GENERALIZED DOMAIN ADAPTATION FOR SEQUENCE LABELING IN NATURAL LANGUAGE PROCESSING

A Dissertation
Submitted to
the Temple University Graduate Board

In Partial Fulfillment
of the Requirements for the Degree
DOCTOR OF PHILOSOPHY

---

by
Min Xiao
May 2016

Examining Committee Members:

Yuhong Guo, Advisory Chair, Dept. of Computer and Information Sciences
Zoran Obradovic, Dept. of Computer and Information Sciences
Slobodan Vucetic, Dept. of Computer and Information Sciences
Sining Chen, External Member, Dept. of Statistics and Learning Res., Bell Labs

ABSTRACT

Sequence labeling tasks have been widely studied in the natural language processing area, such as part-of-speech tagging, syntactic chunking, dependency parsing, and etc. Most of those systems are developed on a large amount of labeled training data via supervised learning. However, manually collecting labeled training data is too time-consuming and expensive. As an alternative, to alleviate the issue of label scarcity, domain adaptation has recently been proposed to train a statistical machine learning model in a target domain where there is no enough labeled training data by exploiting existing free labeled training data in a different but related source domain. The natural language processing community has witnessed the success of domain adaptation in a variety of sequence labeling tasks.

Though the labeled training data in the source domain are available and free, however, they are not exactly as and can be very different from the test data in the target domain. Thus, simply applying naive supervised machine learning algorithms without considering domain differences may not fulfill the purpose. In this dissertation, we developed several novel representation learning approaches to address domain adaptation for sequence labeling in natural language processing. Those representation learning techniques aim to induce latent generalizable features to bridge domain divergence to enable cross-domain prediction.

We first tackle a semi-supervised domain adaptation scenario where the target domain has a small amount of labeled training data and propose a distributed representation learning approach based on a probabilistic neural language model. We then

relax the assumption of the availability of labeled training data in the target domain and study an unsupervised domain adaptation scenario where the target domain has only unlabeled training data, and give a task-informative representation learning approach based on dynamic dependency networks. Both works are developed in the setting where different domains contain sentences in different genres. We then extend and generalize domain adaptation into a more challenging scenario where different domains contain sentences in different languages and propose two cross-lingual representation learning approaches, one is based on deep neural networks with auxiliary bilingual word pairs and the other is based on annotation projection with auxiliary parallel sentences. All four specific learning scenarios are extensively evaluated with different sequence labeling tasks. The empirical results demonstrate the effectiveness of those generalized domain adaptation techniques for sequence labeling in natural language processing.

To my parents.

ACKNOWLEDGEMENTS

I owe a debt of gratitude to many people, whom I relied on and learned a lot from. Many thanks go to my advisor, Dr. Yuhong Guo, who introduced me to this field and inspired me every aspect of research. Dr. Guo is a creative researcher with rigorous attitude and always gives me constructive comments and suggestions. Without her, this book cannot be done.

I would like to thank my other committee members, Dr. Zoran Obradovic, Dr. Slobodan Vucetic for giving me good feedback towards my work. I am grateful to Dr. Sining Chen for being my external committee member and mentoring me during my summer internship in Bell Labs. It was a very good time and I enjoyed every second working with her. Thanks also go to Dr. Alexander Yates, who taught me many things about NLP, and all other faculties and staff members in the department of computer and information sciences, who taught and helped me a great deal.

I would also like to thank my lab-mates, Xin Li and Feipeng Zhao, and all other friends. They have helped me and cheered me up so many times in so many ways.

Finally, I would like to thank my parents for raising me up and supporting me in every possible way. Nothing in the world can express my love for them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the natural language processing (NLP) area, sequence labeling tasks have been popularly studied such as part-of-speech (POS) tagging, syntactic chunking, dependency parsing and etc. The tasks themselves bear various different syntactic or semantic information, which can also be used to facilitate downstream applications such as machine translation, relation extraction or question answering. Those sequence labeling systems are usually developed on a large amount of labeled data via supervised learning. However, manually collecting labeled training data is too time-consuming and expensive. Recently, domain adaptation has been proposed to train a statistical machine learning model on a *target* domain where there is no enough labeled training data by exploiting labeled training data from a different but related *source* domain. Empirically studies have justified those learning techniques in reducing annotation effort in the target domain for various sequence labeling tasks.

1.1 Sequence Labeling in Natural Language Processing

Sequence labeling is the task of assigning a sequence of categorical labels to an observed sentence such as part-of-speech tagging, syntactic chunking, named entity recognition, dependency parsing, and etc [46, 83].

1.1.1 Part-of-speech Tagging

Part-of-speech (POS) tagging aims to assign a syntactic POS tag to each individual token in a sentence [17, 11, 17, 65], which is of great importance as POS tags are widely used as input features to boost performance for high-level NLP applications such as chunking, named entity recognition, or dependency parsing. Figure 1.1 gives a sequence of words labeled with POS tags from Penn Treebank [49].

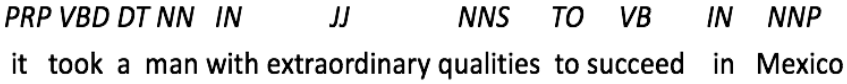

FIGURE 1.1: A sequence of words labeled with part-of-speech tags from Penn Treebank [49]. POS tags are shown in italic type.

1.1.2 Syntactic Chunking

Syntactic chunking aims to divide a sentence into several syntactically correlated segments of words, such as noun phrase (NP) or verbal phrase (VP) [77]. Here is an example of chunking from the Wall Street Journal (WSJ) corpus [49]: "[NP He] [VP reckons] [NP the current account deficit] [VP will narrow] [PP to] [NP only # 1.8 billion] [PP in] [NP September] ." Phrase chunking can also be viewed as a sequence labeling problem as it assigns a phrase chunk tag to each individual token (see Figure 1.2).

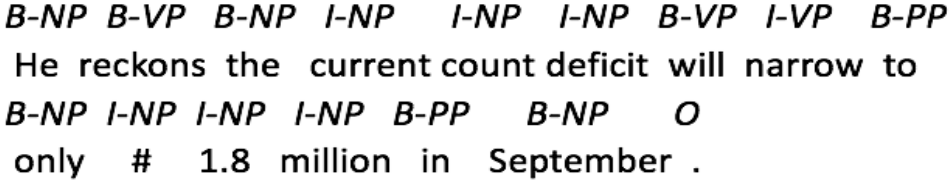

FIGURE 1.2: A sequence of words labeled with chunk tags from the WSJ corpus [49]. The chunk tags are shown in italic type. *B-* meas the beginning of this chunk, *I-* means the remaining part of this chunk, and *O* stands for items outside of chunks.

1.1.3 Named Entity Recognition

Similar as chunking, named entity recognition (NER) aims to segment words into different named entity phrases, which may describe an organization, a location, or a person [22, 82, 81]. The following example, " [PER Wolff] , currently a journalist in [LOC Argentina] , played with [PER Del Bosque] in the final years of the seventies in [ORG Real Madrid] ." [76], can be also represented as sequence labeling (see Figure 1.3).

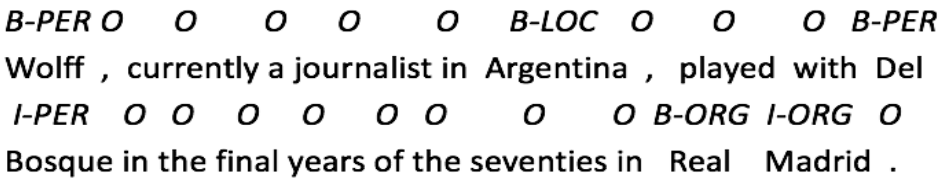

FIGURE 1.3: A sequence of words labeled with named entity tags [76]. The named entity tags are shown in italic type.

1.1.4 Dependency Parsing

Dependency parsing aims to infer a dependency graph for an observed sentence, where each node corresponds to a word in the sentence and each arc represents the dependency relationship between two words [50, 52]. For example, in Figure 1.4, each observation word is connected with a parent word via the dependency arc. Thus, we can view it as a sequence labeling problem as we label each observation word with its parent word. Equivalently, we can also use the parent word's position in this sentence as the label.

1.2 Generalized Domain Adaptation for Sequence Labeling

Though sequence labeling is very popular in NLP, they are usually developed via supervised learning using a large amount of labeled data. However, manually collecting labeled data is too time-consuming and expensive. Recently, domain adaptation has been proposed to alleviate that issue. Instead of manually labeling new

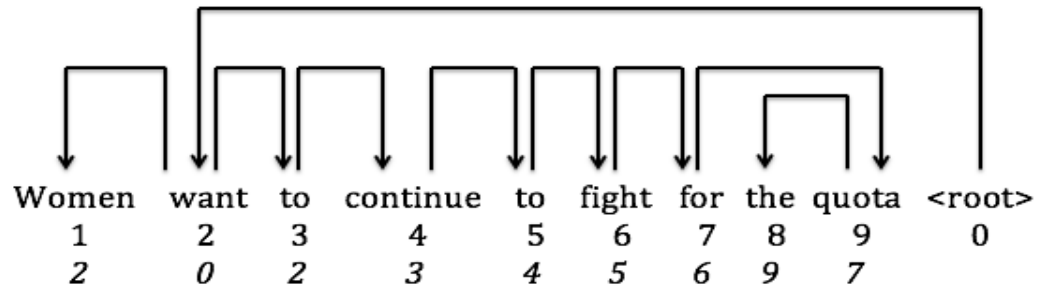

FIGURE 1.4: A sentence labeled with dependency relationships. The upper line shows the sequence of words, the middle line gives the positions of all words in this sentence, and the bottom line in italic type shows the position of its parent word for each observation word.

training instances, we may exploit existing free labeled instances from other sources [2, 3, 12, 23, 24]. For example, we want to develop a POS tagger for biomedical sentences but there is no enough labeled biomedical sentences. However, there are plenty of labeled newswire sentences. Thus, we can borrow them as the training resource. Another example is for cross-lingual learning. We aim to develop a dependency parser for Danish sentences and we do not have labeled Danish sentences. However, there is a large amount of free labeled English sentences. We then exploit them as the training data. Those two applications are two concrete examples of generalized domain adaptation we focus here. In this dissertation, we focus on generalized domain adaptation for sequence labeling where different domains contain sentences in different genres or in different languages. We exploit free labeled data in a label-rich source domain to train a sequence labeling system and use it for prediction in a label-rare target domain. Next, we briefly introduce four specific learning scenarios within generalized domain adaptation: a semi-supervised adaptation learning and a unsupervised adaptation learning for two different genre domains, two cross-lingual adaptation learning for two different language domains.

1.2.1 Semi-Supervised Domain Adaptation for Sequence Labeling

Semi-supervised domain adaptation assumes that there is a small amount of labeled training data in the target domain and the auxiliary source domain has a large amount of labeled training data. Besides, the two domains have plenty of unlabeled training data. By leveraging all those data sources, we aim to develop a sequence labeling system and apply it into the target domain. Empirical evaluations have justified the effectiveness of semi-supervised domain adaptation for different sequence labeling tasks in reducing target annotation costs [30].

1.2.2 Unsupervised Domain Adaptation for Sequence Labeling

Unsupervised domain adaptation assumes that the labeled training data exist only in the source domain and the target domain only has unlabeled training data. By combining labeled samples in the source domain and unlabeled data in the two domains, unsupervised domain adaptation aims to train a sequence labeling system and use it for prediction in the target domain. Comparing to semi-supervised domain adaptation, unsupervised domain adaptation is more practical and applicable in real-world applications. Many works have been developed in the literature with good empirical results for a variety of sequence labeling tasks [14, 1, 40, 41].

1.2.3 Cross-Lingual Adaptation for Sequence Labeling

Most previous works on domain adaptation for sequence labeling assume that different domains contain different genres of texts (e.g., newswrie sentences in the source domain and biomedical sentences in the target domain [14]) but all texts are in the same language. In this dissertation, we *generalize* the domain adaptation setting by assuming that texts of different domains are belong to different genres or even different languages. This *generalized domain adaptation* setting is very important in the NLP area as it also addresses a series of important cross-lingual sequence labeling tasks. Cross-lingual sequence labeling aims to develop a sequence labeler in a label-

rare target language by using labeled training data in a label-rich source language [53, 33, 72]. In particular, we studied two sub cases: cross-lingual adaptation with bilingual word pairs and cross-lingual adaptation with parallel sentences. Both of bilingual word pairs and parallel sentences are used as auxiliary resources to build connections between two different language domains to bridge the language domain divergence.

1.3 Challenges and Issues

In this work, we focus on generalized domain adaptation for sequence labeling tasks in the NLP area where different domains contain sentences in different genres or different languages. As is said, most competitive in-domain sequence labeling systems are developed via supervised learning with manually crafted lexical features [77, 76, 78]. However, those traditional lexical features may not perform well in the generalized domain adaptation setting. Typically, different genres may use different vocabularies to express meanings in the text corpora, which renders the lexical feature-based sequence labeling systems perform poorly on the out-of-domain test data. For example, the newswire sentences contain frequent terms like "CEO", "corporation" while the biomedical sentences contain frequent terms like "genomic", "metastases". Thus, a lexical feature-based POS tagger trained on the newswire sentences cannot correctly infer the POS tags for biomedical sentences. In general, the two domains have a *feature representation divergence*. To the extreme, the sentences in two different language domains could have disjoint representation spaces in terms lexical features.

Thus how to appropriately address those issues serves as a key part in designing sequence labeling systems in generalized domain adaptation. Recently, a variety of representation learning approaches have been proposed to induce latent generalizable features to bridge domain divergence and enable cross-domain prediction. Next, we present the background of representation learning for generalized domain adaptation.

1.4 Representation Learning for Generalized Domain Adaptation

In the generalized domain adaption setting for sequence labeling, there is a source domain $\mathcal{D}_s$ and a target domain $\mathcal{D}_t$. For simplicity, we only consider one source domain and one target domain, but we can analogously extend it to multiple domains. Each domain is defined as the distribution over the set of the instances from this domain. Let $\mathcal{X}_s$ be the set of instances from the source domain and $\mathcal{X}_t$ be the set of instances from the target domain. Let $\mathcal{Y}$ be the output set of the sequence labeler. For example, for the POS tagging, $\mathcal{X}_s$ is the set of all sentences from the source domain, $\mathcal{X}_t$ is the set of all sentences from the target domain, and $\mathcal{Y}$ is the set of all sequences of POS tags. Similarly, for the cross-lingual dependency parsing, $\mathcal{X}_s$ is the set of sentences in the source language (e.g, English) and the $\mathcal{X}_t$ is the set of sentences in the target language (e.g, Danish), and $\mathcal{Y}$ is the set of all dependency trees. Let $\mathcal{X} = \mathcal{X}_s \cup \mathcal{X}_t$ and f be the prediction function of the sequence labeling problem in the generalized domain adaptation setting such that $f : \mathcal{X} \rightarrow \mathcal{Y}$. Specifically, the generalized domain adaptation problem witnesses a set of labeled (and unlabeled) training examples, of which most of the labeled samples are from the source domain and the target domain contains a few or no labeled training data, and use those training samples to develop a good sequence labeler to tag test sentences from the target domain. A *representation* is defined as a set of features describing the sequence labeling problem, and *representation learning* is defined as finding a proper function g to map the instance to a set of features such that $g : \mathcal{X} \rightarrow \mathcal{Z}$, where $\mathcal{Z}$ is some suitable feature space. Instead of manually crafting features with the cost of human ingenuity, we can automatically learn those features by performing representation learning. We can now view representation learning for generalized domain adaptation as two steps. First, we automatically learn this function g by using all (labeled and unlabeled) training data from the two domains to induce latent features. Then, we can map each instance x to the latent feature space and train a desirable sequence labeler on

the *transformed* instances via standard machine learning.

Recent theoretical works show that a good feature representation is very important to develop a robust prediction model for the target domain [6, 5]. [6] presented a theoretical bound on the target domain error by using the Vapnik-Chervonenkis (VC) theory [80]. It justified that a good representation helps to achieve low error rate and small domain divergence.

As for the generalized domain adaptation for sequence labeling with representation learning, we can cut the human ingenuity effort of feature engineering by automatically discovering effective latent features. Moreover, the traditional lexical features may render the sequence labeling system not generalize well to the new target domain since they are too specific to the domains. By using both data from the two domains to conduct representation learning, the induced latent features could capture similarities across the two domains and improve the generalization capability of the trained sequence labeling systems.

1.5 Contributions

This dissertation focuses on generalized domain adaptation for NLP sequence labeling tasks where sentences of different domains are sampled from different genres or different languages. For different genre domains, we separately studied a semi-supervised domain adaptation, where target domain contains a few labeled training data, and an unsupervised domain adaptation, where target domain contains no labeled training data. Accordingly, we proposed two representation learning approaches. One is based on a probabilistic language adaptation model [85] and the other is based on dynamic dependency networks [89]. In addition, we extended domain adaptation into different language domains and presented two cross-lingual adaptation approaches, one uses auxiliary bilingual word pairs [87] and the other uses parallel sentences [88]. All approaches are extensively evaluated with empirical studies. The experimental results demonstrated the superior prediction capability of the proposed approaches

in the generalized domain adaptation.

1.6 Organization

The remainder of this dissertation is organized as follows. We present the semi-supervised domain adaptation with a probabilistic language adaptation model in Chapter 2 and the unsupervised domain adaptation with dynamic dependency networks in Chapter 3. We then generalize the domain adaptation into cross-lingual adaptation and present a cross-lingual representation learning with bilingual word pairs in Chapter 4, and a cross-lingual representation learning with parallel sentences in Chapter 5. We then conclude the work and provide discussions on the future work in Chapter 6.

CHAPTER 2

SEMI-SUPERVISED DOMAIN ADAPTATION FOR SEQUENCE LABELING

In this chapter, we focus on semi-supervised domain adaptation for sequence labeling where the target domain has a *small* amount of labeled data and plenty of unlabeled data. By exploiting auxiliary data resource especially the labeled training samples from the source domain, semi-supervised domain adaptation aims to develop a machine learning model for target test data. Next, we will first introduce the problem and survey related works, then we present the main approach and empirical studies. This chapter is based on the work published in the International Conference on Machine Learning (ICML) [85].

2.1 Introduction

Semi-supervised domain adaptation aims to deploy a prediction model in a label-scarce *target* domain where there is a small amount of labeled training data by exploiting information in a label-rich *source* domain where there is a large amount of labeled training data [29]. Semi-supervised domain adaptation has achieved success in various sequence labeling tasks in the natural language processing (NLP) area, like part-of-speech (POS) tagging [43, 30] and named entity recognition (NER) [43].

