mathbf{r}_t$, which is then fed into an extension of bilinear transformation called a *biaffine* function to obtain the score for an arc from w_i (head) to w_j (dependent):

$$\mathbf{r}_t = \text{Bi-LSTM}(\mathbf{x}_t), \tag{3.1}$$

$$\begin{aligned} \mathbf{h}_i^{(arc-head)} &= \text{MLP}^{(arc-head)}(\mathbf{r}_i), \\ \mathbf{h}_j^{(arc-dep)} &= \text{MLP}^{(arc-dep)}(\mathbf{r}_j), \\ s_{i,j}^{(arc)} &= \mathbf{h}_i^{\text{T}(arc-head)} U^{(arc)} \mathbf{h}_j^{(arc-dep)} \\ &\quad + \mathbf{h}_i^{\text{T}(arc-head)} \mathbf{u}^{(arc)}, \end{aligned} \tag{3.2}$$

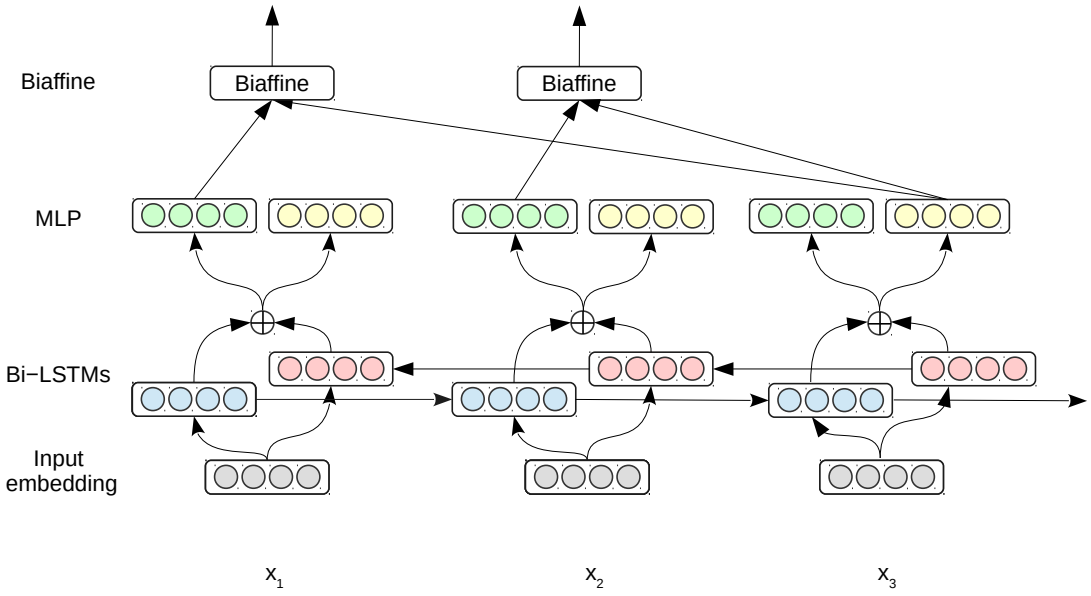

Figure 3.1: Overview of the biaffine model.

where MLP is a multi layer perceptron. A weight matrix $U^{(arc)}$ determines the strength of a link from w_i to w_j while $\mathbf{u}^{(arc)}$ is used in the bias term, which controls the prior headedness of w_i.

After obtaining the best unlabeled tree from these scores, we assign the best label for every arc according to $\mathbf{s}_{i,j}^{(label)}$, in which the k-th element corresponds to the score of k-th label:

$$\begin{aligned}
\mathbf{h}_i^{(label-head)} &= \mathrm{MLP}^{(label-head)}(\mathbf{r}_i), \\
\mathbf{h}_j^{(label-dep)} &= \mathrm{MLP}^{(label-dep)}(\mathbf{r}_j), \\
\mathbf{h}_{i,j}^{(label)} &= \mathbf{h}_i^{(label-head)} \oplus \mathbf{h}_j^{(label-dep)}, \\
\mathbf{s}_{i,j}^{(label)} &= \mathbf{h}_i^{\mathrm{T}(label-head)} \mathbf{U}^{(label)} \mathbf{h}_j^{(label-dep)} \\
&\quad + \mathbf{h}_{i,j}^{\mathrm{T}(label)} W^{(label)} + \mathbf{u}^{(label)},
\end{aligned} \tag{3.3}$$

where $\mathbf{U}^{(label)}$ is a third-order tensor, $W^{(label)}$ is a weight matrix, and $\mathbf{u}^{(label)}$ is a bias vector. We jointly train the parameters for arc score $\mathbf{s}^{(arc)}$ and for label score $\mathbf{s}^{(label)}$ by optimizing the sum of their softmax cross-entropy losses.