Typically, the source domain and the target domain usually have *very different vocabularies*, which renders the lexical feature-based NLP systems perform poorly on the target domain [6, 5]. For example, a statistical machine learning model for POS tagging systems based on lexical features trained on newswire data with frequent terms like "CEO", "corporation" cannot correctly infer POS tags for biomedical text with frequent terms like "metastases", "sequencing" or "genomic". Moreover, the learning machine based on lexical features may produce *inconsistent prediction functions* across domains. For example, "signaling" in "signaling that ..." from the Wall Street Journal (WSJ) domain is primarily as a present participle (VBG), but predominantly as a noun (NN) in "signaling pathway" from the MEDLINE domain. Recently, much work has been proposed to copy with those problems in order to improve the prediction performance for out-of-domain NLP systems, including instance-weighting based semi-supervised adaptation learning method [43] and feature augmentation based semi-supervised adaptation learning method [30].

In this chapter, we propose to port sequence labeling systems in NLP from a source domain to another different but related target domain by inducing distributed representations using a log-bilinear language adaptation (LBLA) model. It combines the advantages of representation learning methods from [14], which employ generalizable features across domains to reduce domain divergence, and feature augmentation based (semi-)supervised adaptation learning methods from [28, 29], which exploit domain-specific features from both domains as well to enable target domain learning capability. Specifically, the LBLA model simultaneously models the source distribution by using generalizable and source-specific features and the target distribution by using domain-independent and target-specific features from induced distributed representations. We then propose using two domain-specified mapping functions to map the original data to learned feature representations as augmenting features, which will be then incorporated into supervised sequence labeling systems. The proposed learning technique is empirically evaluated for cross domain POS tagging systems

on articles from WSJ and MEDLINE domains, syntactic chunking and name entity recognition systems on sentences from WSJ and Brown corpora, and is shown to outperform comparison methods.

2.2 Related Work

Semi-supervised domain adaptation has been studied in the literature for sequence labeling problems in the NLP area. [30] proposed an EA++ method to address semi-supervised domain adaptation for sequence labeling. The EA++ method is a semi-supervised extension of the EA method [28], which is a purely supervised domain adaptation method and proposed to identify source-specific features, target-specific features and common features by using only labeled training data in the two domains. [30] extended it into semi-supervised domain adaptation by adding additional unlabeled target training data and empirically investigated it with NLP sequence labeling tasks. [43] investigated instance weighting method for semi-supervised domain adaptation by assigning more weights to labeled source and target data, removing misleading training instances in the source domain, and augmenting target training instances with predicted labels. They empirical evaluated their method for cross domain part-of-speech tagging and named entity recognition to justify its efficacy.

Distributed representations are widely exploited in the natural language processing area [68, 15]. Recently, [7, 8] introduced the neural network language models and demonstrated how to combine neural network probability predictions with distributed representations for words in order to outperform standard n-gram models. [15] demonstrated that those learned distributed representations of symbols make sense linguistically. Other NLP tasks, such as syntactic chunking, named entity recognition[79], semantic role labeling [26], and parsing [68] also demonstrated the effectiveness of those learned distributed representations.

2.3 Representation Learning for Semi-Supervised Domain Adaptation

We consider the semi-supervised domain adaptation problem from a source domain $\mathcal{S}$ to a target domain $\mathcal{T}$. In the source domain, we have l_s labeled sentences $\{(X_i^s, Y_i^s)\}_{i=1}^{l_s}$ and u_s unlabeled sentences $\{(X_i^s)\}_{i=1}^{u_s}$ for $n_s = \ell_s + u_s$, where X_i^s is the ith input sentence, i.e., a sequence of words, $w_1, w_2, \ldots, w_{T_i}$, and Y_i^s is its corresponding label sequence, e.g., the sequence of POS tags. Similarly, in the target domain, we have l_t labeled sentences $\{(X_i^t, Y_i^t)\}_{i=1}^{l_t}$ and u_t unlabeled sentences $\{(X_i^t)\}_{i=1}^{u_t}$ for $n_t = \ell_t + u_t$. In this semi-supervised domain adaptation scenario, ℓ_t is much smaller than ℓ_s. Previous theoretic studies suggest that seeking for a proper feature representation is crucial to generalized domain adaptation [6, 5], which has also been empirically evaluated in the literature for semi-supervised domain adaptation [30]. In this chapter, we propose to tackle semi-supervised domain adaptation by learning generalizable distributed representations of words from sentence structures to address NLP sequence labeling problems.

Distributed representations, which are dense, low-dimensional, and continuous-valued, are called word embeddings [79]. Each dimension of the word embedding stands for a latent feature of the word, hopefully capturing useful semantic and syntactic regularities. The basic idea to learn a distributed representation is to link each word with a real-valued feature vector, typically by using neural language models. A sentence can thus be transformed into a sequence of these learned feature vectors. The neural language model learns to map the sequence of feature vectors to a prediction of interest, such as the conditional probability distribution over the target word given its previous context, and pushes the learned word features to form grammatical and semantic similarities [7, 8]. The advantage of this distributed representation method is that it allows the model to generalize well to sequences that do not appear in the training set, but are similar to training sequences with respect to their distributed representations [7]. The simplest neural language model is the log-bilinear language

(LBL) model [55], which performs linear predictions in the semantic word feature space. Despite its simplicity, the LBL model has been shown to outperform n-grams on a large dataset [55, 57]. Based on this simple language model, we present a log-bilinear language adaptation (LBLA) model below to learn adaptive distributed word representations for domain adaptation over sequence labeling tasks.

2.3.1 Log-Bilinear Language Adaptation Model

For simplicity, we use a common word vocabulary V for both domains though the two domains may have very different vocabularies. We adapt the log-bilinear language model to learn distributed representations across domains by simultaneously modeling two different but related data distributions in the source and target domains. The distributed representations are encoded as real-valued vectors for words in the vocabulary. We refer to the matrix with all word representation vectors as R and denote the representation vector for word w as $R(w)$. Motivated by [28], we split the representation vector into three parts to capture both domain-sharing and domain-specific properties of each word. Thus the representation vector for a word w can be expressed as

$$R(w) = [R^s(w); R^c(w); R^t(w)] \tag{2.1}$$

where $R^s(w)$ represents source-specific latent features, $R^c(w)$ represents common latent features, and $R^t(w)$ represents target-specific latent features. Naturally, we assume that the source domain contains no target-specific features and the target domain contains no source-specific features. In practice, we define two mappings, Φ^s and Φ^t, to map the observed source and target words to cross domain word embeddings.

$$\Phi^s(w) = [R^s(w); R^c(w); \mathbf{0}^t] \tag{2.2}$$

$$\Phi^t(w) = [\mathbf{0}^s; R^c(w); R^t(w)] \tag{2.3}$$

where $\mathbf{0}^t$ is the zero vector in the same size of $R^t(w)$ and $\mathbf{0}^s$ is the zero vector in the same size of $R^s(w)$. Note that Φ^s and Φ^t are different from those mapping functions

exploited in previous work on domain adaptation [28], since we propose to learn the latent features using a log-bilinear language model while they perform simple feature replication. Moreover, the generalizable and domain-specific features may be different in our proposed model, while they use two identical copies for both two parts.

The three-part distributed representation learning framework can explicitly model the relationship between two data sources through the common representation part, while still maintaining the unique semantic and syntactic information of each data source through the domain-specific parts. For example, a POS tagging task uses WSJ as the source domain and MEDLINE as the target domain. The word "signaling" in a sentence "signaling that ..." from the source domain is a verb (VBG), but it is a noun (NN) in a sentence "signaling pathway ..." from the target domain. This syntactic difference of the same word in two domains can be encoded in the domain-specific latent features in the distributed representation.

Recall that we have two sets of training sentences sampled from two domains, $\mathcal{S}$ and $\mathcal{T}$. The LBLA model thus includes a set of conditional distributions, $P_{\mathcal{D}}(w|h)$ of each word w given its previous $(n_c - 1)$ words (denoted as the context h), for each domain $\mathcal{D} \in \{\mathcal{S}, \mathcal{T}\}$,

$$P_{\mathcal{D}}(w|h;\theta) = \frac{\exp\left(-E_{\mathcal{D}}(w,h;\theta)\right)}{Z_{\mathcal{D}}(h;\theta)} \tag{2.4}$$

$$Z_{\mathcal{D}}(h;\theta) = \sum_{w'} \exp\left(-E_{\mathcal{D}}(w',h;\theta)\right)$$

where $E_{\mathcal{D}}(w,h;\theta)$ is a log-bilinear energy function,

$$E_{\mathcal{D}}(w,h;\theta) = -\hat{\Phi}^d(w)^T \Phi^d(w) - b_w \tag{2.5}$$

for $d \in \{s,t\}$ correspondingly, and it quantifies the compatibility of word w with context h in domain $\mathcal{D}$. b_w is the bias parameter used to capture the popularity of word w across domains. We refer the bias vector for the whole words as $\mathbf{b}$. $\hat{\Phi}^d(w)$ is the predicted representation vector for the target word w given its context h in the domain indexed by d, which can be computed by linearly combining the feature

vectors for the context words, such that

$$\hat{\Phi}^s(w) = \sum_{i=1}^{n_c-1} [C_i^s R^s(w_i); C_i^c R^c(w_i); \mathbf{0}^t] \tag{2.6}$$

$$\hat{\Phi}^t(w) = \sum_{i=1}^{n_c-1} [\mathbf{0}^s; C_i^c R^c(w_i); C_i^t R^t(w_i)] \tag{2.7}$$

where C_i^s, C_i^c, C_i^t are the position-dependent context weight matrices, for source-specific, common, and target-specific features respectively. Thus, the negated log-bilinear energy function $-E_{\mathcal{D}}(w, h; \theta)$ measures the similarity between the current word feature vector $\Phi^d(w)$ and the predicted feature vector $\hat{\Phi}^d(w)$.

Overall, the proposed LBLA model simultaneously models two sets of conditional distributions $P_{\mathcal{S}}(.; \theta)$ and $P_{\mathcal{T}}(.; \theta)$ respectively on the domains based on distributed feature representations. These two sets of distributions reflect both domain-sharing properties of the data, parameterized with $\{R^c, \{C_i^c\}, b_w\}$, and domain-specific properties of the data, parameterized with $\{R^s, \{C_i^s\}\}$ and $\{R^t, \{C_i^t\}\}$.

2.3.2 Training with Noise-Contrastive Estimation

We propose to train the LBLA model using noise-contrastive estimation (NCE) [36, 37], which has been recently introduced for training unnormalized probabilistic models, and is shown to be less expensive than maximum likelihood learning and more stable than importance sampling [9] for training neural probabilistic language models [57]. Assume that we have a source data distribution $P_{\mathcal{S}}(w|h)$ and a target data distribution $P_{\mathcal{T}}(w|h)$, which are the distributions of words occurring after a particular context h, on the source domain and the target domain respectively. We propose to distinguish the observed source data and the observed target data from noise samples, which are artificially generated by a unigram distribution. We denote the context-independent noise distribution as $P_n(w)$. We would like to fit the context-dependent model $P_{\mathcal{S}}(w|h; \theta)$ and $P_{\mathcal{T}}(w|h; \theta)$ to $P_{\mathcal{S}}(w|h)$ and for $P_{\mathcal{T}}(w|h)$.

We assume that observed samples appear k times less frequently than noise sam-

ples. Thus data samples come from the mixture distribution

$$\frac{k}{k+1}P_n(w) + \frac{1}{k+1}P_{\mathcal{D}}(w|h) \tag{2.8}$$

on the domain $\mathcal{D}$. Since we are fitting $P_{\mathcal{D}}(w|h,\theta)$ to $P_{\mathcal{D}}(w|h)$, we will replace $P_{\mathcal{D}}(w|h)$ with $P_{\mathcal{D}}(w|h;\theta)$. Then given a context h, the posterior probabilities that a sample w comes from the observed source data distribution and the observed target data distribution are

$$P_{\mathcal{S}}(D=1|w,h;\theta) = \frac{P_{\mathcal{S}}(w|h;\theta)}{kP_n(w) + P_{\mathcal{S}}(w|h;\theta)} \tag{2.9}$$

$$P_{\mathcal{T}}(D=1|w,h;\theta) = \frac{P_{\mathcal{T}}(w|h;\theta)}{kP_n(w) + P_{\mathcal{T}}(w|h;\theta)} \tag{2.10}$$

However, evaluating Eq (2.9) and Eq (2.10) is too expensive due to the normalization computation for $P_{\mathcal{D}}(w|h;\theta)$ (Eq. 2.4). To tackle this issue, instead of conducting explicit normalization, NCE treats the normalization constants as parameters and parameterizes the model with respect to learned normalization parameters $z^s(h), z^t(h)$ and an unnormalized distribution $P_{\mathcal{S}}(.|h;\theta^0)$ and $P_{\mathcal{T}}(.|h;\theta^0)$, such that

$$\log P_{\mathcal{S}}(w|h;\theta) = \log P_{\mathcal{S}}(w|h;\theta^0) + z^s(h) \tag{2.11}$$

$$\log P_{\mathcal{T}}(w|h;\theta) = \log P_{\mathcal{T}}(w|h;\theta^0) + z^t(h) \tag{2.12}$$

where $\theta = \{\theta^0, z^s(h), z^t(h)\}$ and θ^0 are the parameters of the unnormalized distribution.

To fit the context-dependent model to the data, given a context h, we simply maximize an objective $J_{\mathcal{D}}(h;\theta)$ for each domain $\mathcal{D} = \{\mathcal{S}, \mathcal{T}\}$. It is the expectation of $\log P_{\mathcal{D}}(D|w,h;\theta)$ under the mixture distribution of the noise and observed data samples,

$$J_{\mathcal{D}}(h;\theta) = kE_{P_n}\left[\log \frac{kP_n(w)}{kP_n(w) + P_{\mathcal{D}}(w|h;\theta)}\right] + E_{P_{\mathcal{D}}(.|h)}\left[\log \frac{P_{\mathcal{D}}(w|h;\theta)}{kP_n(w) + P_{\mathcal{D}}(w|h;\theta)}\right] \tag{2.13}$$

Since the conditional distributions for different contexts share parameters, these distributions can then be learned jointly by optimizing a global NCE objective, which is defined as the combination of weighted per-context NCE objectives in the two domains,

$$J(\theta) = \sum_{\mathcal{D}\in\{\mathcal{S},\mathcal{T}\}} \sum_{h_\mathcal{D}} P(h_\mathcal{D}) J_\mathcal{D}(h_\mathcal{D};\theta) \tag{2.14}$$

where $P(h_\mathcal{D})$ is the empirical context probability of $h_\mathcal{D}$ in domain $\mathcal{D}$.

In practice, given an observation word w in context h from domain $\mathcal{D}$, we generate k noise datapoints $x_1, x_2, \ldots, x_k$ using a unigram noise distribution $P_n(w)$ and consider an approximate objective $\hat{J}_\mathcal{D}(w,h;\theta)$ such that

$$\hat{J}_\mathcal{D}(w,h;\theta) = \log P_\mathcal{D}(D=1|w,h;\theta) + \sum_{i=1}^{k} \log(1 - P_\mathcal{D}(D=1|x_i,h;\theta)). \tag{2.15}$$

Its gradient can be computed as

$$\frac{\partial}{\partial\theta}\hat{J}_\mathcal{D}(w,h;\theta) = \frac{kP_n(w)}{kP_n(w) + P_\mathcal{D}(w,h;\theta)} \frac{\partial}{\partial\theta} \log P_\mathcal{D}(w|h;\theta) - \sum_{i=1}^{k} \frac{P_\mathcal{D}(x_i|h;\theta)}{kP_n(x_i) + P_\mathcal{D}(x_i|h;\theta)} \frac{\partial}{\partial\theta} \log P_\mathcal{D}(x_i|h;\theta) \tag{2.16}$$

Based on this, we then use an empirical global NCE objective for gradient computation in each iteration of a gradient ascent training procedure, which can be expressed as a sum of the generated approximate objectives for each context-word pair appeared in all sentences of the two domains, i.e.,

$$\hat{J}(\theta) = \sum_{i=1}^{n_s} \sum_{j=1}^{|X_i^s|} \hat{J}_\mathcal{S}(w_{ij}^s, h_{ij}^s;\theta) + \sum_{i=1}^{n_t} \sum_{j=1}^{|X_i^t|} \hat{J}_\mathcal{T}(w_{ij}^t, h_{ij}^t;\theta). \tag{2.17}$$

Here w_{ij}^s denotes the jth word of the sentence X_i^s, and h_{ij}^s denotes the context of w_{ij}^s, i.e, its previous $(n_c - 1)$ words in the sentence X_i^s; the same for w_{ij}^t and its context h_{ij}^t. The gradient of $\hat{J}(\theta)$ can be easily obtained by summing over the gradients of each context-word pair objective, which can be computed following Equation (2.16).

2.3.3 Semi-Supervised Domain Adaptation with Distributed Representations

After training the LBLA model, we obtain two feature mapping functions, Φ^s in the source domain and Φ^t in the target domain. Then for a sentence in the source domain $w_1, w_2, \ldots, w_T$, we use the feature mapping function Φ^s to map each word to a feature vector as augmenting features. Similarly, we produce an augmenting feature vector of each word in the target sentences using the feature mapping function Φ^t. Finally, we combine the labeled sentences from both domains, represented using both the original features and the augmenting features, to train supervised NLP systems such as POS tagging, syntactic chunking and named entity recognition, and apply these systems into the target domain.

2.4 Experiments

We conducted experiments to evaluate the proposed distributed representation learning approach for semi-supervised domain adaptation for sequence labeling tasks: part-of-speech tagging, syntactic chunking and named entity recognition. For each task, we compared the proposed LBLA method with the following three baseline methods and four domain adaptation methods: (1) *SRCONLY* – a baseline that conducts training only on the labeled source data; (2) *TGTONLY* – a baseline that train conducts training only on the labeled target data; (3) *ALL* – a baseline that conducts training on the labeled data from both domains; (4) *SCL* – the Structural Correspondence Learning (SCL) domain adaptation technique developed in [14]; (5) *LBL* – the method that uses LBL model to produce distributed representation features as augmenting features for NLP systems; (6) *EA* – the feature augmentation based supervised domain adaptation method developed in [28]; (7) *EA++* – the feature augmentation based semi-supervised domain adaptation method developed in [29].

Table 2.1: Empirical results in terms of error rates on semi-supervised domain adaptation for part-of-speech tagging.

Adaptation Models	Error Rates
SRCONLY	12.02%
TGTONLY	4.15%
ALL	5.43%
SCL	3.90%
LBL	3.58%
EA	3.61%
EA++	3.52%
LBLA	**3.09**%

2.4.1 Semi-Supervised Domain Adaptation for POS Tagging

For POS tagging, we use the same experimental setting as [28, 14]. The source domain contains articles from Wall Street Journal (WSJ), with 39,832 manually tagged sentences from sections 02-21 and 100,000 unlabeled sentences from a 1988 subset. The target domain contains biomedical articles from MEDLINE, with 1,061 labeled sentences and about 100,000 unlabeled sentences. Among the 1,061 biomedical sentences, we use the same 561 sentences as test data while keeping the rest 500 as training data from the target domain. The task is to assign each word with one POS tag from the tagset, which is the superset of the Penn Treebank POS tags [49]. Among the tags, two tags cannot be seen in the newswire articles, *HYPH* (for hyphens) and *AFX* (for common post-modifiers for biomedical entities such as genes). These two tags were introduced because of the importance of hyphenated entities in biomedical text, which are about 1.8% of the words in the 561 labeled sentences.

We build a vocabulary with all sentences from the source and target domain. In order to reduce the vocabulary size, we map lower frequency (0-2) words to a single unique identifier in our vocabulary and sole-digit words into a single unique identifier. On all processed sentences except the 561 biomedical sentences which we will keep as

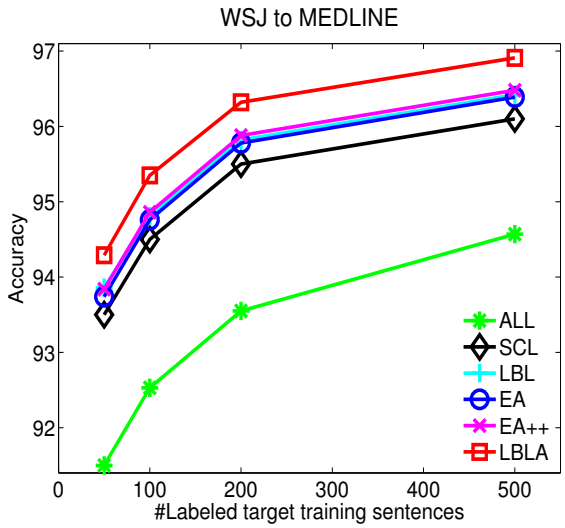

FIGURE 2.1: Empirical results with respect to the number of labeled training data in the target domain on semi-supervised domain adaptation for part-of-speech tagging.

test data, we applied the proposed log-bilinear language adaptation model to perform distributed representation learning. There are a few hyperparameters to be set when applying LBLA model. We set the word-embedding size for source-specific features, target-specific features and domain-sharing (common) features equally as 100. Thus the total size for a word embedding is 300. We set the context size n_c to be 3, which means we only consider the previous two words for each target word. We set k value for noise-contrastive estimation as 25. We randomly initialize the word embeddings R, the position-dependent context weight matrices $C = \{C_i^d : i \in \{1, 2\}, d \in \{c, s, t\}\}$, and initialize the bias vector $\mathbf{b}$ and the normalizing parameters $\{z^s(h), z^t(h)\}$ with all zeros. The same hyperparameters and initializations were used for LBL model as well. After augmenting each sentence with the learned representation features, standard supervised POS tagging were performed.

For supervised POS tagging, we used the SEARN algorithm, which is used in [28] as well. We used 39,832 labeled newswire sentences from the WSJ domain and 500 labeled biomedical sentences from MEDLINE domain as training data, while the test data contains 561 biomedical sentences with 14,554 tokens. Under this setting, the

Table 2.2: Empirical results in terms of error rates on semi-supervised domain adaptation for syntactic chunking.

Adaptation Models	Error Rates
SRCONLY	5.22%
TGTONLY	6.63%
ALL	4.33%
SCL	4.15%
LBL	3.86%
EA	3.97%
EA++	3.82%
LBLA	**3.30**%

test results of the comparison methods in terms of error rate are reported in Table 2.1. We can see that the LBLA method apparently outperforms all the other comparison methods on cross-domain POS tagging. We then conducted further experiments to investigate the performance of each method by varying the number of labeled training sentences from the target domain from 50 to 500. The test results in term of accuracy are plotted in Figure 2.1. We can see that the LBLA method consistently outperforms all the other comparison methods across the range of different number of training sentences from the target domain.

2.4.2 Semi-Supervised Domain Adaptation for Syntactic Chunking

For syntactic chunking, we use WSJ as the source domain and Brown corpus data as the target domain. We used the same source domain data as we did in POS tagging experiments. The target domain contains 3 sections (ck01-ck03) of Brown corpus data, with 426 labeled "general fiction" sentences and about 57,000 unlabeled sentences. Labeled sentences from both domains are tagged with syntactic chunking tags in IOB2 format, which is a standard format widely used for syntactic chunking. In IOB2 format, the chunk tags has two parts. The first part denotes the position of the corresponding token in the chunk and the second part represents the chunk

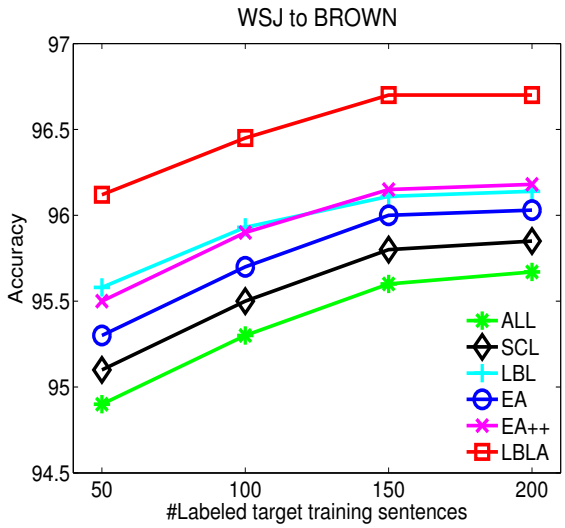

FIGURE 2.2: Empirical results with respect to the number of labeled target training data on semi-supervised domain adaptation for syntactic chunking.

type. For example, the chunk tag of *B-VP* is used for the first word of a verb phrase. We also give an example of sentence labeled with syntactic chunking tags in IOB2 format from the source domain: "The/B-NP $/I-NP 1.4/I-NP billion/I-NP robot/I-NP spacecraft/I-NP faces/B-VP a/B-NP six-year/I-NP journey/I-NP to/B-VP explore/I-VP Jupiter/B-NP and/O its/B-NP 16/I-NP known/I-NP moons/I-NP ./O "

We build the same processing procedure to a vocabulary as the POS tagging experiments. On all processed sentences except 226 "general fiction" sentences from the target domain which we will use as test data, we applied the LBLA and LBL models separately to perform distributed representation learning. We used the same hyperparameter setting and initializations as we did in the POS tagging experiments. After augmenting each sentence with the learned representation features, standard supervised syntactic chunking can be performed.

We used the same SEARN algorithm for supervised syntactic chunking. We used 39, 832 labeled newswire sentences from the source domain and 200 "general fiction" sentences from the target domain as training data, and used 226 "general fiction"

Table 2.3: Empirical results in terms of error rates on semi-supervised domain adaptation for named entity recognition.

Adaptation Models	Error Rates
SRCONLY	2.87%
TGTONLY	2.75%
ALL	2.36%
SCL	2.21%
LBL	1.97%
EA	2.06%
EA++	1.93%
LBLA	**1.53**%

sentences from the target domain as test data. In addition to the traditional features, we also extracted POS tag features as inputs. Under this setting, the test results in term of error rate are reported in Table 2.2, which show the proposed LBLA based cross domain syntactic chunking outperforms all the other methods. We then conducted experiments to investigate the performance of each method by varying the number of labeled training sentences from the target domain between 50 and 200. The test results in term of accuracy are plotted in Figure 2.2. The proposed method demonstrated consistent advantages over all the other methods.

2.4.3 Semi-Supervised Domain Adaptation for Named Entity Recognition

For named entity recognition task, we used the same data as the syntactic chunking experiments. The labeled data are tagged with name entity chunking tags in IOB2 format. The task is to label each word with one of the name entity tags, which represent the position of the word in the name entity chunk and the type of the name entity. For example, *I-LOC* is used for the remaining words of a phrase that represents a location and *B-PER* is used for the first word of a phrase that represents a person. Words located outside of name entity chunks receive the tag *O*, representing miscellaneous names. We also used the same procedure of distributed representation learning as we

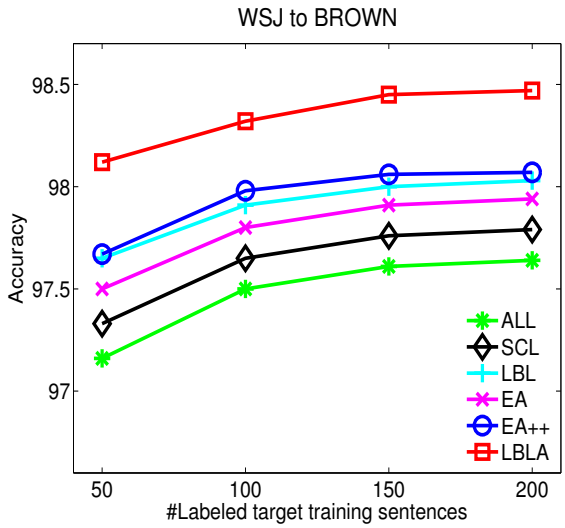

FIGURE 2.3: Empirical results with respect to the number of labeled training data in the target domain on semi-supervised domain adaptation for name entity recognition.

employed in syntactic chunking experiments to produce augmentation features for supervised named entity recognition. Here is an example of tagged sentence from WSJ domain: "Bell/B-ORG ,/O based/O in/O Los/B-LOC Angeles/I-LOC ,/O makes/O and/O distributes/O electronic/O ,/O computer/O and/O building/O products/O ./O "

For supervised name entity recognition task, again we used 39, 832 labeled newswire sentences from the source domain and 200 labeled "general fiction" sentences from the target domain as training data, and used 226 "general fiction" sentences from the target domain as test data. We use the same SEARN algorithm to perform named entity recognition. In addition to previous feature set, we also extracted syntactic chunking tags as phrase chunking features. The experimental results in term of error rate are reported in Table 2.3. We can see that the proposed LBLA representation learning based method outperforms all other methods. We then investigated how the number of labeled training sentences from the target domain affects the performance of each comparison method on named entity recognition. The results in term of accuracy are plotted in Figure 2.3, which show the proposed method clearly outperforms

all other methods across the range of experiments.

2.5 Conclusion

In this chapter, we proposed to tackle semi-supervised domain adaptation problems for sequence labeling tasks in NLP by developing a log-bilinear language adaptation (LBLA) model. The LBLA model learns distributed representations of the words across domains which encode both generalizable features and domain-specific features. The distributed representation vector for each word can be then used as augmenting features for supervised natural language processing systems. We empirically evaluated the proposed LBLA based domain adaptation method on WSJ and MEDLINE domains for POS tagging systems, and on WSJ and Brown corpora for syntactic chunking and named entity recognition tasks. The results show that LBLA method consistently outperforms all other comparison methods for cross domain sequence labeling tasks.

CHAPTER 3

UNSUPERVISED DOMAIN ADAPTATION FOR SEQUENCE LABELING

In the chapter, we consider unsupervised domain adaptation where the target domain only has unlabeled training data. By exploiting additional labeled and unlabeled training data in the label-rich source domain, we propose a task-informative representation learning approach based on dynamic dependency networks. We first introduce the background and related works, and then present the main approach and experiments. This chapter is based on the work published in the International Conference on Computational Linguistics (COLING) [89].

3.1 Introduction

Unsupervised domain adaptation assumes that the target domain has only unlabeled data, which is much more challenging compared to semi-supervised adaptation scenarios as no supervised information is provided in the target domain. Though difficult, unsupervised domain adaptation is more practically useful in real-world applications as it requires no in-domain labeled training data and has more powerful capability of reducing target annotation effort. A variety of works have demonstrated the effectiveness and success of unsupervised domain adaptation for sequence labeling tasks

such as part-of-speech tagging [14, 41, 13], syntactic chunking [40] or named entity recognition [79].

The basic issue for unsupervised domain adaptation also comes from the feature representation divergence. Though the traditional lexical features are widely used in supervised training for in-domain applications and are demonstrated with good empirical results, it is not suitable for cross-domain adaptation scenarios. Different genres of sentences of the two domains may lead to very different vocabularies, rendering the traditional lexical-feature based representations not generalizable across different genre domains. The complete non-availablility of labeled training data in the target domain makes the adaptation learning much more difficult. Recently, some unsupervised representation learning techniques are proposed to induce generalizable latent features across domains by exploiting a large amount of unlabeled data from two domains [14, 40, 41]. However, the latent features produced by the unsupervised representation learning techniques provide no task-specific discriminative information over the labels of NLP tasks.

To tackle this issue, in this chapter, we propose a semi-supervised Dynamic Dependency Network (DDN) model to induce task-specific discriminative latent features across domains. Besides exploiting large amount of unlabeled data from two domains, the DDN model will also leverage the already-existing labeled data from the source domain. It combines the advantages of semi-supervised learning methods from [14, 28] with the sequence models from [40, 41], while maintaining desirable properties like computational tractability and the modeling flexibility to incorporate lots of features. This model is more appealing than unsupervised representation learning techniques when a target NLP task is known. Moreover, though we perform representation learning in a semi-supervised manner, we only exploit the existing labeled data in the source domain. Thus our model can be applied to arbitrary new domains without any extra annotation effort. The proposed model is empirically evaluated for out-of-domain POS tagging systems on articles from WSJ and MEDLINE as well as

for out-of-domain syntactic chunking systems from WSJ and Brown corpora, and is shown to outperform unsupervised representation learning techniques.

3.2 Related Work

Literature has witnessed the success of unsupervised domain adaptation for a variety of sequence labeling tasks. They exploit a large amount of unlabeled data in the two domains to induce latent generalizable features as augmenting features which are then added to train supervised sequence labeling systems. [14] proposed a structural correspondence learning (SCL) method for domain adaptation. They first manually select a set of lexical features, which occur frequently in the unlabeled data across the two domains, as the pivot features. Then they model the correlation between pivot features and non-pivot features to automatically learn generalizable features. [40] proposed to train a Hidden Markov Model (HMM) [64] on the large amount of unlabeled data in the two domains and then use the model to infer hidden states on the sentences as latent features. The induced hard clusters (hidden states) are then used to augment the original sentences to train a sequence labeling system. Further, [41] proposed to train multiple HMMs with different initializations on same data to learn multi-dimensional feature representations as augmenting features in training a cross-domain sequence labeling system. [79] empirically investigated a set of features such as Brown clusters, Collobert and Weston embeddings [26], and HLBL embeddings [56] as augmenting features for out-of-domain NER tasks. Though unsupervised representation learning achieves great empirical performance for out-of-domain NLP tasks, it underutilizes the source data, since it completely neglects the existing task-specific labels when performing representation learning. The DDN model we propose in this work can suitably address this problem. DDNs can naturally exploit labels for decoding hidden states when performing semi-supervised representation learning.

3.3 Representation Learning for Unsupervised Domain Adaptation

We consider the unsupervised domain adaptation problem from a source domain $\mathcal{S}$ to a target domain $\mathcal{T}$. In the source domain, we have l_s labeled sentences $\{(X_i^s, Y_i^s)\}_{i=1}^{l_s}$ and u_s unlabeled sentences $\{(X_i^s)\}_{i=1}^{u_s}$ for $n_s = \ell_s + u_s$, where X_i^s is the ith input sentence, i.e., a sequence of words, $w_1, w_2, \ldots, w_{T_i}$, and Y_i^s is its corresponding label sequence, e.g., the sequence of POS tags. However, in the target domain, we only have u_t unlabeled sentences $\{(X_i^t)\}_{i=1}^{u_t}$. We aim to combine those training data to develop a robust sequence labeling system for prediction on the target domain.

In this chapter, we propose a semi-supervised Dynamic Dependency Network (DDN) model to induce task-specific discriminative latent features across domains. In the following, we will introduce the dynamic dependency network and describe how to use it for semi-supervised representation learning.

3.3.1 Dynamic Dependency Network

The representation learning approach is based on dynamic dependency networks. A dynamic dependency network is a dynamic extension of dependency networks [38] to model data with sequential observations and labels. Dependency networks are cyclic directed graphical models. Similar to the directed acyclic Bayesian networks [95], dependency networks allow simple local parameters estimations given fully observed data. But by dropping acyclicity constraints, dependency networks are more flexible on modeling interdependencies between variables than acyclic Bayesian networks. Following the same principle of Dynamic Bayesian Networks (DBNs) [58], we extend dependency networks into sequential models to form Dynamic Dependency Networks. Although with directed cycles a DDN model will lose the ability to handling time series data that requires time forward directed arcs (not vice versa), it has increased capacity on modeling word or label interdependencies within local contexts of sentences, comparing to DBNs. Figure 3.1 demonstrates an example of the DDN models we will use for semi-supervised representation learning.

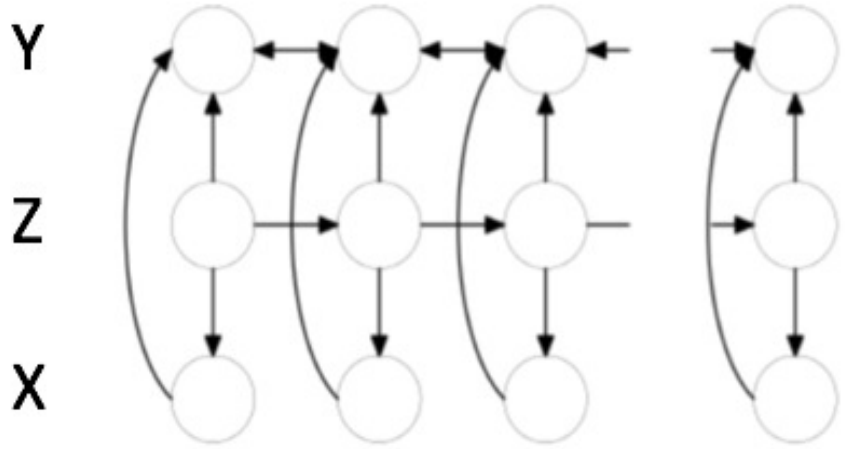

FIGURE 3.1: A Dynamic Dependency Network (DDN). X denotes the observation sequence, Z denotes the hidden state sequence, and Y denotes the label sequence.

In this DDN model (Figure 3.1), the variables are partitioned into three interconnected sequences $X = \{X_1, \ldots, X_T\}$, $Z = \{Z_1, \ldots, Z_T\}$ and $Y = \{Y_1, \ldots, Y_T\}$, representing observations, hidden states and labels respectively. Similar to the HMM model used in [40], the state sequence is hidden in our model and the state Z_t at location t takes values from the predefined set of state values; the observation sequence X is produced from the observed sequence; given Z_t, we assume X_t is independent of $X_{t'}$ for $t \neq t'$. But in addition to the two layers, X and Z in HMMs, our DDN model adds another task specific label layer Y. For example, for the POS tagging task, Y will be the sequence of POS tags. Moreover, we take the bidirectional sequential dependency between labels into consideration by connecting each neighbor pairs of labels using bi-directional arcs. At each location t, X_t, Z_t are both parents of Y_t, since we assume both the sentence observation and the hidden state representation determine the sequence label. By incorporating the label sequence Y into the model, we expect to identify more task discriminative latent sequence representations.

3.3.2 Training and Inference

Although we have an additional bi-directional Y layer in DDNs, the structures over the hidden layer Z, and between Z and the observed sequence X are similar to that in HMMs. Thus inference over the hidden states and parameter learning in DDNs are as tractable as that in HMMs. Assume that we are given a data set of N i.i.d. samples $D = \{(X^i, Y^i)\}, i = 1, 2, \ldots, N$, where X^i is the i'th sentence and Y^i is the corresponding sequence of labels, e.g., POS tags, for X^i. Given the training data D, the log-likelihood is

$$L(\theta) = \sum_{i=1}^{N} \log P(X^i, Y^i|\theta) \tag{3.1}$$

where θ denotes the set of model parameters.