Chapter 4

Domain Adaptation

Now we describe our approach to domain adaptation, which relies on two important concepts: separate layers of LSTMs for capturing domain-specific and domain-general representations, and adversarial training, which strengthens the generalizability of the domain-general LSTMs.

The overall architecture is depicted in Figure 4.1, which shows how each word representation $\mathbf{r}_t$ in Eq. 3.1 is obtained in our network. Each token on the input sentence is passed to the biaffine model through this network. Here we present our full network architecture in order. We examine the effects of a specific component in this network with an ablation study in the next section.

4.1 Domain-specific and domain-general LSTMs

A popular approach to domain adaptation in the classical feature-based models is tying model parameters across domains by introducing domain-general features in addition to domain-specific ones [9, 12]. Inspired by the success of these works, we introduce an additional LSTM layer (shared LSTMs) that we expect to capture the domain invariant syntactic roles of a word.

Our approach can be applied for multi-domain adaptation with d domains (one for the source, and $d-1$ for the target domains). In this case the model consists of $d+1$ LSTM layers in total, one of which is domain-general shared by all d domains.

The intuitive idea on the role of domain-general LSTMs is that the syntactic roles of some specific words, or some POS tags, do not change across domains while others do, and they learn such domain-general representations of common terms. The final

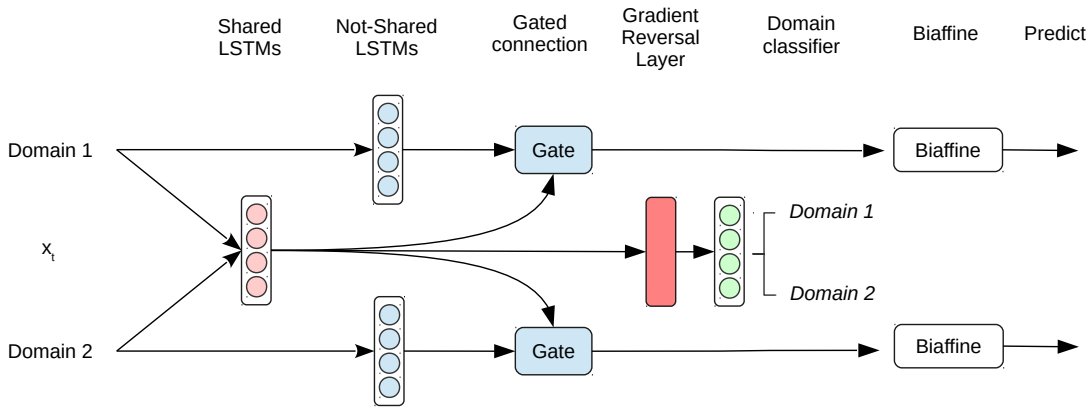

Figure 4.1: The architecture of our domain adaptation.

word representation $\mathbf{r}_t$ fed into the biaffine function is obtained by mixing them with a gate that balances two different LSTM outputs:

$$\begin{aligned} \mathbf{g}_t &= \sigma(U^{(gate)}(\mathbf{r}_t^{dom} \oplus \mathbf{r}_t^{share}) + \mathbf{u}^{(gate)}), \\ \mathbf{r}_t^{gate} &= \mathbf{g}_t \cdot \mathbf{r}_t^{share} + (1 - \mathbf{g}_t) \cdot \mathbf{r}_t^{dom}, \end{aligned}$$

where σ is the sigmoid function. $\mathbf{r}_t{}^{share}$ is the output of the shared Bi-LSTMs while $\mathbf{r}_t{}^{dom}$ is the output of the domain-specific Bi-LSTMs. To make the mixing ratio $\mathbf{g}_t$ context dependent, we take a simple approach and use the LSTM outputs themselves for this gating calculation.