Let $q(Z)$ be any non-zero distribution over hidden variables Z, we can get a lower bound for $L(\theta)$. For notational convenience, we will drop the subscript i in the following formulas.

$$\begin{aligned} \ell(\theta) =& \log \sum_{Z} q(Z) \frac{P(X, Y, Z|\theta)}{q(Z)} && (3.2) \\ \geqslant& \sum_{Z} q(Z) \log \frac{P(X, Y, Z|\theta)}{q(Z)} \\ =& L(\theta) - KL(q(Z) \parallel P(Z|X, Y, \theta)) && (3.3) \end{aligned}$$

We denote the objective in (3.3) as $F(q, \theta)$. We then conduct training by maximizing $F(q, \theta)$ using iterative Expectation-Maximization (EM) algorithm [4, 31]. For the $(k + 1)$ iteration, in the E-step, we update q given fixed θ^k

$$q^{k+1} = \arg\max_{q} F(q, \theta^k) \tag{3.4}$$

which has the following solution

$$q^{k+1}(Z) = P(Z|X, Y, \theta^k) \tag{3.5}$$

In the M-Step, we update θ given fixed q^{k+1}

$$\theta^{k+1} = \arg\max_{\theta} F(q^{k+1}, \theta) \tag{3.6}$$

Similar to HMMs, the parameter estimation in (3.6) requires computation of $P(Z_{t-1}, Z_t|X, Y)$ and $P(Z_t|X, Y)$ for all t in step (3.5). We extend the Baum-Welch algorithm used in HMMs to conduct the required computation under the current model parameters θ. Let $\alpha_t(z) = P(X_1, Y_1, \cdots, X_t, Y_t, Z_t = z|\theta)$ and $\beta_t(z) = P(X_{t+1}, Y_{t+1}, \cdots, X_T, Y_T|Z_t = z, \theta)$. The set of $\{\alpha_t(z)\}$ and $\{\beta_t(z)\}$ can be solved inductively by a forward procedure and a backward procedure respectively, which are analogous to the forward and backward procedures used for HMMs. Then

$$P(Z_t = z|X, Y, \theta) = \frac{\alpha_t(z)\beta_t(z)}{\sum_{\widehat{z}} \alpha_T(\widehat{z})} \tag{3.7}$$

$$\begin{aligned} & P(Z_t = z, Z_{t+1} = z\prime|X, Y, \theta) \qquad (3.8) \\ = \quad & \frac{\alpha_t(z)\beta_{t+1}(z\prime)}{\sum_{\widehat{z}} \alpha_T(\widehat{z})} P(Z_{t+1} = z\prime|Z_t = z) \\ & P(Y_{t+1}|Y_t, Y_{t+2}, X_{t+1}, Z_{t+1} = z\prime) \end{aligned}$$

The major difference from HMMs is that the computation of (3.8) requires the additional local probabilities, $P(Y_t|Y_{t-1}, Y_{t+1}, X_t, Z_t)$, in the bidirectional Y sequence. The typical conditional probability table (CPT) parameters for $P(Y_t|Y_{t-1}, Y_{t+1}, X_t, Z_t)$, requires a storage space in the size of $(L-1) \times L^2 \times V \times S$, where L is the number of discrete label values for Y, V is the number of discrete word features for X, an S is the number of discrete states for Z. To reduce the computational cost and memory size for storing such a large CPT, we exploit a multi-class logistic regression to model this conditional probability and store the model parameters of the logistic regression model instead.

The logistic regression classifier is trained in the M-step with data collected at each location t, over four types of features $Y_{t-1}, Y_{t+1}, X_t, Z_t$. The trained logistic regression

model only requires a model parameter matrix W in the size of $L \times (2L + V + S + 1)$ to calculate the probability $P(Y_t|Y_{t-1}, Y_{t+1}, X_t, Z_t, W)$ for any inputs. Apparently, the space required to store the W matrix is much smaller than the space required for the original conditional probability table. To avoid overfitting, we trained a $L2$-norm regularized logistic regression model using the second order Newton method [54].

Given the model parameters, the hidden state values of sequence Z can be computed using the Viterbi inference algorithm used in HMMs. With the computed marginal probabilities and induced hidden states, the model parameters θ of the DDN can be re-estimated in a similar way as in HMMs in addition to the retraining of the logistic regression classifier.

3.3.3 Unsupervised Domain Adaptation with Task-Informative Features

We have introduced above how to train DDNs with labeled sentences and conduct inference to induce the hidden states. For the unsupervised domain adaptation scenario here, the target domain only has unlabeled training data. Compared to the unlabeled training data in the two domains, the amount of labeled training data in the source domain is much smaller. We then show how to exploit unlabeled training data in the DDN model. Note in the DDN model we introduced, dropping the label layer Y does not affect either the structure or the parameter of the other two layers, but simplify a DDN model into a HMM. Thus we can use DDNs as HMMs on unlabeled sentences by sharing common model parameters across labeled and unlabeled sentences. With this task-informative representation learning, we expect to inference latent features that are not only generalizable in different domains, but also more informative or discriminative about the target task labels.

Our overall system follows a similar architecture of [40]. First we train a DDN model over both the labeled sentences in the source domain and the unlabeled sentences in both domains, as we described above. Then we use the trained DDN model to produce latent features (i.e, hidden state values Z) for the training and test sen-

tences using Viterbi inference algorithm. Finally we train a classification model, e.g., CRFs, over the training sentences for the target task, e.g. POS tagging, using the latent features as augmented inputs, and then perform classification on the test sentences. We expect the task-informative representation learning to help improve out-of-domain prediction performance.

3.4 Experiments

In this section, we conduct extensive empirical studies to evaluate the proposed task-informative representation learning approach on both cross-domain part-of-speech tagging and syntactic chunking.

3.4.1 Unsupervised Domain Adaptation for POS Tagging

In this section, we report our empirical study on how the induced task-informative features can improve unsupervised domain adaptation on the part-of-speech tagging problem. We used the same datasets as [14, 40, 41]. The source domain is articles from Wall Street Journal (WSJ), with 39,832 manually tagged sentences from sections 02-21 and 100,000 unlabeled sentences from a 1988 subset. The target domain is biomedical articles from MEDLINE, with 561 labeled sentences, which are manually annotated as part of the Penn BioIE project, and about 100,000 unlabeled sentences. The task is to assign words with one of the POS tags from the Penn Treebank POS tags [49] and two more tags from MEDLINE dataset. Among the tags, two tags cannot be seen in the newswire articles, *HYPH* (For hyphens) and *AFX* (For common post-modifiers for biomedical entities such as genes). These two tags were introduced because of the importance of hyphenated entities in biomedical text, which are about 1.8% of the words in the 561 labeled sentences.

We build a vocabulary with those about 140,000 sentences from WSJ and about 100,561 sentences from MEDLINE. In order to reduce the vocabulary size, we further adopt the following preprocessing steps as [40, 41]: we map lower frequency (0-2)

words to a single unique identifier in our vocabulary and sole-digit words into a single unique identifier. With these preprocessed sentences, we apply representation learning models (DDNs or HMMs) to derive hidden states as additional features for our supervised POS taggers.

We adopt HMMs to perform unsupervised representation learning on 13, 982 unlabeled newswire sentences and 100, 000 unlabeled biomedical sentences following [40]. Then we decode the hidden states for 4,000 newswire sentences as well as 561 biomedical sentences as additional features for supervised POS tagging. In the unsupervised representation training, one hyperparameter, the number of hidden states, has to be set. A large number of hidden states would make the model more capable to derive latent features, however, it also needs more memory storage and high computation cost. We used 80 states in our experiments, following [40].

We use the proposed DDN model for semi-supervised representation learning on 39, 832 labeled and 100, 000 unlabeled newswire sentences as well as 100, 000 unlabeled biomedical sentences. The labels we used in semi-supervised representation learning are the same labels we will use later to train POS taggers. Thus semi-supervised representation learning does not require additional annotation effort, but make use of existing source data, comparing to unsupervised representation learning. In our semi-supervised representation learning, we need to choose two hyperparameters, the number of hidden states and the L2 regularization parameter. We set the former as 80, same as in unsupervised representation learning, and the latter as 0.5.

For supervised POS tagging, the training data is 39,832 labeled newswire sentences and the test data is 561 biomedical sentences. The 561 biomedical sentences contain 14,554 tokens, of which 23% are OOV tokens. We test our semi-supervised representation learning with supervised Conditional Random Field (CRF) POS taggers as they exhibit high-accuracy for out-of-domain words [40]. We adopt a fast-training CRF package developed by [62]. For supervised POS tagger, we present the CRF feature set in Table 3.1. Specifically, we extract unigram features. We also add or-

Table 3.1: CRF feature set used in our supervised CRF POS taggers. T_i variables stand for labels to be predicted, W_i represent word tokens. Z_i stands for hidden state values decoded from HMM or DDN models.

Feature Type	Feature Description
Transition	$T_i = t$ $T_i = t$ and $T_{i-1} = t'$
Word	$W_i = w$ and $T_i = t$
Orthography	For every s ∈ {-ing, -ogy, -ed, -s, -ly, -ion, -tion, -ity}, suffix(W_i)=s and $T_i = t$ W_i is capitalized and $T_i = t$ W_i has a digit and $T_i = t$
HMM features	$T_i = t$ and $Z_i = z$
DDN features	$T_i = t$ and $Z_i = z$

thographical features such as suffix (-ing, -ogy, -ed, -ly, -s, -ion, -tion, -ity), as well as capitalization. Orthographical features contribute to improve tagging accuracy for out-of-vocabulary words as is demonstrated by [46]. In addition, we add the latent states as state features for each word from the learned representations. For example, a sentence like "He is the CEO ." contains 5 words: 4 regular words and a "period". We learn one state feature for each of them.

Our experimental results in terms of per-token accuracy with different representation learning methods are presented in Table 3.2. For all test results reported in this paper, the "*All Words*" results are average accuracies over all words in the test data, the "*OOV Words*" results are average accuracies over OOV words that appear less than 3 times in the training data. We report the empirical results for the following approaches including our proposed semi-supervised representation learning method. (1) *Baseline*- a baseline trained with CRF without representation learning; (2) *ASO*- an Alternating Structural Optimization (ASO) technique developed by [1], *SELF-CRF*- a comparison method using self-training paradigm. We first train a CRF without representation learning on the training data and apply it to the test data, then retrain it on the training data plus the test data with predicted labels; (3) *PLAIN-SEM*- a

Table 3.2: Test results on the unsupervised domain adaptation for part-of-speech tagging.

Approaches	All Words	OOV Words
Baseline	88.3%	67.3%
ASO	88.4%	70.9%
SELF-CRF	88.5%	70.4%
PLAIN-SEM	88.5%	69.8%
SCL	88.9%	72.0%
SEM-CRF	90.0%	71.9%
HMM	90.5%	75.2%
DDN	**91.3**%	**76.1**%

representation learning technique by using contrastive estimation [67]. We used the modified version proposed by [41]; (4) *SCL*- a Structural Correspondence Learning (SCL) technique developed by [14]; (5) *SEM-CRF*- a representation learning method proposed by [41]; (6) *HMM*- a representation learning by using Hidden Markov Models developed by [40]; (7) *DDN*- the proposed semi-supervised representation learning method. From Table 3.2, we can see DDN-based semi-supervised representation learning consistently outperforms HMM-based unsupervised representation learning as well as other comparison methods for out-of-domain POS tagging. Those results justify that *DDN* can produce more effective task-specific features by incorporating existing labels from the source domain.

3.4.2 Unsupervised Domain Adaptation for Syntactic Chunking

In this section, we empirically study how unsupervised domain adaptation for syntactic chunking could be benefited from the induced representations. We used the datasets from the Penn Treebank as the source domain, which consists of sections 02-21 of the Wall Street Journal (WSJ) portion. We perform our tests on the Brown corpus [45]. The test data contains 3 sections (ck01-ck03) of propbanked Brown corpus data, which consists of 426 sentences containing 7,159 tokens. Besides the labeled training and test data, we also incorporate unlabeled data from both domains.

We add about 100,000 unlabeled news sentences for the source domain and about 57,000 unlabeled sentences for the target domain. While the source domain contains newswire text, the test sentences are drawn from the domain of "general fiction" and contain entirely different styles of English.

For all data from both domains, we represent the chunking labels in IOB2 format. IOB2 format is a standard format for various sequence tasks like syntactic chunking; IOB2 is widely used in previous works including the CoNLL 2000 shared task [77]. In IOB2 format, the chunk tags consist of two parts. The first part represents the position of the token in this chunk and the second part stands for the name of the chunk type. For example, the chunking type of *VP* is used for verb phrase words and the chunking type of *NP* is used for noun phrase words. For words forming a chunk of type k, the first word receives the B-k tag (Begin), and the remaining words receive the tag I-k (Inside). Words outside a chunk receive the tag O. We now give an example of sentence labeled with chunking tags in IOB2 format from the source domain: "The/B-NP $/I-NP 1.4/I-NP billion/I-NP robot/I-NP spacecraft/I-NP faces/B-VP a/B-NP six-year/I-NP journey/I-NP to/B-VP explore/I-VP Jupiter/B-NP and/O its/B-NP 16/I-NP known/I-NP moons/I-NP ./O"

For all sentences from both domains, we build a vocabulary with those about 140,000 sentences from WSJ corpus and about 58,000 sentences from Brown corpus. In order to reduce the vocabulary size, we further adopt the same preprocessing steps as we did in POS tagging experiments, mapping lower frequency (0-2) words to a single unique identifier in our vocabulary and sole-digit words into a single unique identifier. With these preprocessed sentences, we apply representation learning models (DDNs or HMMs) to derive hidden states as additional features for our supervised syntactic chunking systems.

We adopt HMMs to perform unsupervised representation learning on about 14,000 unlabeled newswire sentences and 57,000 unlabeled "general fiction" sentences from Brown corpus. Then we decode the hidden states for 4,000 newswire sentences as

well as 426 "general fiction" sentences as additional features for supervised syntactic chunking. In the unsupervised representation training, one hyperparameter, the number of hidden states, has to be set. We used 80 states in our experiments in consideration of model capability, memory storage as well as computation cost.

We use the proposed DDN model for semi-supervised representation learning on about 40,000 labeled and 100, 000 unlabeled newswire sentences as well as about 57,000 unlabeled "general fiction" sentences. Again, our proposed semi-supervised representation learning does not require additional annotation effort, but make use of existing source data, comparing to unsupervised representation learning. In our semi-supervised representation learning, we need to choose two hyperparameters, the number of hidden states and the L2 regularization parameter. We set the former as 80, same as in unsupervised representation learning, and the latter as 0.5.

For supervised syntactic chunking, the training data is about 40,000 labeled newswire sentences and the test data is 426 "general fiction" sentences. The 426 "general fiction" sentences contain 7,159 tokens. We test our semi-supervised representation learning with supervised Conditional Random Field (CRF) syntactic chunking. We adopt the same fast-training CRF package developed by [62]. For syntactic chunking, besides the same CRF feature set from Table 3.1 as we used in POS tagging experiments, we also extract POS tag features. All features are represented with boolean values.

Our experimental results with different representation learning methods are presented in Table 3.3. We evaluated the performance with macro F1 measure, which is widely used in the syntactic chunking task[40, 21]. We report the empirical results for the following approaches including our proposed semi-supervised representation learning method as well as the state-of-the-art chunker on this datasets: (1) *UPC Chunker*- a chunking system based on Voted Perceptrons [20]. [21] trained such a chunker on WSJ sections 02-21 and tested it on three sections of the Brown corpus (ck01-03). The reported results serve as the current state-of-the-art performance

Table 3.3: Test results on the unsupervised domain adaptation for syntactic chunking.

Methods	F1
Baseline	89.93%
SELF-CRF	90.21%
SCL	90.62%
HMM	91.79%
DDN	**93.05**%
UPC Chunker	91.73%

on this experimental settings; (2) *Baseline-* a baseline trained with CRF without representation learning; (3) *SELF-CRF-* a comparison method based on self-training paradigm; (4) *SCL-* a structural correspondence learning technique developed by [14]; (5) *HMM-* a representation learning by using Hidden Markov Models [40]; (6) *DDN-* the proposed semi-supervised representation learning method. From Table 3.3, we can see DDN-based semi-supervised representation learning consistently outperforms HMM-based unsupervised representation learning for out-of-domain syntactic chunking. Those results justify that *DDN* can produce more effective task-specific features by incorporating existing labels from the source domain.

3.5 Conclusion

In this chapter, we addressed unsupervised domain adaptation for sequence labeling as unsupervised domain adaptation is practically useful and applicable as we mainly exploit unlabeled data in the target domain and the labeled training data are solely from the source domain. We proposed a dynamic dependency network model to induce task-specific features to address it. In addition to the large amount of unlabeled data from two domains, it incorporates the task-specific labels from the source training data into representation learning. We then used the induced generalizable state features to augment source training sentences and target test sentences for two cross domain NLP tasks: part-of-speech tagging and syntactic chunking. Our empirical

studies show that the proposed representation learning outperforms comparison representation learning based on HMMs on out-of-domain test data for both POS tagging system and syntactic chunking system. All results suggest the proposed representation learning can better bridge the domain gap between training sentences and test sentences by exploiting task-specific label information in the representation learning process.

CHAPTER 4

CROSS-LINGUAL ADAPTATION FOR SEQUENCE LABELING WITH BILINGUAL WORD PAIRS

In the previous two chapters, we studied domain adaptation scenarios where different domains contain sentences in different genres but in the same language domain. Next, we generalize domain adaptation to a more challenging scenario where different domains contain sentences in different languages – the *cross-lingual adaptation scenario*. In the following two chapters, we focus on cross-lingual adaptation for sequence labeling and present two representation learning approaches based on different auxiliary resources to bridge language gap. In this chapter, we will exploit bilingual word pairs and give an interlingual word representation learning method based on deep neural networks to address cross-lingual sequence labeling problems. This chapter is based on the work published in the Conference on Natural Language Learning (CoNLL) [87].

4.1 Introduction

With the rapid development of linguistic resources and tools in multiple languages, it is very important to develop cross language natural language processing (NLP) systems. Cross-lingual dependency parsing is the task of inferring dependency trees

for observed sentences in the target language where there is few or no labeled training data by using a dependency parser trained with a large amount of sentences with annotated dependency trees in a related label-rich source language [33, 53, 92]. Cross-lingual dependency parsing is popularly studied in the natural language processing area as it can greatly reduce the expensive manual annotation effort in the target language by exploiting the dependency information from a source language [25, 33, 53, 72].