4.2 Adversarial Training on Domain-general LSTMs

We additionally utilize adversarial training [13], which we introduce for making the domain general LSTMs to be truly general across domains. The key of this mechanism is a domain classifier, which receives $\mathbf{r}_t^{share}$, and tries to classify the domain of the input word. During training, the classifier is trained to correctly classify the domain of the input. At the same time, we train the domain-general LSTMs so that the domain classification task becomes harder. By this *adversarial* mechanism, we expect the domain-general LSTMs to be not specific to a particular domain as much as possible, while the domain-specific LSTMs can focus more on capturing domain-specific syntactic knowledge of a word.

This technique can be implemented by introducing the gradient reversal layer (GRL). The GRL has no parameters associated with it (apart from the hyper-parameter λ, which is not updated with backpropagation). During forward propagation GRL acts as an identity transform, while during backpropagation GRL takes the gradient from the subsequent layer, multiplies it by λ and passes it to the preceding layer.

The parameter λ controls the trade-off between the two objectives (the domain classifier and the biaffine model) that shape the feature representation during training. The GRL layer $R_\lambda(\mathbf{x})$ for forward and backward propagation is defined as follows:

$$R_\lambda(\mathbf{x}) = \mathbf{x}; \quad \frac{dR_\lambda}{d\mathbf{x}} = -\lambda\mathbf{I}.$$

The final loss function of our model L can be written as:

$$L = L_{parser} + \lambda L_{domain},$$

where L_{parser} is the biaffine model loss and L_{domain} is the domain classifier loss defined by the softmax cross-entropy.

4.3 Separate output layers

The final design choice in our architecture is whether we share the same network parameters associated with the biaffine score function (above $\mathbf{r}_t$ in Figure 3.1, e.g., $\text{MLP}^{(arc-head)}$ and $U^{(arc)}$) or not. Sharing the same output layer means that we can increase the training data for learning it, with a burden that every level of domain-specific parser behavior must be managed at the LSTM layers, which may lose the flexibility. In our experiments below, we separate the output layers for each task in default, and later present an ablation study to show that this separation is important for improving the performance.

Chapter 5

Experiment

5.1 Settings

All experiments are performed on Universal Dependencies 2.0 [25] used in the CoNLL 2017 shared task [38].

For evaluation, apart from the convention that assumes gold preprocessing on sentence splitting, tokenization, and POS tagging, we follow the new standard employed in the CoNLL shared task and evaluate the parser performance against the raw input text. We perform all preprocessing using UDPipe [35] and report the official F1 LAS used in the shared task.

For the biaffine parameters we follow the settings in the original work [10], and use 3-layer 400-dimensional LSTMs for each direction, 500-dimensional MLP for arc prediction, and 100-dimensional MLP for label prediction. Word embeddings are initialized with 100-dimensional vectors pre-trained by word2vec [23], which are provided by the organizers of CoNLL 2017 shared task [38] and trained with CommonCrawl and Wikipedia for each language. POS embeddings are 100-dimensional and randomly initialized. We fix λ in the adversarial loss to 0.5, and apply dropout [34] with a rate of 0.33 at the input and output layers. For optimization we use Adam [16] with the batch size of 64 and gradient clipping of 5. We use early stopping [4] based on the performance on the development set. Our system is implemented with Chainer [36].

5.2 Cross-Domain Experiment

We first see the effects of our full architecture for domain adaptation (Figure 4.1) by comparing it with several competitive baselines across languages. As we mentioned in the introduction, some languages in UD are divided into two or three different treebanks, and we regard each treebank as a different domain. We only pick up such languages with multiple treebanks. We assume the multi-domain scenario and when there are three treebanks we jointly train with all of them. Note that for some languages (e.g., Ancient Greek; grc) all treebanks are equally large (Table 5.1). While this is a non-standard setting as domain adaptation, in this study we blindly apply each technique using all data regardless of data sizes.