One basic challenging for cross-lingual dependency parsing is the word-level representation divergence across different languages. Since sentences in different languages are expressed using different vocabularies, it would fail if we train a dependency parser on the word-level representation with sentences from a source language and apply it into the target language. A variety of work in the literature proposes to bridge the word-level representation divergence in the two language domains. One intuitive idea is to delexicalize the dependency parser, which replaces the language-specific word-level representations with language-independent features such as universal part-of-speech tags [63]. Since those features are comparable cross different languages, it provides a possible way to transfer dependency parsing information from the source language to the target language and is demonstrated with some good empirical results [53]. However, those features are too simple and limited especially for two very different languages. Some other work proposes to improve the delexicalized system by learning more effective cross-lingual features [33].

In this chapter, we propose to address cross-lingual dependency parsing by learning interlingual word distributed representations via a deep neural network architecture. We first combined all sentences from both languages and built language connections between the two languages via a bilingual dictionary. We then induced a real-valued dense feature vector for each word in a sentence via deep learning as the high-level abstract interlingual representations, which helps to capture semantic similarities across the two languages. After cross language representation learning, we used the induced

word distributed representations as augmenting features to train a dependency parser on the labeled sentences in the source language and applied it into the target language. In order to evaluate the proposed learning technique, we conducted extensive experiments on eight cross language tasks with nine different languages. The experimental results justified the efficacy of our approach and showed that it is more effective to transfer dependency parser across languages than the other comparison methods.

4.2 Related Work

Cross-lingual adaptation learning is of great importance in the NLP area especially in the multilingual community. Various works have been proposed and developed in the literature to address different types of tasks, including cross-lingual sequence labeling problems. Some works use bilingual word pairs to bridge language gap in cross-lingual adaptation as we did. [33] used bilingual word pairs to project feature values from one language to the alternative language to achieve cross-lingual adaptation. They empirically demonstrated its effectiveness on dependency parsing tasks. Though simple, bilingual word pairs are effective in bridge language domain divergence in cross-lingual adaptation.

Deep learning techniques have been widely used in the natural language processing area [26, 27, 39, 69, 74, 79, 94]. [69] applied recursive auto encoders to address sentence-level sentiment classification problem. [27] employed deep learning idea to implement a system, called SENNA, and empirically evaluated it with four tasks, part-of-speech (POS) tagging, chunking, named entity recognition (NER), and semantic role labeling (SRL). [26] proposed a deep learning framework for jointly performing multi-task learning and empirically evaluated it on several natural language processing tasks such as POS tagging, chunking, named entity recognition and semantic role labeling. [39] proposed discriminative training methods of a neural network statistical parser. [74] extended the Incremental Sigmoid Belief Networks [73] to a

generative latent variable model for dependency parsing. [79] empirically used neural networks to induce word representations for sequence labeling tasks such as named entity recognition.

4.3 Representation Learning for Cross-Lingual Adaptation with Bilingual Word Pairs

In this chapter, we aim to tackle generalized domain adaptation for sequence labeling where sentences of the two domains are in different languages. We propose to learn interlingual word distributed representations via a deep neural network architecture. We first combined all sentences from both languages and built language connections between the two languages via a bilingual dictionary. We then induced a real-valued dense feature vector for each word in a sentence via deep learning as the high-level abstract interlingual representations, which helps to capture semantic similarities across the two languages. Afterwards, we use the induced word distributed representations as augmenting features to train a sequence labeling system on the labeled sentences in the source language and applied it into the target language.

4.3.1 Building Cross Language Connections

To induce cross-lingual word representations, we first need to build connections between the source and target languages. In this work, we produce such connections by finding cross-lingual word pairs using the Wikitionary [1], which works as free bilingual dictionaries between language pairs.

Specifically, we first constructed a source language dictionary with all words that appear in the sentences from the source language domain and translate those words to the target language using the Wikitionary. Then we filtered the produced word-to-word translations by dropping the ones where either the same source language word has multiple different translations in the target language side or the same target

[1] http://en.wiktionary.org

language word corresponds to multiple different source language words. We further dropped the word pairs where the translated word in the target language does not appear in the given sentences in the target language domain. After the processing, we have a set of one-to-one bilingual word pairs to build connections between the two language domains. Finally, we built a unified bilingual vocabulary V with words from all sentences of the two language domains. For each one-to-one bilingual word pair we constructed, we assume the two words have equivalent semantic meaning and map them to the same entry in V. Next, we will learn a distributed vector representation for each entry of the bilingual vocabulary V using deep neural networks. By sharing the same representation vectors, the constructed bilingual word pairs will serve as the bridge across languages.

4.3.2 Interlingual Word Representation Learning

Given the constructed bilingual vocabulary V with v entries, we will learn a latent word embedding matrix $R \in \mathbb{R}^{k \times v}$ over the sentences in the two language domains by using a deep neural network model. This embedding matrix will map each word w in the vocabulary V into a real valued representation vector $R(w)$ with length k. For each bilingual pair of words that are mapped into the same entry of V, they will be mapped into the same vector in R as well. Following the strategy of [27], we construct a simple two-class classification problem over the given sentences. We use the sub-sentences with fixed window size c constructed from the given sentences in the two language domains as positive samples and construct the negative samples by replacing the middle word of each positive sub-sentence with a random word from V. We then train a deep neural network for this two-class classification problem, while simultaneously learning the latent embedding matrix R.

The deep neural network architecture is given in Figure 4.1. The bottom layer of the deep architecture is the input layer, which takes a sequence of word tokens, $\mathbf{x} = w_1, w_2, \ldots, w_c$, with a fixed window size c as the input instance. Then we map

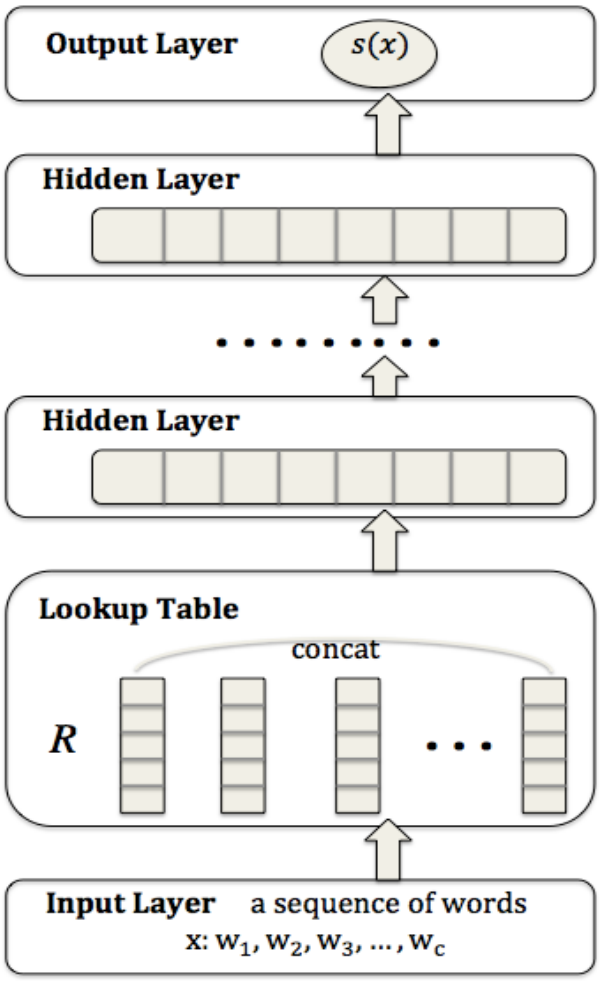

FIGURE 4.1: The architecture of the deep neural network for learning cross-lingual word distributed representations. Each word w_i from the training sample $\mathbf{x}$ is mapped to an interlingual representation vector $R(w_i)$ through the embedding matrix R.

each word w_i in this sequence to an embedding vector $R(w_i)$ by treating the bilingual embedding matrix R as a look-up table. The embedding vector of the sequence of words $\mathbf{x}$ will be concatenated into a long vector $R(\mathbf{x}) \in \mathbb{R}^{ck}$ such that

$$R(x) = [R(w_1); R(w_2); \ldots; R(w_c)]. \tag{4.1}$$

$R(\mathbf{x})$ will then be used as input for the hidden layer above it. The deep neural network has multiple hidden layers. The first hidden layer applies a nonlinear hyperbolic tangent activation function over the linear transformation of its input vector $R(\mathbf{x})$,

such that

$$H_1(\mathbf{x}) = \tanh\left(W_1 \times R(\mathbf{x}) + \mathbf{b}_1\right) \tag{4.2}$$

where $W_1 \in \mathbb{R}^{h_1 \times ck}$ is the model weight parameter matrix, $\mathbf{b}_1 \in \mathbb{R}^{h_1}$ is the bias parameter vector, $H_1(\mathbf{x}) \in \mathbb{R}_1^h$ is the output vector, and h_1 is the number of hidden units in the first hidden layer. Similarly, each of the other hidden layers takes the previous layer's output as its input and performs a nonlinear transformation to produce an output vector. For example, for the i-th hidden layer, we used $H_{i-1}(\mathbf{x})$ as its input and $H_i(\mathbf{x})$ as its output such that

$$H_i(\mathbf{x}) = \tanh\left(W_i \times H_{i-1}(\mathbf{x}) + \mathbf{b}_i\right) \tag{4.3}$$

where $W_i \in \mathbb{R}^{h_i \times h_{i-1}}$ is the weight parameter matrix and $\mathbf{b}_i$ is the bias parameter vector for the i-th hidden layer; h_i denotes the number of hidden units of the i-th hidden layer.

Given t hidden layers, the output representation of the last layer will then be used to generate a final score value for the prediction task, such that

$$s(\mathbf{x}) = \theta \times H_t(\mathbf{x}) + u \tag{4.4}$$

where $\theta \in \mathbb{R}^{h_t}$ is the weight parameter vector and u is the bias parameter for the output layer.

In summary, the model parameters of the deep neural network architecture includes the look-up table R, the parameters $\{W_i, \mathbf{b}_i\}_{i=1}^t$ for the hidden layers, and the output layer parameter (θ, u).

4.3.3 The Training Procedure

The model parameters of the deep network architecture are learned by training a two-class classification model over the constructed positive and negative samples. Let $D = \{\mathbf{x}_i, \hat{\mathbf{x}}_i\}_{i=1}^N$ denote the constructed training set, where $\mathbf{x}_i$ is a positive sample and $\hat{\mathbf{x}}_i$ is a negative sample constructed by replacing the middle word of $\mathbf{x}_i$ with a random word from V. It is desirable for the model to produce an output score

$s(\mathbf{x}_i)$ that is much larger than the score $s(\hat{\mathbf{x}}_i)$ for each pair of the training instances. Thus, we perform training to maximize the separation margins between the pairs of scores over positive and negative samples under a hinge loss; that is we minimize the following training loss

$$J(D) = \frac{1}{N}\sum_{i=1}^{N} \max(0, 1 - s(\mathbf{x}_i) + s(\hat{\mathbf{x}}_i)) \tag{4.5}$$

We perform a random initialization over the look-up table and weight model parameters, and set all the bias model parameters to be zeros. Then we use a stochastic gradient descent [16] algorithm to perform optimization.

4.3.4 Cross Language Sequence Labeling

The training of deep network model above will produce a word embedding matrix R for all words in the two language domains. Moreover, by having each translated bilingual pair of words sharing the same representation vector in R in the training process, the embedding matrix R is expected to capture consistent and comparable semantic meanings across languages, and provide a language-independent and distributed representation for each word in the bilingual dictionary V.

Given R, for each sentences $\mathbf{x} = w_1, w_2, \ldots, w_n$ from the two language domains, we retrieved the representation vector $R(w_i)$ for each word w_i. Moreover, we further delexicalized the sentence by replacing the sequence of language-specific words with a sequence of universal POS tags [63]. Finally, we train a delexicalized sequence labeling system on the labeled sentences in the source language based on the universal POS tag features and the learned distributed features, and apply it to perform sequence tagging on the sentences in the target language domain.

4.4 Experiments

We empirically evaluated the proposed cross-lingual word representation learning for cross-lingual dependency parsing. In this section, we first describe the experimental

setup and then report empirical results.

4.4.1 Experimental Setup

We used the dataset from the CoNLL shared task [18, 61] for cross-lingual dependency parsing. We conducted experiments with the following nine languages: English (EN), Danish (DA), German (DE), Greek (EL), Spanish (ES), Italian (IT), Dutch (NL), Portuguese (PT) and Swedish (SV). For each language, there is a separate training and test set. We used English, which usually has more labeled resources, as the source language, while treating the others as target languages. We thus constructed eight cross-lingual dependency parsing tasks (EN2DA, EN2DE, EN2EL, EN2ES, EN2IT, EN2NL, EN2PT, EN2SV), one for each of the eight target languages. For example, the task *EN2DA* means that we used Danish (DA) as the target language while using *English (EN)* as the source language. For each cross language dependency parsing tasks, we first performed representation learning and then conducted dependency parsing training and test.

In this dataset, each sentence is labeled with gold standard POS tags. To produce delexicalized cross-lingual dependency parsers, we mapped these language-specific POS tags into twelve universal POS tags [63]: ADJ (adjectives), ADP (prepositions or postpositions), ADV (adverbs), CONJ (conjunctions), DET (determiners), NOUN (nouns), NUM (numerals), PRON (pronouns), PRT (particles), PUNC (punctuation marks), VERB (verbs) and X (for others).

4.4.2 Cross-Lingual Dependency Parsing with Representation Learning

To perform distributed cross-lingual representation learning using the proposed deep network architecture, we first constructed the two-class training set from all the sentences (training and test sentences) fo the two language domains. This requires the creation of sub-sentences with fixed window size c from the given sentences. We used window size $c = 5$ in the experiments. For example, for a given sentence "I

Table 4.1: The feature templates used for the cross language dependency parsing. The *dir* denotes the direction of the dependency relationship, which has two values $\{left, right\}$, *dist* denotes the distance between the head word and the dependent word, which has five values {1, 2, 3-5, 6-10, 11+}.

Feature Template	**Description**
$UPOS(w_h)$	the head word's universal POS tag
$UPOS(w_d)$	the dependent word's universal POS tag
$UPOS(w_h, w_d)$	the universal POS tag pair of the head and dependent word
$R(w_h)$	the head word's distributed representations
$R(w_d)$	the dependent word's distributed representations
$dir\&UPOS$	dependency direction based conjunction features
$dist\&UPOS$	dependency distance based conjunction features
$dir\&dist\&UPOS$	dependency direction and distance based conjunction features

visited New York .", we can produce a number of sub-sentences, "<PAD> <S> I visited New", "<S> I visited New York", "I visited New York .", "visited New York . </S>" and "New York . </S> <PAD>", where <PAD> is a special token to fill the length requirement. Negative samples are constructed by simply replace the middle word of each sub-sentence with a random word. With the constructed training data, we then performed training over the deep neural network. We used 3 hidden layers with 100 hidden units in each layer, considering the model capacity and the training effort. The dimension k of the embedding word vectors in R is set as 200.

We used the MSTParser [50, 52] as the basic dependency parsing model. MSTPaser uses spanning tree algorithms to seek for the candidate dependency trees and employs an online large margin training optimization algorithm. MSTParser is widely used in the literature for dependency parsing tasks and is demonstrated with good empirical results in the CoNLL shared tasks on multilingual dependency parsing [18, 61]. For this dependency parsing model, there are a few parameters to be set: the number of maximum iterations for the perceptron training, the number of best-k dependency

tree candidates. We set the number of iterations to be 10 and only considered the best-1 dependency tree candidate.

For the proposed cross-lingual dependency parsing approach, we used the delexicalized universal POS tag based features and the language-independent word features produced from the deep learning as input features for the MSTParser. The set of universal POS tag based feature templates is given in Table 4.1. For each dependency relationship between a head word w_h and a dependent word w_d, a set of features can be produced from the feature templates in Table 4.1, which can be further augmented by $R(w_h)$ and $R(w_d)$. We compared our proposed approach (*Proposed*) with the three other methods. The *Baseline* method uses a delexicalized MSTParser based only on the universal POS tag features. The *Proj* method is developed in [33], which uses a bilingual dictionary to learn cross-lingual representations and used them as augmenting features to train a delexicalized MSTParser. The *X-lingual* method uses unlabeled parallel sentences to learn cross-lingual word clusters and used them as augmenting features to train a delexicalized dependency parser [72]. All parsers except *X-lingual* are trained on the labeled sentences in the source language domain and tested on the test sentences in the target language domain in the given dataset. The performance is measure using the standard unlabeled attachment score (UAS). The *X-lingual* method uses different auxiliary resources (parallel sentences) and we hence directly cited the results reported in the previous work [72] on the same dataset.

4.4.3 Results and Discussions

We reported the empirical comparison results in terms of UAS in Table 4.2. We can see that the *Baseline* method, which is a delexicalized dependency parser trained on the labeled sentences in the source language, performed poorly across all the tasks. The average unlabeled attachment score for this approach across all the eight tasks is very low (about 55.14), which suggests that the twelve universal POS tags are far from enough to produce a good cross-lingual dependency parser. Considering the

Table 4.2: Test performance in terms of UAS on the eight cross-lingual dependency parsing tasks. Δ denotes the improvements for each method over the *Baseline* method.

Tasks	**Baseline**	**Proj**	Δ	**Proposed**	Δ	**X-lingual**
EN2DA	36.53	41.25	4.72	42.56	6.03	38.70
EN2DE	46.24	49.15	2.91	49.54	3.30	50.70
EN2EL	61.53	62.36	0.83	62.96	1.43	63.00
EN2ES	52.05	54.54	2.49	55.72	3.67	62.90
EN2IT	56.37	57.71	1.34	59.05	2.68	68.80
EN2NL	61.96	64.41	2.45	65.13	3.17	54.30
EN2PT	68.68	71.47	2.79	72.38	3.70	71.00
EN2SV	57.79	60.99	3.20	61.88	4.09	56.90
Average	55.14	57.74	2.60	58.90	3.51	58.29

Table 4.3: Statistic differences. For each task, we report the percentage of sentences in the test data from the target language which shares the same sequence of universal POS tags in the source language but with different dependency trees.

Target Language	**Sentence Difference**
Danish	0.31%
Dutch	1.81%
German	1.40%
Greek	1.20%
Italian	2.40%
Portuguese	1.04%
Spanish	0.97%
Swedish	2.31%

small number of universal POS tags, its limited discriminative capacity as input features for dependency parsing is understandable. To further verify this, we calculated the percentage of sentences in the test data which share the same sequence of universal POS tags with a training sentence in the source language but have different dependency parsing structures. The values for the eight tasks are presented in Table 4.3. The non-trivial values reported verified the universal POS tags drawback on

Table 4.4: The number of selected bilingual word pairs for each of the experimented language pairs.

Language Pairs	**# Soruce Words**	**# Target Words**	**# Bilingual Word Pairs**
English vs Danish	26599	17934	1140
English vs Dutch	26599	27829	2976
English vs German	26599	69336	1905
English vs Greek	26599	13318	869
English vs Italian	26599	13523	2347
English vs Portuguese	26599	27782	2408
English vs Spanish	26599	16465	2910
English vs Swedish	26599	19072	1779

lack- ing discriminative capacity.