We compare the following single systems and domain adaptation methods:

- UDPipe: An off-the-shelf transition-based neural parser used as the official baseline system in the CoNLL shared task [35]. We run it on each single treebank without adaptation. We add this baseline to compare our approach with an existing system.
- Biaffine: The single biaffine model [10] without adaptation. This can be seen as our *target only* baseline, which we train independently on each treebank.
- SRC-ONLY: The *source only* baseline. For each language, we pick up the largest treebank as a source domain[1] and run the model of it on the other domains.
- FINETUNE: A popular model transfer method for neural networks [19, 24], which is also employed in the top system of the CoNLL shared task [5]. We first train a biaffine model on the source domain and retrain it on the target domain.
- ALL: A simple adaptation technique often employed as a baseline [12], for each language on the union of all treebanks of that language.
- ADV: Our full network architecture (Figure 4.1).

The main results on the test data are summarized in Table 5.1. Asterisks * indicates source treebanks for SRC-ONLY and FINETUNE. The overall picture is that 1) our

[1] When the difference of treebank sizes is relatively small, we regard the one without a subscript as a source domain. This is the case for Spanish (es), Finnish (fi), Ancient Greek (grc), and Portuguese (pt). We mark the source treebanks with an asterisk (*) in Table 5.1.

baseline biaffine parser outperforms UDPipe in most treebanks, and 2) all the adaptation techniques using the target treebanks (FINETUNE, ALL, and ADV) bring some score improvements from the biaffine baseline, of which our ADV performs the best with only a few exceptions; it is in fact only on Russian (ru and ru_syntagrus) that our ADV is outperformed by other systems with a non-negligible score gap.

We also note that the score gap between SRC-ONLY and the biaffine model (target only) is quite large in many languages, suggesting that there are serious domain shifts between treebanks in this dataset.

An interesting observation is that ADV as well as ALL improve the performances of the source domains with larger treebanks. This is an advantage of our approach over FINETUNE, which can only operate on the target domains. Comparing three approaches in more detail, we find FINETUNE and ADV work particularly better than ALL for the target treebanks with less amount of data (e.g., en_lines, en_partut, fr_partut, among others), and the best result is often obtained by ADV. These observations suggest our ADV architecture is a better domain adaptation technique for syntactic parsers.

Language	# sentences	LAS					
		UDPipe	Biaffine	SRC-ONLY	FINETUNE	ALL	ADV
cs*	68,495	82.87	86.18	-	-	86.81	**87.19**
cs_cac	23,478	82.46	85.28	84.98	86.79	**87.06**	87.04
cs_cltt	465	71.64	73.10	70.19	80.23	79.51	**80.75**
en*	12,543	75.84	78.04	-	-	77.96	**78.05**
en_lines	2,738	72.94	75.76	69.56	76.81	75.58	**77.22**
en_partut	1,090	73.64	75.10	71.94	78.89	77.62	**78.94**
es*	14,187	81.47	84.23	-	-	84.65	**84.68**
es_ancora	14,305	83.78	87.24	72.71	87.12	87.27	**87.31**
fi*	12,217	73.75	73.53	-	-	74.31	**75.04**
fi_ftb	14,981	74.03	71.94	40.92	73.02	73.42	**74.59**
fr*	14,553	80.75	82.54	-	-	82.57	**82.58**
fr_partut	620	77.38	78.24	75.76	81.31	78.10	**82.25**
fr_sequoia	2,231	79.98	81.92	76.85	83.57	81.45	**83.83**
gl*	2,276	77.31	79.77	-	-	79.72	**80.02**
gl_treegal	600	65.82	66.17	51.75	67.28	64.65	**67.71**
grc*	11,476	56.04	58.51	-	-	62.21	**62.23**
grc_proiel	14,846	65.22	67.01	44.19	67.00	68.66	**68.82**
la	1,334	43.77	43.83	35.69	46.51	51.38	**51.87**
la_ittb*	15,808	76.98	81.60	-	-	81.58	**82.10**
la_proiel	14,192	57.54	59.50	34.33	60.06	61.41	**61.50**
nl*	12,330	68.90	73.08	-	-	73.16	**74.22**
nl_lassysmall	6,041	78.15	83.81	61.38	84.33	84.10	**85.03**
no_bokmaal*	15,696	83.27	85.94	-	-	87.01	**87.10**
no_nynorsk	14,174	81.56	84.14	76.75	85.01	85.18	**85.21**
pt*	8,331	82.11	84.08	-	-	84.35	**84.72**
pt_br	9,664	85.36	**87.76**	70.68	87.72	87.30	87.74
ru	3,850	74.03	75.89	58.46	**79.72**	75.54	78.46
ru_syntagrus*	48,814	86.76	88.77	-	-	**89.29**	89.10
sl*	6,478	81.15	83.85	-	-	83.94	**84.31**
sl_sst	2,137	46.45	45.35	39.21	49.54	49.55	**50.14**
sv*	4,303	76.73	78.83	-	-	79.06	**79.49**
sv_lines	2,738	74.29	76.15	67.60	76.99	76.44	**77.11**
AVG.	-	74.12	76.16	-	-	77.21	**77.98**

Table 5.1: The main results of cross-domain adaptation on the test data.

5.3 Cross-Lingual Experiment

We next investigate whether our ADV can learn the cross-linguistically generalizable syntactic knowledge for parser transfer. We first focus on the five low-resource treebanks in UD 2.0, which contain less than 1,000 training sentences (See Table 5.2). Here, our main research question is which language we should choose as a source language given a target language.

Language	# sentences
Hungarian (hu)	910
Ukrainian (uk)	863
Uyghur (ug)	100
Kazakh (kk)	31
Irish (ga)	566
Finnish (fi)	12,217
English (en)	12,543
Russian (ru_syntagrus)	48,814
Turkish (tr)	3,685

Table 5.2: The number of training sentences for the cross-lingual experiment.

Table 5.3 summarizes the results, in which we choose two different treebanks as a source: one is a typologically close language to the target language, and another is English. Irish (ga) is a Celtic language, a member of Indo-European family, but no other languages belong to it. We thus only experiment with English as a source language. We can see that for ADV, choosing the source from the same family consistently leads to better performance than using English. With these source languages, comparing with FINETUNE, ADV outperforms it in four out of five languages, suggesting that ADV is a better method for cross-lingual adaptation as well.

To understand the model behavior in the cross-lingual setting in more depth, below we present two types of controlled experiments on the development data.

Target	Training data	LAS	
		FINETUNE	ADV
hu	hu	-	60.68
	hu, fi	61.40	**61.52**
	hu, en	61.29	61.41
uk	uk	-	61.24
	uk, ru_syntagrus	**65.12**	64.14
	uk, en	62.69	62.99
ug	ug	-	32.05
	ug, tr	32.88	**33.61**
	ug, en	30.79	31.82
kk	kk	-	23.50
	kk, tr	22.79	**23.53**
	kk, en	21.67	21.71
ga	ga	-	61.80
	ga, en	62.83	**63.16**

Table 5.3: The main results of cross-lingual adaptation on the test data.

Learning curve One critical difference between cross-domain and cross-lingual adaptation is that in the former the basic syntactic knowledge can be preserved since they are the same *language*, while it is not the case in the latter. This suggests the model may not benefit from the source language when the target language treebank is sufficiently large to be able to learn the basic syntax with it. To verify this hypothesis, we present a learning curve in Figure 5.1 when choosing Spanish (es) as the target and French (fr) as the source language, both of which are a Romance language. LAS on Spanish development data in the monolingual setting (mono) and the cross-lingual setting (cross) with French as the source language. X-axis means the number of target (Spanish) training sentences. We use the full 14,553 sentences of French treebank, and increase the number of Spanish sentences from 100 to 14,187 (the entire data). Comparing to the scores of the monolingual parser without adaptation, we can see that the improvements with the help of French are remarkable when the training size is small ($\leq$ 1,000 sentences). It is interesting that modest score improvements are still observed when the target data size is 5,000 sentences (84.45 for cross vs. 84.03 for mono). Con-