By *relexicalizing* the delexicalized MSTParser via augmenting the POS tag sequences with learned interlingual features, both the *Proj* method and the proposed method overcome the drawback of using solely universal POS tags and produce significant improvements over the *Baseline* method across all the tasks. Moreover, the proposed approach consistently outperform both of the *Baseline* method and the *Proj* method for all the eight tasks. By exploiting only free bilingual dictionaries, the proposed method achieves similar average performance to the *X-lingual* method which requires additional parallel sentences. All those results demonstrated the efficacy of our word representation learning method for cross-lingual dependency parsing.

4.4.4 Impact of the Number of Bilingual Word Pairs

For the eight language pairs, we have reported the numbers of words in each language domain and the numbers of selected bilingual word pairs in Table 4.4. Next we investigated how the number of word pairs affects the performance of the proposed cross-lingual dependency parsing. With the selected full set of bilingual word pairs in Table 4.4, we random selected m% of them with m $\in$ {50, 75, 100} to conduct

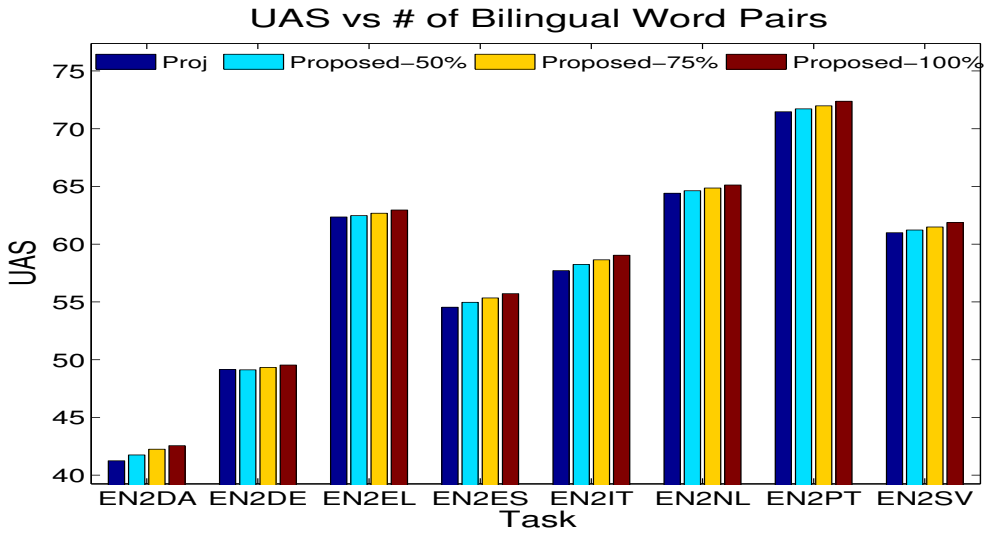

FIGURE 4.2: Unlabeled attachment score on the test sentences in the target language by varying the number of bilingual word pairs.

experiments. Note when m = 50, we only used 435 word pairs for the EN2EL (English vs. Greek) task, which is 1.6% of the number of source words and 3.3% of the number of target words. The results are reported in Figure! 4.2. We can see that by reducing the number of bilingual word pairs, the performance of the proposed cross-lingual dependency parsing method degrades on all tasks. This is reasonable since the word pairs serve as the pivots for learning cross-lingual word embeddings. Nevertheless, by preserving 75% of the selected word pairs, the proposed approach can still outperform the Proj method across all the tasks. Even with only 50% of the word pairs, our method still outperforms the Proj method on most tasks. These results suggest that the proposed cross-lingual word embedding method only requires a reasonable amount of bilingual word pairs to effectively transfer a dependency parser from the source language to the target language.

4.4.5 Impact of Labeled Training Data in Target Language

In the experiments above, all the labeled sentences for dependency parsing training are from the source language. We wonder how much benefit we can get if there are a small number of labeled sentences in the target language as well. To answer this question,

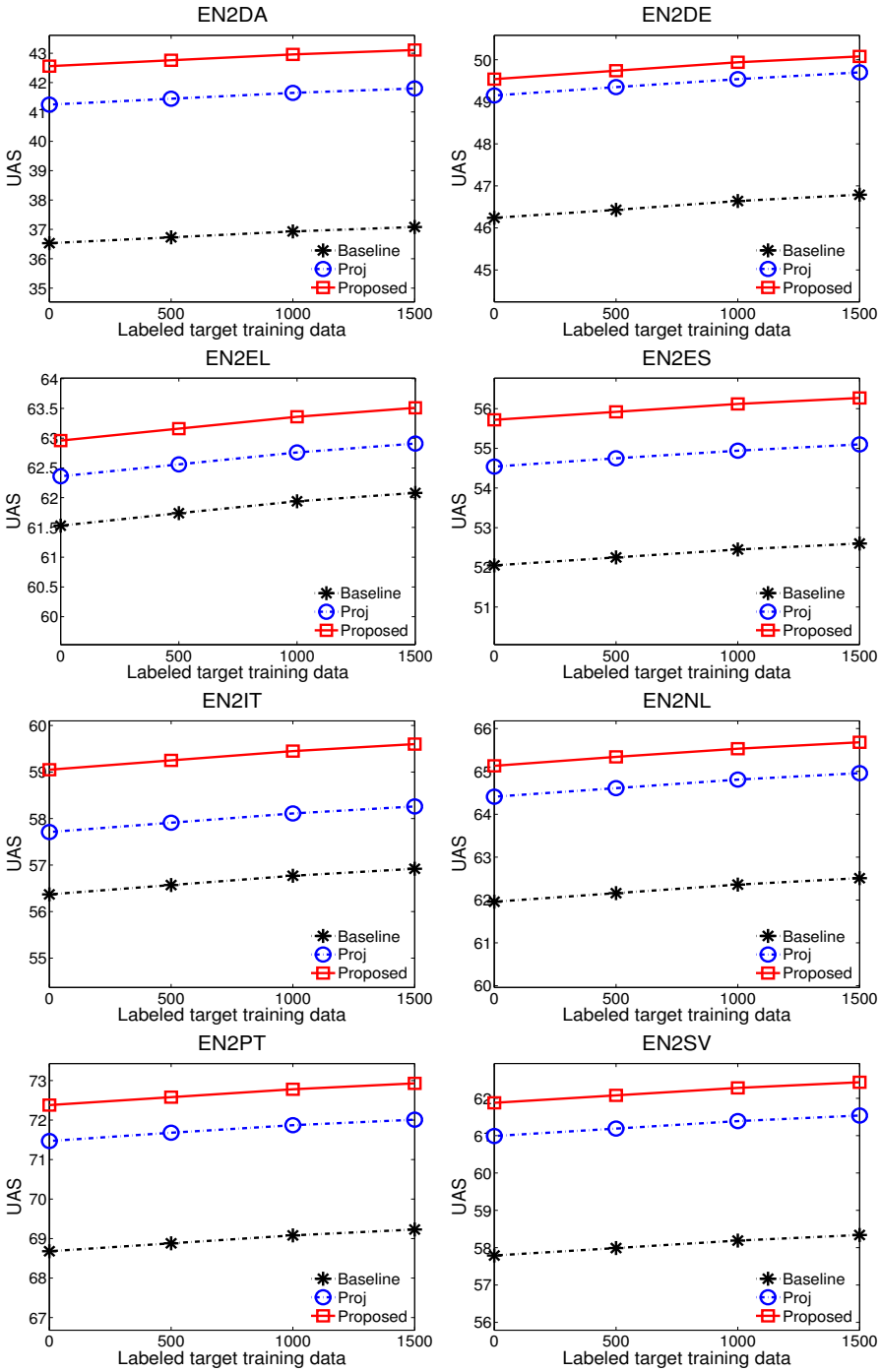

FIGURE 4.3: Unlabeled attachment score (UAS) on the test sentences in the target language by using a dependency parser trained with different number of labeled training sentences in the target language.

we conducted experiments by using a small number (ℓ_t) of labeled sentences in the target language domain together with the labeled sentences in the source language

domain to train cross-lingual dependency parsers. Again the performance of the parsers are evaluated on the test sentences in the target language. We tested a few different ℓ_t values with $\ell_t \in \{500, 1000, 1500\}$. We reported the unlabeled attachment score for all the eight cross-lingual dependency parsing tasks in Figure 4.3. We can see that the Baseline method still performs poorly across the range of different setting for all the eight tasks. The Proj method and the proposed method again consistently out-perform the baseline method across all the tasks, while the proposed method achieves the best results across all the eight tasks.

4.5 Conclusion

In this chapter, we focused on cross-lingual adaptation with representation learning in aid of bilingual word pairs. We proposed to automatically learn language-independent features within a deep neural network architecture. We first constructed a set of bilingual word pairs with Wikitionary, which serve as the pivots in the bilingual vocabulary for building connections across languages. We then conducted distributed word representation learning by training a constructed auxiliary classifier using deep neural networks, which induced a real-valued embedding vector for each word of the bilingual vocabulary to capture consistent semantic similarities for words in the two language domains. The distributed word embedding vectors were then used as augmenting features for cross-lingual adaptation. In particular, we focused on cross-lingual dependency parsing in the empirical studies. We empirically evaluated the proposed method on eight cross-lingual dependency parsing tasks between eight language pairs. The experimental results demonstrated the effectiveness of the proposed method, comparing to other cross-lingual dependency parsing methods.

CHAPTER 5

CROSS-LINGUAL ADAPTATION FOR SEQUENCE LABELING WITH PARALLEL SENTENCES

In this chapter, we continue to work on cross-lingual adaptation but in a different specific setting where the two language domains have additional unlabeled parallel sentences. We then use those additional parallel sentences as auxiliary resources to induce cross-lingual representations to address cross-lingual sequence labeling. In particular, we focus on cross-lingual dependency parsing. Next, we introduce the problem and present the main representation learning approach and provide empirical studies. This chapter is based on the work published in the Conference on Natural Language Learning (CoNLL) [88].

5.1 Introduction

The multilingual community has witnessed an enormous development of cross-lingual applications. Cross-lingual dependency parsing aims to train a dependency parser in the label-rare target language domain by exploiting labeled sentences from a label-rich source language domain and is popularly studied in the literature [33, 51, 71, 70]. The fundamental issue of cross-lingual dependency parsing lies in how to effectively transfer the *annotation* information from the source language domain to the target

language domain. Due to the language divergence over the word-level representations and the sentence structures, simply training a monolingual dependency parser on the labeled source language data without adaptation learning will fail to produce a dependency parser that works in the target language domain. To tackle this problem, a variety of works in the literature have designed better algorithms to exploit the annotated resources in the source languages, including the cross-lingual annotation projection methods [42, 66, 92], the cross-lingual direct transfer with linguistic constraints methods [34, 60, 59], and the cross-lingual representation learning methods [33, 72, 91].

In this chapter, we propose a novel representation learning method to address cross-lingual dependency parsing, which exploits annotation projections on a large amount of unlabeled parallel sentences to induce latent cross-lingual features via matrix completion. It combines the advantages of the cross-lingual annotation projection methods, which project labeled information into the target language domain, and the cross-lingual representation learning methods, which learn latent interlingual features. Specifically, we first train a dependency parser on the labeled source language data and use it to infer labels for the unlabeled source language sentences of the parallel resources. We then project the annotations from the source language to the target language via the word alignments on the parallel sentences. Afterwards, we define a set of interlingual features and construct a word-*feature* matrix by associating each word with these language-independent features. We then use the original labeled source language data and the predicted (or projected) labeled information on the parallel sentences to fill in the observed entries of the word-feature matrix, while matrix completion is performed to fill the remaining missing entries. The completed word-feature matrix provides a set of consistent cross-lingual representation features for the words in both languages. We use these features as augmenting features to train a dependency parsing system on the labeled data in the source language and perform prediction on the test sentences in the target language. To evaluate the proposed

learning method, we conduct experiments on eight cross-lingual dependency parsing tasks with nine different languages. The experimental results demonstrate the superior performance of the proposed cross-lingual transfer learning method, comparing to other approaches.

5.2 Related Work

A variety of cross-lingual adaptation approaches have been developed to address cross-lingual dependency parsing in the literature with parallel sentences. Much work developed in the literature is based on annotation projection [42, 48, 66, 92]. Basically, they exploit parallel sentences and first project the annotations of the source language sentences to the corresponding target language sentences via the word level alignments. Then, they train a dependency parser in the target language by using the target language sentences with projected annotations. The performance of annotation projection-based methods can be affected by the quality of word-level alignments and the specific projection schema. Therefore, [42] proposed to heuristically correct or modify the projected annotations in order to increase the projection performance while [66] used a more robust projection method, quasi-synchronous grammar projection, to address cross-lingual dependency parsing. Moreover, [48] proposed to project the discrete dependency arcs instead of the treebank as the training set. These works however assume that the parallel sentences are already available, or can be obtained by using free machine translation tools. Instead, [92] considered the cost of machine translation and used a bilingual lexicon to obtain a translated treebank with projected annotations from the source language.

A number of works are developed based on representation learning [33, 72, 91]. In general, these methods first automatically learn some language-independent features and then train a dependency parser in this interlingual feature space with labeled data in the source language and apply it on the data in the target language. [72] used unlabeled parallel sentences to induce cross-lingual word clusterings and used

these word clusterings as interlingual features. Both [33] and [72] assumed that the twelve universal part-of-speech (POS) tags [63] are available and used them as the basic interlingual features. Moreover, [91] proposed to automatically map language-specific POS tags to universal POS tags to address cross-lingual dependency parsing, instead of using the manually defined mapping rules.

Besides, some other works are proposed based on multilingual linguistic constraints [34, 35, 60, 59, 84]. Basically, they first construct a set of linguistic constrains and then train a dependency parsing system by incorporating the linguistic constraints via posterior regularization. The constraints are expected to bridge the language differences. [34] automatically learned the constraints by using parallel data while some other works manually constructed them by using the universal dependency rules [60] or the typological features [59].

5.3 Representation Learning for Cross-Lingual Adaptation with Parallel Sentences

In this section, we present a novel representation learning method for cross-lingual dependency parsing using auxiliary parallel sentences, which combines annotation projection and matrix completion-based feature representation learning together to produce effective interlingual features.

We consider the following cross-lingual dependency parsing setting. We have a large amount of labeled sentences in the source language and a set of unlabeled sentences in the target language. In addition, we also have a large set of auxiliary unlabeled parallel sentences across the two languages. We aim to learn interlingual feature representations such that a dependency parser trained in the source language sentences can be applied in the target language domain. The framework for the proposed cross-lingual representation learning system is given in Figure 5.1. The system has two steps: cross-lingual annotation projection and cross-lingual representation learning. We present each of the two steps below.

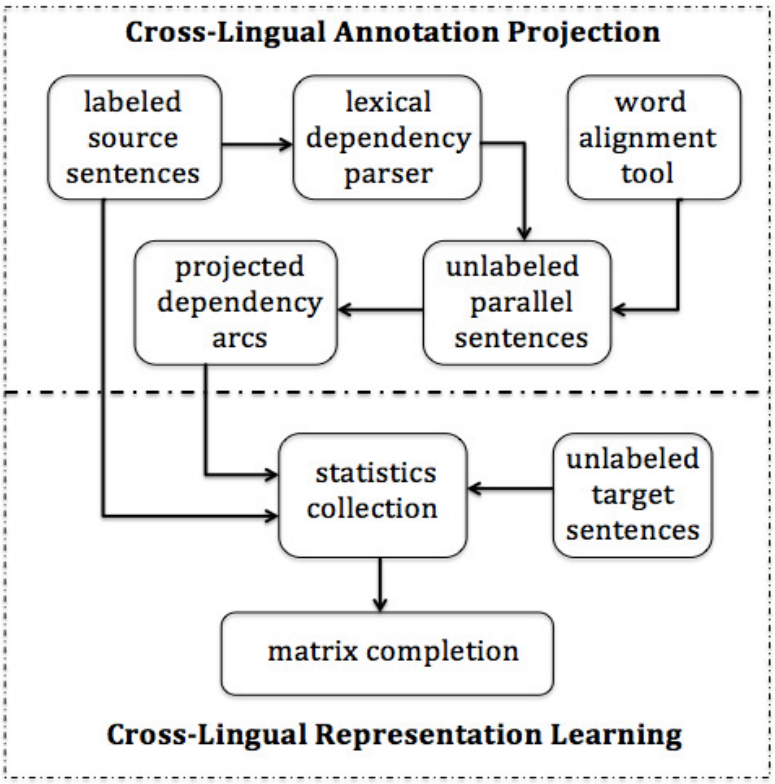

FIGURE 5.1: The architecture of the proposed cross-lingual representation learning framework, which consists of two steps, cross-lingual annotation projection and cross-lingual representation learning.

5.3.1 Cross-Lingual Annotation Projection

In the first step, we employ a large amount of unlabeled parallel sentences to transfer dependency relations from the source language to the target language. Parallel data-based annotation projection has been exploited in the multilingual community for different NLP tasks such as POS tagging or chunking [90]. Here, we exploit it to address cross-lingual dependency parsing. We first train a lexicalized dependency parser with the labeled training data in the source language. Then we use this parser to produce parse trees on the source language sentences of the auxiliary parallel data. Simultaneously, we perform word-level alignments on the unlabeled parallel sentences using existing alignment tools. Finally, we project the predicted dependency relations of the source language sentences to their parallel counterparts in the target language

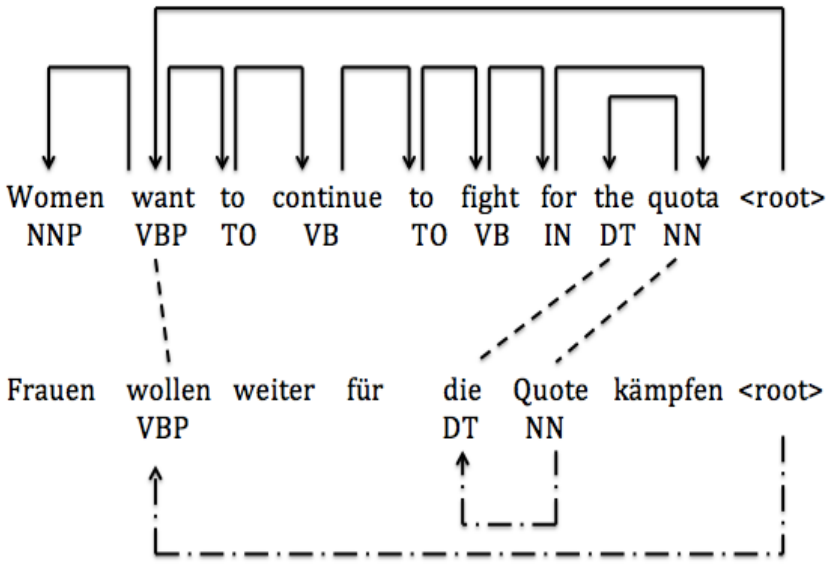

FIGURE 5.2: An example of cross-lingual annotation projection, where a partial word-level alignment is shown to demonstrate two cases of annotation projection.

via the word-level alignments. Instead of projecting the whole dependency trees, which requires more sophisticated algorithms, we simply project each dependency arc on the source sentences to the target language side.

We now use a specific example in Figure 5.2 to illustrate the projection step. This example contains an English sentence and its parallel sentence in German. The English sentence is fully labeled with each dependency relation indicated by a solid directed arc. The dashed lines between the English sentence and the German sentence show the alignments between them. For each dependency arc instance, we consider the following properties: the parent word, the child word, the parent POS, the child POS, the dependency direction, and the dependency distances. The projection of the dependency relations from the source language to the target language is conducted based on the word-level alignment. There are two different scenarios. The first scenario is that the two source language words involved in the dependency relation are aligned to two different words in the corresponding target sentence. For example, the English words “the” and “quota” are aligned to German words “die” and “Quote”

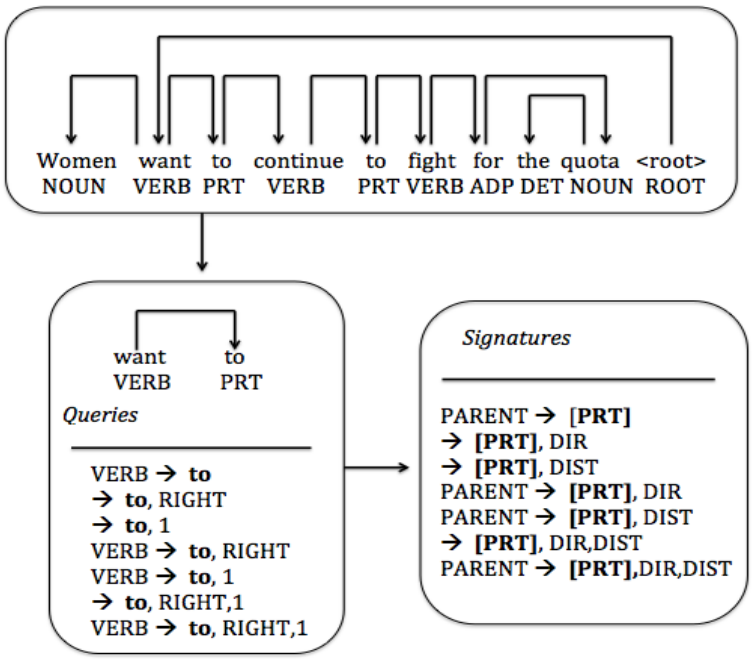

FIGURE 5.3: Example of how to collect queries for each specific dependency relation and how to obtain the abstract signatures (adapted from [33]).

separately. We then copy this dependency relation into the target language side. The second scenario is that a source language word is aligned to a word in the target language sentence and has a dependency relation with the "<root>" word. For example, the English word "want" is aligned to "wollen" and it has a dependency arc with "<root>". We then project the dependency relation from the English side to the German side as well. Moreover, we also directly project the POS tags of the source language words onto the target language words. Since the word order for each aligned word pair in parallel sentences can be different, we recalculate the dependency direction and the dependency distance for the projected dependency arc instance. Note the example in Figure 2 only shows a partial word-level alignment to demonstrate the two cases of the annotation projection. The word alignment tool can align more words than shown in the example.

5.3.2 Cross-Lingual Representation Learning

After cross-lingual annotation projection, we have a set of projected dependency arc instances in the target language. However, the sentences in the target language are not fully labeled. Dependency relation related features are not readily available for all the words in the target language domain. Hence, in this step, we first generate a set of interlingual features and then automatically fill the missing feature values for the target language words with matrix completion based on the projected feature values.

We use the signature method in [33] to construct a set of interlingual *features* for the words in the source and target language domains. The signatures proposed in [33] for dependency parsing are universal across different languages, and have numerical values that are computed in specific dependency relations. Here we illustrate the signature generation process by using an example in Figure 5.3, which is adapted from [33]. Note for each dependency relation between a parent (also known as the head) word and a child (also known as the dependent) word, we can collect a number of queries based on the dependency properties. For example, given the dependency arc between "want" and "to" in the English sentence in Figure 5.3, and assuming we consider the child word "to", we produce queries by considering a non-empty subset of the dependency properties (the parent POS, the dependency direction, the dependency distance), which provides us 7 queries: "VERB→to", "→to, RIGHT", "→to, 1", "VERB →to, RIGHT", "VERB→to, 1", "→to, RIGHT, 1", "VERB→to, RIGHT, 1", where VERB is the parent POS tag, RIGHT is the dependency direction and 1 is the dependency distance. Then we can abstract the specific queries to generate the signatures by replacing the considered word ("to") with its POS tag ("PRT"), and replacing the parent POS tag with "PARENT", the specific dependency distance with "DIST" and the dependency direction with "DIR". This produces the following 7 signatures: "PARENT→[PRT]", "→[PRT], DIR", "→[PRT], DIST",

Table 5.1: The number of induced "features" of each signature for a given word.

Signatures	# Features
[PRT] → DIR	2
[PRT] → DIST	5
[PRT] → CHILD	13
[PRT] → DIR, DIST	10
[PRT] → CHILD, DIR	26
[PRT] → CHILD, DIST	65
[PRT] → CHILD, DIR, DIST	130
→ [PRT], DIR	2
→ [PRT], DIST	5
PARENT → [PRT]	13
→ [PRT], DIR, DIST	10
PARENT → [PRT], DIR	26
PARENT → [PRT], DIST	65
PARENT → [PRT], DIR, DIST	130
Total	502

"PARENT→[PRT], DIST", "PARENT→[PRT], DIST", "→[PRT], DIR, DIST", and "PARENT→[PRT], DIR, DIST", where the brackets indicate the POS tags are for the considered word. Similarly, we can perform the same abstraction process for the parent word "want" and get another 7 signatures (see Table 5.1). Since each signature contains one POS tag and there are 13 different POS types (12 universal POS tags and 1 special type for the "<root>" word), we can get a total of $7 \times 2 \times 13 = 182$ signatures. These signatures are independent of specific languages, though their numerical values should be computed in a specific dependency relation for each considered target word.

A set of interlingual *features* can then be generated from these abstractive signatures by considering different instantiations of their items. For a given target word with an observed POS tag, it has 14 signatures (see Table 5.1). For each signature, we consider all possible instantiations of its other items given the fixed target word.

For example, for the target word "to", its signature "→[PRT], DIR" can be instantiated into 2 features: "→ LEFT" and "→ RIGHT". Similarly, its signature "→[PRT], DIST" can be instantiated into 5 features since DIST has 5 different values ({1, 2, 3–5, 6–10, 11+}), and its signature "[PRT]→CHILD" can be instantiated into 13 features since CHILD denotes the child word's POS tags and can have 13 different values. Hence as shown in Table 5.1, we can get 502 features from the 14 signatures.

The signature-based 502 interlingual features together with the 13 universal POS tag features can be used as *language independent features* for all the words in the vocabulary constructed across the source and target language domains. In particular, we can form a *word-feature* matrix with the constructed vocabulary and the total 515 language independent features. For each word that appeared in the dependency relation arcs, we can use the number of appearances of its interlingual features as the corresponding feature values. However, the sentences in the target language are not fully labeled. Some words in the target language domain may not be observed in the projected dependency arc instances, and we cannot compute their feature values for the 502 interlingual features, though the 13 universal POS tag features are available for all words. Moreover, since we only have a limited number of projected dependency arc instances in the target language, even for some target words that appeared in thee projected arc instances of the parallel data, we may only observe a subset of features among the total 502 interlingual features, with the rest features missing. Hence the constructed *word-feature* matrix is only partially observed, as shown in Figure 5.4. Furthermore, there could also be some noise in the observed feature values as some word features may not have received sufficient observations.

To solve the missing feature problem and simultaneously perform data denoising, we exploit a feature correlation assumption: the 502 constructed interlingual features and the 13 universal POS tags are not mutually independent; they usually contain a lot statistical correlation information. For example, for a word "want" with POS tag "VERB", its feature value for "VERB → **want**, RIGHT" is likely to be very small such as zero, while its feature value for "**want**→ NOUN, LEFT" is likely to be large. Moreover, the existence of any one of the two interlingual features in this example can also indicate the non-existence of the other feature. The existence of feature correla-

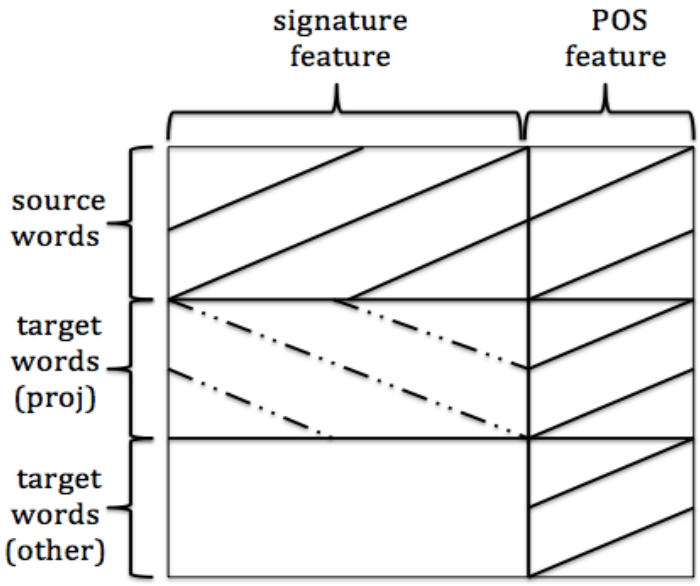

FIGURE 5.4: The word-feature matrix. There are three parts of words: the source language words, target language words from the projected dependency arc instances, and additional target language words. The signature features are the 502 interlingual features and the POS features are the 13 universal POS tags. Solid lines indicate observed entries, dashed lines indicate partially observed entries, while empty indicates missing entries.

tions establishes the low-rank property of the word-feature matrix. We hence propose to fill the missing feature values and reduce the noise in the word-feature matrix by performing matrix completion. Low-rank matrix completion has been successfully used in many applications to fill missing entries of partially observed low-rank matrices and perform matrix denoising [19, 86] by exploiting the feature correlations and underlying low-dimensional representations. Following the same principle, we expect to automatically discover the missing feature values in our word-feature matrix and perform denoising through low-rank matrix completion.

Let $M^0 \in \mathbb{R}^{n \times k}$ denote the partially observed word-feature matrix in Figure 5.4, where n is the number of words and k is the dimensionality of the feature set, which is 515 in this study. Let Ω denote the set of indices for the observed entries. Hence for each observed entry $(i, j) \in \Omega$, M^0_{ij} contains the frequency collected for the j-th feature of the i-th word. We then formulate matrix completion as the following optimization problem to recover a full matrix M from the partially observed matrix

Table 5.2: Feature templates for training a basic delexicalized dependency parser. *upos* stands for the universal POS tag, h stands for the head word, d stands for the dependent word, *dist* stands for the dependency distance, which has five values $\{1, 2, 3-5, 6-10, 11+\}$, and *dir* stands for the dependency direction, which has two values {left, right}.

Basic	**Conj with dist**	**Conj with dir**	**Conj with dist and dir**
upos_h	dist, upos_h	dir, upos_h	dist, dir, upos_h
upos_d	dist, upos_d	dir, upos_d	dist, dir, upos_d
upos_h, upos_d	dist, upos_h, upos_d	dir, upos_h, upos_d	dist, dir, upos_h, upos_d

M^0:

$$\min_{M \geqslant 0} \gamma \|M\|_* + \alpha \|M\|_{1,1} + \sum_{(i,j) \in \Omega} (M_{ij} - M^0_{ij})^2 \tag{5.1}$$

where the trace norm $\|M\|_*$ enforces the low-rank property of the matrix, and $\|M\|_{1,1}$ denotes the entrywise L1 norm. Since many words usually only have observed values for a small subset of the 502 interlingual features due to the simple fact that they are only associated with very few POS tags, a fully observed word-feature matrix is typically sparse and contains many zero entries. Hence we use the L1 norm regularizer to encode the sparsity of the matrix M. The nonnegativity constraint $M \geqslant 0$ encodes the fact that our frequency based feature values in the word-feature matrix are all nonnegative. The minimization problem in Eq (5.1) can be solved using a standard projected gradient descent algorithm [86].

5.3.3 Cross-Lingual Dependency Parsing

After matrix completion, we can get a set of interlingual features for all the words in the word-feature matrix. We then use the interlingual features for each word as augmenting features and train a delexicalized dependency parser on the labeled sentences in the source language. The parser is then applied to perform prediction on the test sentences in the target language, which are also delexicalized and augmented with the interlingual features.

5.4 Experiments

In this section, we empirically evaluate the proposed representation learning method for cross-lingual dependency parsing. We first present the experimental setup and then provide results and discussions.

5.4.1 Experimental Setup

We used the multilingual dependency parsing dataset from the CoNLL-X shared tasks [18, 61] and experimented with nine different languages: *Danish (Da), Dutch (Nl), English (En), German (De), Greek (El), Italian (It), Portuguese (Pt), Spanish (Es)* and *Swedish (Sv)*. For each language, the original dataset contains a training set and a test set. We constructed eight cross-lingual dependency parsing tasks, by using English as the label-rich source language and using each of the other eight languages as the label-poor target language. For example, the task *En2Da* means that we used English sentences as the source language data and Danish sentences as the target language data. For each task, we used the original training set in English as the labeled source language data, and used the original training set in the target language as unlabeled training data and the original test set in the target language as test sentences. Each sentence from the dataset is labeled with gold standard POS tags. We manually mapped these language-specific POS tags to 12 universal POS tags: NOUN (nouns), NUM (numerals), PRON (pronouns), ADJ (adjectives), ADP (prepositions or postpositions), ADV (adverbs), CONJ (conjunctions), DET (determiners), PRT (particles), PUNC (punctuation marks), VERB (verbs) and X (for others).

We used the unlabeled parallel sentences from the *European parliament proceedings parallel corpus* [44], which contains parallel sentences between multiple languages, as auxiliary unlabeled parallel sentences in our experiments. For the representation learning over each cross-lingual dependency parsing task, we used all the parallel sentences for the given language pair from this corpus. The number of parallel sentences for the eight language pairs ranges from $1,235,976$ to $1,997,775$, and the number of tokens involved in these sentences in each language ranges from $31,929,703$ to

$50,602,994$.

5.4.2 Representation Learning

For the proposed representation learning, we first trained a lexicalized dependency parser on the labeled source language data using the MSTParser tool (proj with the first order set) [50] and used it to predict the parsing annotations of the source language sentences in the unlabeled parallel dataset. The sentences of the parallel data only contain sequences of words, without additional POS tag information. We then used an existing POS tagging tool [27] to infer POS tags for them. Next we produced word-level alignments on the unlabeled parallel sentences by using the Berkeley alignment tool [47]. With the word alignments, we then projected the predicted dependency relations from the source language sentences of the parallel data to the target language side, which produces a set of dependency arc instances in the target language. Finally, we constructed the partially observed word-feature matrix from these labeled data and conducted matrix completion to recover the whole matrix. For matrix completion, we used the first task *En2Da* to perform parameter selection based on the test performance. We selected γ from $\{0.1, 1, 10\}$ and selected α from $\{10^3, 10^4, 10^5\}$. The selected values $\gamma = 1$ and $\alpha = 10^{-4}$ were then used for all the experiments.

5.4.3 Experimental Results

We first compared the proposed representation learning with annotation projection method, *RLAP*, to the following methods in our experiments: *Delex, Proj2, X-lingual*. The *Delex* method is a baseline method, which replaces the language-specific word sequence with the universal POS tag sequence and then trains a delexicalized dependency parser. We listed the feature templates used in this baseline delexicalized dependency parser in Table 5.2. The *Proj2* method is from [33]. [33] proposed to use bilingual lexicon to learn cross-lingual features and provided two ways to construct the bilingual lexicon, one is based on Wikitionary and the other is based on unlabeled parallel sentences with observed word-level alignments. Hence, since we studied a scenario where parallel sentences are available, we then use the parallel sentence-based

Table 5.3: Comparison results in terms of unlabeled attachment score (UAS) for the eight cross-lingual dependency parsing tasks (English is used as the source language). The evaluation results are on *all the test sentences*. The *Delex* method uses no auxiliary resource, *Proj2, RLAP*, and *X-lingual* use parallel sentences as auxiliary resources. ∇ denotes the improvements of each method over the baseline *Delex* method. The bottom row contains the average results over the eight tasks.

Tasks	**Delex**	**Proj2**	∇	**RLAP**	∇	**X-lingual**
En2Da	36.5	42.9	6.4	43.6	7.1	38.7
En2De	46.2	49.7	3.5	50.5	4.3	50.7
En2El	61.5	63.5	2.0	64.3	2.8	63.0
En2Es	52.1	56.2	4.1	56.3	4.2	62.9
En2It	56.4	59.2	2.8	60.4	4.0	68.8
En2Nl	62.0	64.9	2.9	66.1	4.1	54.3
En2Pt	68.7	71.9	3.2	72.8	4.1	71.0
En2Sv	57.8	62.9	5.1	63.7	5.9	56.9
Average	55.2	58.9	3.8	59.7	4.6	58.3

version as comparison. The *X-lingual* method uses unlabeled parallel sentences to induce cross-lingual word clusters as augmenting features for delexicalized dependency parser [72]. For *X-lingual*, we cited its results reported in its original paper. For other methods, we used the MSTParser [50] as the underlying dependency parsing tool. To train the MSTParser, we set the number of maximum iterations for the perceptron training as 10 and set the number of best-k dependency tree candidates as 1.

We first evaluated the empirical performance of each comparison method on all the test sentences. The comparison results on the eight cross-lingual dependency parsing tasks in terms of unlabeled attachment score (UAS) are reported in Table 5.3. We can see that the baseline method, *Delex*, performs poorly across the eight tasks. This is not surprising since the sequence of universal POS tags are not discriminative enough for the dependency parsing task. Note even for two sentences with the exact same sequence of POS tags, they may have different dependency trees. By using unlabeled parallel sentences as an auxiliary resource, the two methods, *Proj2* and *RLAP*, consistently outperform the baseline *Delex* method, while *X-lingual* outperforms *Delex* on six tasks. Our proposed approach, *RLAP*, which has the capacity of exploiting the

Table 5.4: Comparison results on the short test sentences with length of 10 or less in terms of unlabeled attachment score (UAS). ∇ denotes the improvements of each method over the baseline *Delex* method.

Tasks	**Delex**	**Proj2**	∇	**RLAP**	∇
En2Da	46.7	54.6	7.9	55.7	9.0
En2De	62.0	63.0	1.0	64.0	2.0
En2El	60.9	61.9	1.0	63.2	2.3
En2Es	55.2	58.3	3.1	59.6	4.4
En2It	55.5	56.9	1.4	58.3	2.8
En2Nl	60.3	62.5	2.2	63.7	3.4
En2Pt	80.2	84.5	4.3	85.7	5.5
En2Sv	73.4	76.0	2.6	76.4	3.0
Average	61.8	64.7	2.9	65.8	4.1

Table 5.5: Previous results on the short test sentences with length of 10 or less in terms of unlabeled attachment score (UAS).