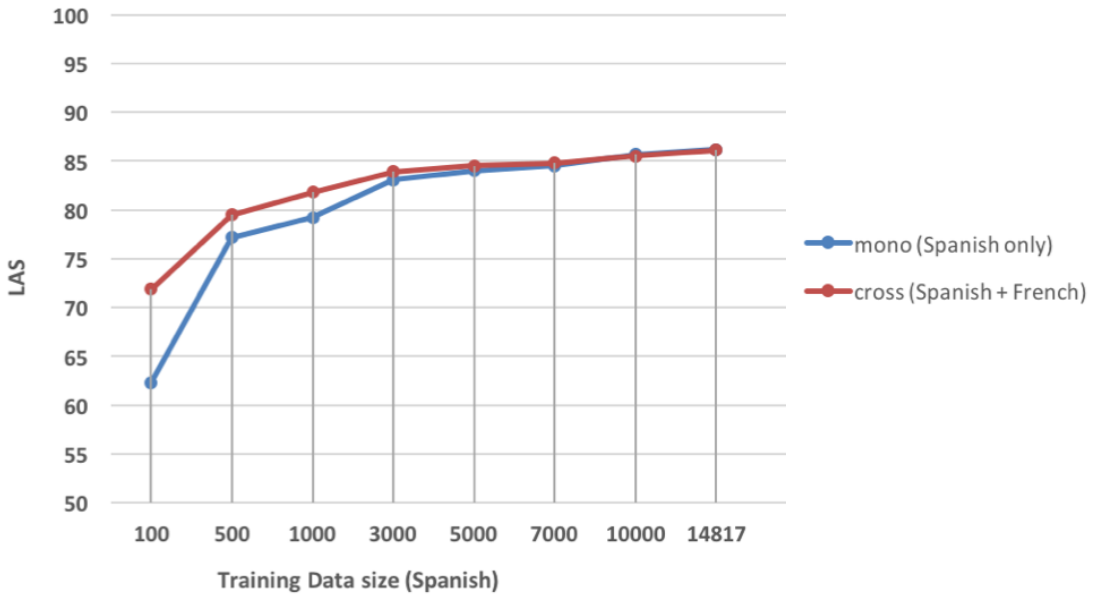

Figure 5.1: LAS on Spanish French in the cross-lingual setting.

sidering that 5,000 sentences seem sufficient to learn the basic word order and usage by themselves, we conjecture that our neural adaptation also helps to learn more subtle syntactic knowledge of the target language by best utilizing the source language.

More on language families In Table 5.3 we have shown that typological similarities are important for better transfer, but the comparison is limited to English only. Here we present a more systematic study to see how much language similarity is important under a controlled setting with the consistent training data sizes. We pick up three different language groups and select two languages from each, in total six languages: German and Dutch (Germanic), Czech and Polish (Slavic), and Arabic and Hebrew (Semitic), and try the all combinations of two languages from them. Table **??** shows the results, and it is clear that the best performance is always achieved with the same group language as the target. This confirms our first observations that our typological knowledge of the language can help to select a better partner.

5.4 Ablation Experiment

We finally evaluate how each component of our full architecture contributes for better adaptation. For this purpose we pick up three languages (English, Dutch, and French), and experiment on the networks without the separate output layers (sep-out), i.e., we share the biaffine parameters above LSTMs across domains, and without adversarial training (adv). See Table 5.5, where we can see that both components contribute to the scores of the full model in a varying degree depending on the language. Sep-out means the separated output layers and adv means the adversarial layer. A number in a parenthesis shows the difference from the full model. For example both treebanks of Dutch (nl) largely benefit from the separate output layers (-0.47 and -0.61), while adversarial training works a lot for nl_lassysmall (-0.8) and fr_partut (-0.78).

Language	LAS			
	Biaffine	w/o sep-out	w/o adv	Full
en	82.31 (-0.21)	82.37 (-0.15)	82.44 (-0.08)	**82.52**
en_lines	77.23 (-1.32)	78.44 (-0.11)	78.37 (-0.18)	**78.55**
en_partut	76.73 (-3.82)	80.40 (-0.15)	80.18 (-0.37)	**80.55**
nl	80.37 (-0.6)	80.50 (-0.47)	80.85 (-0.12)	**80.97**
nl_lassysmall	80.91 (-2.99)	83.29 (-0.61)	83.10 (-0.8)	**83.90**
fr	88.16 (-0.06)	88.16 (-0.06)	88.18 (-0.04)	**88.22**
fr_partut	84.43 (-3.04)	87.36 (-0.11)	86.69 (-0.78)	**87.47**
fr_sequoia	84.55 (-1.68)	86.05 (-0.18)	86.16 (-0.07)	**86.23**

Table 5.5: The ablation results on the development data in the cross-domain setting.

We also carry out a similar experiment on the cross-lingual setting, which we summarize in Table 5.6 for some selected language pairs from Table 5.2.[2] Again, we can see that both two components are essential. A different observation is that the score degradations are larger in this setting, in particular for the typologically close language pairs. For example, for Ukrainian (uk) and Uyghur (ug), when choosing the appropriate partner the degradations are more than 5.0 points when removing adversarial training, and more than 2.0 points without the separate output layers. Among these observations, it seems reasonable that adversarial training is important for close language pairs, as they would have larger syntactic commonalities to be exploited.

[2]The scores diverge from those of Table 5.3 since we evaluate on the development data for this study.

Language	LAS			
	Biaffine	w/o sep-out	w/o adv	Ours
hu, fi	63.18 (-0.86)	63.04 (-1.0)	63.73 (-0.31)	**64.04**
hu, en	63.18 (-0.57)	62.39 (-1.36)	63.48 (-0.27)	**63.75**
uk, ru_syntagrus	71.61 (-4.54)	73.23 (-2.92)	70.67 (-5.48)	**76.15**
uk, en	71.61 (-1.9)	73.08 (-0.43)	72.93 (-0.58)	**73.51**
ug, tr	48.09 (-6.11)	49.62 (-4.58)	48.85 (-5.35)	**54.20**
ug, en	**48.09** (+2.29)	45.04 (-0.76)	44.27 (-1.53)	45.80

Table 5.6: The ablation results on the development data in the cross-lingual setting.

Chapter 6

Conclusion

We have shown that for neural parser domain adaptation modeling the domain-general LSTM layers with adversarial training is beneficial for better knowledge transfer. We have also demonstrated that our architecture enables efficient cross-lingual adaptation, for which choosing the typologically close language to the target is crucial. Our future work includes modeling morphology of each word, which facilitates cross-lingual knowledge sharing at the word level, and investigating the better use of unlabeled text.

In our experiments, we obtain an interesting observation that our proposed method improves the performances of the source domains with larger treebanks. With these source languages, our proposed method outperforms the baseline (FINETUNE), suggesting that our method is a better method for cross-lingual adaptation as well.

We also present a systematic study to see how much language similarity is important under a controlled setting with the consistent training data sizes. It is clear that the best performance is always achieved with the same group language as the target in experiment of language families. This confirms our first observations that our typological knowledge of the language can help to select a better partner.

Appendix A

外部発表一覧

- Motoki Sato, Austin J. Brockmeier, Georgios Kontonatsios, Tingting Mu, John Y. Goulermas, Juníchi Tsujii and Sophia Ananiadou, “Distributed Document and Phrase Co-embeddings for Descriptive Clustering”, EACL 2017 (Long Paper).
- Motoki Sato, Hitoshi Manabe, Hiroshi Noji, Yuji Matsumoto, “Adversarial Training for Cross-Domain Universal Dependency Parsing”, CoNLL 2017, (Ranked 6th in the Shared Task).
- Motoki Sato, Hiroyuki Shindo, Ikuya Yamada, Yuji Matsumoto, “Segment-Level Neural Conditional Random Fields for Named Entity Recognition”, IJCNLP 2017 (Short Paper).
- Ikuya Yamada, Motoki Sato, Hiroyuki Shindo: “Ensemble of Neural Classifiers for Scoring Knowledge Base Triples”, WSDM Cup 2017 (Ranked 2nd in the competition).
- 佐藤 元紀, 進藤 裕之, 松本 裕治, “ラティス構造とニューラルネットワークに基づく系列ラベリング”, NLP 2017.
- 佐藤 元紀, 能地 宏, 松本 裕治, “deep-crf”, YANS 2017 (デモ賞受賞).

Bibliography

[1] L. Ahrenberg. Lines: An english-swedish parallel treebank. In *NoDaLiDa*, pp. 270–273, 2007.

[2] H. M. Alonso, Ž. Agic, B. Plank, and A. Søgaard. Parsing universal dependencies without training. In *EACL*, 2017.

[3] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett eds., *NIPS*, pp. 343–351. 2016.

[4] R. Caruana, S. Lawrence, and C. L. Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In T. K. Leen, T. G. Dietterich, and V. Tresp eds., *NIPS*, pp. 402–408. 2001.

[5] W. Che, J. Guo, Y. Wang, B. Zheng, H. Zhao, Y. Liu, D. Teng, and T. Liu. The hit-scir system for end-to-end parsing of universal dependencies. In *CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pp. 52–62, 2017.