Tasks	**USR**	**PGI**	**PR**	**MLC**
En2Da	51.9	41.6	44.0	-
En2De	-	-	39.6	62.8
En2El	-	-	-	61.4
En2Es	67.2	58.4	62.4	57.3
En2It	-	-	-	56.2
En2Nl	-	45.1	37.9	62.0
En2Pt	71.5	63.0	47.8	83.8
En2Sv	63.3	58.3	42.2	74.9

unlabeled parallel sentences, consistently outperforms the other comparison methods across all the eight tasks. It also outperforms the *X-lingual* method on five tasks. The average UAS over all the eight tasks for the *RLAP* method is 1.4 higher than the *X-lingual* method. All these results demonstrated the effectiveness of the proposed representation learning method for cross-lingual dependency parsing.

We also conducted empirical evaluations on short test sentences (with length of 10 or less). We compared *Delex, Proj2* and *RLAP* with four other methods, *USR,*

PGI, PR and *MLC*. The *USR* method is a cross-lingual direct transfer method which uses universal dependency rules to construct linguistic constraints [60]. The *PGI* method is a phylogenetic grammar induction model [10]. The *PR* method is a posterior regularization approach [35]. The *MLC* method is the multilingual linguistic constraints-based method which uses typological features for cross-lingual dependency parsing [59]. Here we used this method in our setting with only one source domain. Moreover, since we do not have typological features for Danish, we did not conduct experiment on the first task with *MLC*. For the methods of *USR, PGI* and *PR*, we cited their results reported in their original papers. All the cited results are also produced on the short sentences of the CoNLL-X shard task dataset. We cited them as references on measuring the progress of cross-lingual dependency parsing on each given target language.

The comparison results are reported in Table 5.4 and Table 5.5. We can see that the results on the short test sentences are in general better than on the whole test set (in Table 5.3) for the same method across most tasks. This suggests that it is easier to infer the dependency tree for a short sentence than for a long sentence. Nevertheless, *Proj2* consistently outperforms *Delex* and *RLAP* consistently outperforms *Proj2* across all the tasks. Moreover, *RLAP* achieves the highest test scores in seven out of the eight cross-lingual tasks among all the comparison systems. This again demonstrated the efficacy of the proposed approach for cross-lingual dependency parsing.

5.4.4 Impact of Labeled Training Data in Target Language

We have also conducted experiments for the learning scenarios where a small set of labeled training sentences from the target language is available. Specifically, we conducted experiments with a few different numbers of additional labeled training sentences from the target language, {500, 1000, 1500}, using three methods, *RLAP, Delex* and *Proj2*. The comparison results on all the test sentences are reported in Figure 5.5. We can see that the performance of all three methods increases very slow but in a similar trend with more additional labeled training instances from the target language. However, both *Proj2* and *RLAP*outperform *Delex* with large margins across all experiments. Moreover, the proposed method, *RLAP*, produces the best results

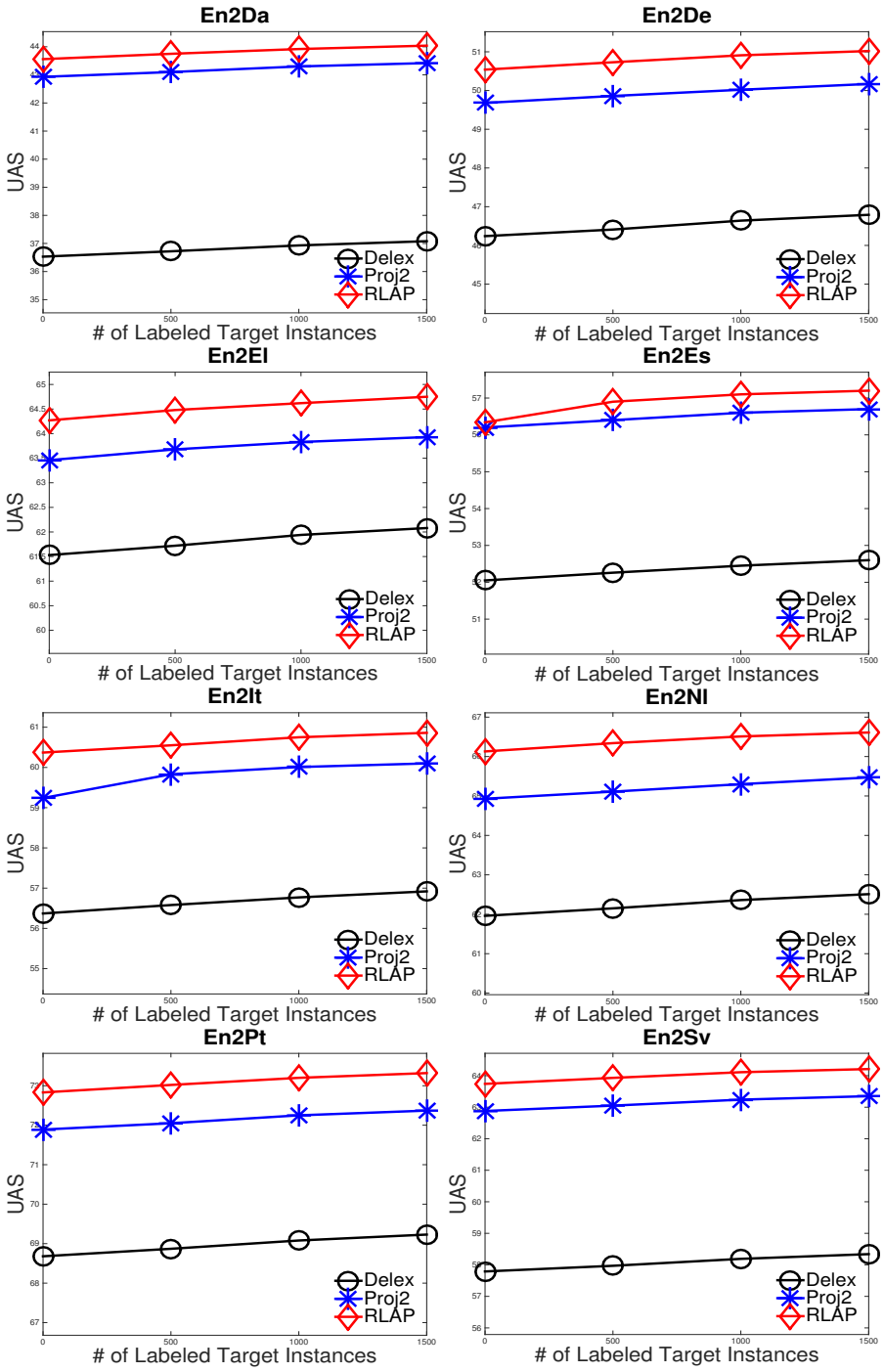

FIGURE 5.5: Unlabeled attachment score on the whole test sentences in the target language by varying the number of labeled training sentences in the target language.

across all the eight tasks. The results again verified the efficacy of the proposed method, demonstrated that filling the missing feature values with matrix completion is indeed useful.

5.5 Conclusion

In this chapter, we proposed a novel representation learning method with annotation projection to address cross-lingual dependency parsing. The proposed approach exploits unlabeled parallel sentences and combines cross-lingual annotation projection and matrix completion-based interlingual feature learning together to automatically induce a set of language-independent numerical features. We used these interlingual features as augmenting features to train a delexicalized dependency parser on the labeled sentences in the source language and tested it in the target language domain. Our experimental results on eight cross-lingual dependency parsing tasks showed the proposed representation learning method outperforms a number of comparison methods.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this dissertation, we focus on generalized domain adaptation where we aim to develop a statistic machine learning model for a target domain where there is a few or no labeled training data by using labeled training data from a different but related source domain. More specific, we studied two specific cases where different domains contain sentences in different genres and give two representation learning approaches, one is for the semi-supervised domain adaptation where the target domain has a small amount of labeled training data, and the other is for unsupervised domain adaptation, where the target domain has no labeled training data. Both representation learning approaches are aimed to induce generalizable features to reduce domain divergence to improve cross-domain learning capability.

We also generalize domain adaptation into cross-lingual learning scenarios where different domains contain sentences in different languages. Based on different auxiliary resources to bridge language gap, we present two representation learning approaches, one uses bilingual word pairs and the other uses parallel sentences with observed word level alignments. Both approaches are empirically evaluated on the task of cross-lingual dependency parsing with promising results.

One interesting direction to explore is to consider *multiple source* domains for

generalized domain adaptation for sequence labeling in NLP [32]. When multiple source domains are available, we can combine them to develop a more robust sequence labeling system for the target domain with properly designed learning algorithms. Meanwhile, different source domains may display different similarities/dissimilarities with the same target domain. Thus, how to select the most useful training data from multiple source domains is also a very interesting direction to explore.

The cross-lingual adaptation is very worth exploring. In this dissertation, we only studied one specific NLP task, cross-lingual dependency parsing. But working on other sequence labeling tasks such as semantic role labeling [93, 75] is also a good research topic. Moreover, we only considered English as the source language. However, English may not be the best source language for some certain target languages. When multiple source languages are available, how to choose the best source language domain is also an interesting direction. The last but not the least, our two cross-lingual adaptation approaches use bilingual word pairs or parallel sentences as auxiliary resource to bridge language gap. However, there are some other resources online in the multilingual community such as code switching data, which are also worth exploring.

BIBLIOGRAPHY

[1] R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research (JMLR)*, 6:1817–1853, 2005.

[2] A. Arnold, R. Nallapati, and W. Cohen. A comparative study of methods for transductive transfer learning. In *Proceedings of the international conference on data mining (ICDM)*, 2007.

[3] N. Ave. A two-stage approach to domain adaptation for statistical classifiers. In *Proceedings of the 16th ACM conference on informaiton and knowledge management*, 2007.

[4] L. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 1970.

[5] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, 2010.

[6] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

[7] Y. Bengio, R. Ducharme, and P. Vincent. A neural probabilistic language model. In *Advances in Neural Information Processing Systems (NIPS)*, 2000.

[8] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *Journal of Machine Learning Research(JMLR)*, 3:1137–1155, 2003.

[9] Y. Bengio and J. Senécal. Quick training of probabilistic neural nets by importance sampling. In *Proceedings of the conference on Artificial Intelligence and Statistics (AISTATS)*, 2003.

[10] T. Berg-Kirkpatrick and D. Klein. Phylogenetic grammar induction. In *Proc. of the Annual Meeting of the Association for Comput. Linguistics (ACL)*, 2010.

[11] C. Biemann and C. Giuliano. Unsupervised part-of-speech tagging supporting supervised methods. In *Proceedings of the International Conference of Recent Advances in Natural Language Processing (RANLP)*, 2007.

[12] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*, 2007.

[13] J. Blitzer, D. Foster, and S. Kakade. Domain adaptation with coupled subspaces. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[14] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2006.

[15] J. Blitzer, K. Weinberger, L. Saul, and F. Pereira. Hierarchical distributed representations for statistical language modeling. In *Advances in Neural Information Processing Systems(NIPS)*, 2004.

[16] L. Bottou. Stochastic gradient learning in neural networks. In *Proceedings of Neuro-Nîmes*, 1991.

[17] T. Brants. Tnt – a statistical part-of-speech tagger. In *Proceedings of the Conference on Applied Natural Language Processing (ANLP)*, 2000.

[18] S. Buchholz and E. Marsi. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, 2006.

[19] R. Cabral, F. Torre, J. Costeira, and A. Bernardino. Matrix completion for multi-label image classification. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[20] X. Carreras and L. Màrquez. Phrase recognition by filtering and ranking with perceptrons. In *Proceedings of the Recent Advances in Natural Language Processing (RANLP)*, 2003.

[21] X. Carreras and L. Màrquez. Introduction to the conll-2005 shared task: semantic role labeling. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, 2005.

[22] W. Che, M. Wang, C. Manning, and T. Liu. Named entity recognition with bilingual constraints. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 2013.

[23] B. Chen, W. Lam, I. Tsang, and T. Wong. Extracting discriminative concepts for domain adaptation in text mining. In *Proceedings of the Interational Conference on Knowledge Discovery and Data Mining (KDD)*, 2009.

[24] M. Chen, K. Weinberger, and J. Blitzer. Co-training for domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[25] W. Chen, J. Kazama, and K. Torisawa. Bitext dependency parsing with bilingual subtree constraints. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[26] R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

[27] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research (JMLR)*, 12:2493–2537, 2011.

[28] H. Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*, 2007.

[29] H. Daumé III, A. Kumar, and A. Saha. Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[30] H. Daumé III, A. Kumar, and A. Saha. Frustratingly easy semi-supervised domain adaptation. In *Proceedings of the Workshop on Domain Adaptation for Natural Language Processing*, 2010.

[31] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[32] M. Dredze, A. Kulesza, and K. Crammer. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79(1-2):123–149, 2010.

[33] G. Durrett, A. Pauls, and D. Klein. Syntactic transfer using a bilingual lexicon. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2012.

[34] K. Ganchev, J. Gillenwater, and B. Taskar. Dependency grammar induction via bitext projection constraints. In *Proceedings of the Joint Conference of the Annual Meeting of the ACL and the International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, 2009.

[35] J. Gillenwater, K. Ganchev, J. Graça, F. Pereira, and B. Taskar. Sparsity in dependency grammar induction. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[36] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[37] M. U. Gutmann and A. Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research (JMLR)*, 13:307–361, 2012.

[38] D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering and data visualization. *Journal of Machine Learning Research (JMLR)*, 1:49–75, 2000.

[39] J. Henderson. Discriminative training of a neural network statistical parser. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2004.

[40] F. Huang and A. Yates. Distributional representations for handling sparsity in supervised sequence labeling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2009.

[41] F. Huang and A. Yates. Exploring representation-learning approaches to domain adaptation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[42] R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11:11–311, 2005.

[43] J. Jiang and C. Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, 2007.

[44] P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proc. of the Machine Translation Summit*, 2005.

[45] H. Kucera and W. Francis. *Computational analysis of present-day American English*. Brown University Press, 1967.

[46] J. Lafferty. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the international conference on Machine Learning (ICML)*, 2001.

[47] P. Liang, B. Taskar, and D. Klein. Alignment by agreement. In *Proc. of the Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (NAACL)*, 2006.

[48] K. Liu, Y. Lü, W. Jiang, and Q. Liu. Bilingually-guided monolingual dependency parsing grammar induction. In *Proceedings of the Conference on Annual Meeting of the Association for Computational Linguistics (ACL)*, 2013.

[49] M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL)*, 1993.

[50] R. McDonald, K. Crammer, and F. Pereira. Online large-margin training of dependency parsers. In *Proceedings of the Annual Meeting on Association for Computational Linguistics (ACL)*, 2005.

[51] R. McDonald, J. Nivre, Y. Quirmbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. Hall, S. Petrov, H. Zhang, O. Tckström, C. Bedini, N. Castelló, and J. Lee. Universal dependency annotation for multilingual parsing. In *Proceedings of the Annual Meeting on Association for Computational Linguistics (ACL)*, 2013.

[52] R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP)*, 2005.

[53] R. McDonald, S. Petrov, and K. Hall. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

[54] T. Minka. A comparison of numerical optimizers for logistic regression. 2003.

[55] A. Mnih and G. Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the international conference on Machine learning (ICML)*, 2007.

[56] A. Mnih and G. Hinton. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[57] A. Mnih and Y. Teh. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

[58] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, 2002.

[59] T. Naseem, R. Barzilay, and A. Globerson. Selective sharing for multilingual dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2012.

[60] T. Naseem, H. Chen, R. Barzilay, and M. Johnson. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2010.

[61] J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. The conll 2007 shared task on dependency parsing. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[62] N. Okazaki. Crfsuite: a fast implementation of conditional random fields (crfs), 2007. http://www.chokkan.org/software/crfsuite/.

[63] S. Petrov, D. Das, and R. McDonald. A universal part-of-speech tagset. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, 2012.

[64] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, 1989.

[65] C. Republic. Semi-supervised training for the averaged perceptron pos tagger. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2009.

[66] D. Smith and J. Eisner. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2009.

[67] N. Smith and J. Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2005.

[68] R. Socher, C. Lin, A. Ng, and C. Manning. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the International Conference on Machine Learning(ICML)*, 2011.

[69] R. Socher, J. Pennington, E. Huang, A. Ng, and C. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

[70] A. Søgaard and J. Wulff. An empirical study of non-lexical extensions to delexicalized transfer. In *Proc. of the Conference on Computational linguistics (COLING)*, 2012.

[71] O. Täckström, R. McDonald, and J. Nivre. Target language adaptation of discriminative transfer parsers. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 2013.

[72] O. Täckström, R. McDonald, and J. Uszkoreit. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 2012.

[73] I. Titov and J. Henderson. Constituent parsing with incremental sigmoid belief networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2007.

[74] I. Titov and J. Henderson. A latent variable model for generative dependency parsing. In *Proceedings of the International Conference on Parsing Technology (IWPT)*, 2010.

[75] I. Titov and A. Klementiev. Crosslingual induction of semantic roles. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2012.

[76] E. Tjong Kim Sang. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*, 2002.

[77] E. Tjong Kim Sang and S. Buchholz. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, 2000.

[78] E. Tjong Kim Sang and F. De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*, 2003.

[79] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[80] V. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995.

[81] M. Wang, W. Che, and C. Manning. Effective bilingual constraints for semi-supervised learning of named entity recognizers. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2013.

[82] M. Wang, W. Che, and C. Manning. Joint word alignment and bilingual named entity recognition using dual decomposition. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2013.

[83] M. Wang and C. Manning. Effect of non-linear deep architecture in sequence labeling. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, 2013.

[84] M. Wang and C. Manning. Cross-lingual pseudo-projected expectation regularization for weakly supervised learning. *Transactions of the Association for Computational Linguistics (TACL)*, 2:55–66, 2014.

[85] M. Xiao and Y. Guo. Domain adaptation for sequence labeling tasks with a probabilistic language adaptation model. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

[86] M. Xiao and Y. Guo. A novel two-step method for cross language representation learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[87] M. Xiao and Y. Guo. Distributed word representation learning for cross-lingual dependency parsing. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*, 2014.

[88] M. Xiao and Y. Guo. Annotation projection-based representation learning for cross-lingual dependency parsing. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*, 2015.

[89] M. Xiao, Y. Guo, and A. Yates. Semi-supervised representation learning for domain adaptation using dynamic dependency networks. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2012.

[90] D. Yarowsky and G. Ngai. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proceedings of the Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies (NAACL)*, 2001.

[91] Y. Zhang, R. Reichart, R. Barzilay, and A. Globerson. Learning to map into a universal pos tagset. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2012.

[92] H. Zhao, Y. Song, C. Kit, and G. Zhou. Cross language dependency parsing using a bilingual lexicon. In *Proceedings of the Joint Conference of the Annual Meeting of the ACL and the International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, 2009.

[93] T. Zhuang and C. Zong. Joint inference for bilingual semantic role labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2010.

[94] W. Zou, R. Socher, D. Cer, and C. Manning. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[95] G. Zoubin. An introduction to hidden markov models and bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(1):9–42, 2001.