[6] X. Chen, Z. Shi, X. Qiu, and X. Huang. Adversarial multi-criteria learning for chinese word segmentation. In *ACL*, pp. 1193–1203, 2017.

[7] X. Chen, Y. Sun, B. Athiwaratkun, C. Cardie, and K. Weinberger. Adversarial deep averaging networks for cross-lingual sentiment classification. *arXiv*, 2016.

[8] J. Cross and L. Huang. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *EMNLP*, pp. 1–11, 2016.

[9] H. Daume III. Frustratingly easy domain adaptation. In *ACL*, pp. 256–263, 2007.

[10] T. Dozat and C. D. Manning. Deep biaffine attention for neural dependency parsing. *ICLR*, 2017.

[11] J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240, 1967.

[12] J. R. Finkel and C. D. Manning. Hierarchical bayesian domain adaptation. In *NAACL-HLT*, pp. 602–610, 2009.

[13] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pp. 1180–1189, 2015.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, pp. 2672–2680, 2014.

[15] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, 2006.

[16] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv*, 2014.

[17] E. Kiperwasser and Y. Goldberg. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. *TACL*, 4:313–327, 2016.

[18] P. Liu, X. Qiu, and X. Huang. Adversarial multi-task learning for text classification. In *ACL*, pp. 1–10, 2017.

[19] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In F. Bach and D. Blei eds., *ICML*, Vol. 37, pp. 97–105, 2015.

[20] D. McClosky, E. Charniak, and M. Johnson. Effective self-training for parsing. In *NAACL-HLT*, pp. 152–159, 2006.

[21] R. T. McDonald, S. Petrov, and K. B. Hall. Multi-source transfer of delexicalized dependency parsers. In *EMNLP*, 2011.

[22] R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. Non-projective dependency parsing using spanning tree algorithms. In *HLT-EMNLP*, pp. 523–530, 2005.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pp. 3111–3119, 2013.

[24] L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, and Z. Jin. How transferable are neural networks in nlp applications? In *EMNLP*, pp. 479–489, 2016.

[25] J. Nivre, Ž. Agić, L. Ahrenberg, et al. Universal Dependencies 2.0, 2017. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague.

[26] N. Peng, H. Poon, C. Quirk, K. Toutanova, and W.-t. Yih. Cross-sentence n-ary relation extraction with graph lstms. *TACL*, 5:101–115, 2017.

[27] S. Petrov and R. McDonald. Overview of the 2012 Shared Task on Parsing the Web. In *NAACL Workshop on SANCL*, 2012.

[28] P. Qi and C. D. Manning. Arc-swift: A novel transition system for dependency parsing. In *ACL*, pp. 110–117, 2017.

[29] E. Rabinovich, S. Nisioi, N. Ordan, and S. Wintner. On the similarities between native, non-native and translated texts. In *ACL*, pp. 1870–1881, 2016.

[30] M. S. Rasooli and M. Collins. Cross-lingual syntactic transfer with limited resources. *TACL*, 5:279–293, 2017.

[31] S. Reddy, Oscar, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, M. Lapata. Transforming dependency structures to logical forms for semantic parsing. *TACL*, 4:127–140, 2016.

[32] S. Riedel, L. Yao, A. McCallum, and B. M. Marlin. Relation extraction with matrix factorization and universal schemas. In *NAACL-HLT*, pp. 74–84, 2013.

[33] N. Silveira, T. Dozat, M.-C. D. Marneffe, S. Bowman, M. Connor, J. Bauer, and C. Manning. A gold standard dependency corpus for english. In *LREC*, 2014.

[34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[35] M. Straka, J. Hajič, and J. Straková. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, pos tagging and parsing. In *LREC*, 2016.

[36] S. Tokui, K. Oono, S. Hido, and J. Clayton. Chainer: a next-generation open source framework for deep learning. In *NIPS Workshop on LearningSys*, 2015.

[37] J. Yu, M. Elkaref, and B. Bohnet. Domain adaptation for dependency parsing via self-training. In *IWPT*, pp. 1–10, 2015.

[38] D. Zeman, M. Popel, M. Straka, J. Hajič, J. Nivre, et al. CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 2017.