LEARNING DEEP REPRESENTATIONS FOR LOW-RESOURCE CROSS-LINGUAL NATURAL LANGUAGE PROCESSING

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Xilun Chen

May 2019

LEARNING DEEP REPRESENTATIONS FOR LOW-RESOURCE CROSS-LINGUAL NATURAL LANGUAGE PROCESSING

Xilun Chen, Ph.D.

Cornell University 2019

Large-scale annotated datasets are an indispensable ingredient of modern Natural Language Processing (NLP) systems. Unfortunately, most labeled data is only available in a handful of languages; for the vast majority of human languages, few or no annotations exist to empower automated NLP technology.

Cross-lingual transfer learning enables the training of NLP models using labeled data from other languages, which has become a viable technique for building NLP systems for a wider spectrum of world languages without the prohibitive need for data annotation. Existing methods for cross-lingual transfer learning, however, require **cross-lingual resources** (e.g. machine translation systems) to transfer models across languages. These methods are hence futile for many low-resource languages without such resources.

This dissertation proposes a deep representation learning approach for low-resource cross-lingual transfer learning, and presents several models that (i) progressively remove the need for cross-lingual supervision, and (ii) go beyond the standard bilingual transfer case into the more realistic multilingual setting. By addressing key challenges in two important sub-problems, namely multilingual lexical representation and model transfer, the proposed models in this dissertation are able to transfer NLP models across multiple languages with no cross-lingual resources.

BIOGRAPHICAL SKETCH

Xilun Chen was born in Jilin City, China. He was intrigued in computers as a kid and started teaching himself to program in Pascal and C in middle school. He then trained and competed in *Olympiad in Informatics* (OI) in high school, and won national awards, on account of which he was admitted to Shanghai Jiao Tong University to study Computer Science. During his undergraduate study, he developed interest in Natural Language Processing, and started conducting research on *deciphering logographic languages* with Professor Hai Zhao.

After obtaining his Bachelor's degree, he joined Cornell University to pursue a Ph.D. degree in Computer Science. During his Ph.D. study, he worked on learning deep representations for low-resource cross-lingual natural language processing to address the scarcity of annotated training data in most human languages. He also did several internships at Google, IBM Research, Facebook, and Microsoft Research, working on various machine learning and natural language processing projects.

To my late Grandfather, Junfeng.

ACKNOWLEDGEMENTS

First and foremost, I am immensely grateful to my advisor, Claire Cardie, for her support, patience, insights, as well as all the guidance and help she provided throughout my Ph.D. study. I am also thankful to my brilliant minor advisors, Jon Kleinberg and John Hopcroft.

Part of this dissertation was based on an internship project I did at Microsoft Research with my mentor Ahmed, and my collaborators Hany and Wei.

Many thanks to Yu, Ben, and Kilian for their collaboration on my first project in cross-lingual NLP, which later became the foundation of my dissertation research.

I would also like to thank Tianze, Arzoo, Xun, Felix, Lillian, Bishan, Lu, Jack, Xanda, Ana, Jon, Moontae, Ozan, Vlad, Kai, Xinya, Liye, Yao, Esin, Ashudeep, Rishi, Ryan, and all other members of the Cornell NLP group for their inspiring discussions and helpful feedback.

I thank Hai Zhao, my Bachelor's thesis advisor, as well as the people at Microsoft Research Asia, Dongdong, Lei, Nan, Mu, Shujie and Ming for introducing me to Natural Language Processing research.

Finally, I thank my family for their love and support, and all my friends for their help and company. Special thanks to Jue, for all the love and joy you brought into my life that made me a much better person.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Cross-Lingual Transfer for Natural Language Processing

A variety of natural language processing (NLP) tasks have benefited from the recent advance of modern deep neural network models. Most of these deep neural models, however, require large-scale annotated datasets for training. While we have witnessed the increasing availability of such annotated datasets for various NLP problems in recent years, most of these annotations only exist in a handful of high-resource languages such as English. Most other world languages, however, are not able to benefit from the deep learning revolution since they do not enjoy such an abundance of labeled data for training modern deep neural networks for a variety of NLP tasks.

As it is prohibitive and inefficient to manually annotate training data for all languages of interest, cross-lingual NLP, or more technically, **cross-lingual transfer learning** (CLTL) comes to the rescue and enables the learning of models for a *target language* using annotated data from other languages (*source languages*) (Yarowsky et al., 2001).

Now we introduce some basic concepts of cross-lingual transfer learning, which will be revisited in more details in Chapter 2.

Bilingual Transfer and Multilingual Transfer Traditionally, research on cross-lingual transfer learning mostly investigates the standard *bilingual transfer* setting, where the training data (for a specific NLP task) comes from a single

source language. In practice, however, it is more often the case that the labeled data is available in a few languages, and it is desirable to be able to utilize all of them when transferring to other languages to further boost the performance on the target language. This scenario is referred to as *multi-source cross-lingual transfer learning* or *multilingual transfer learning*.

Supervision for Cross-Lingual Transfer Learning In this dissertation, we identify three common types of supervision, or resources, for performing cross-lingual transfer, and we name them type 0, I, and II supervision, respectively.

First of all, cross-lingual transfer learning by definition needs some **training data in the source language(s)** for a certain NLP task (type 0 supervision), which is always required.

Another kind of supervision is **task-specific target language annotation** (type I supervision). The availability of such type I supervision usually determines the *objective* of the cross-lingual transfer task. When labeled data is available in the target language, the goal of cross-lingual transfer in this case is usually to leverage additional training data from other language(s) in order to achieve a better performance than using the target language training data alone. On the other hand, there are cases when no target language annotations are available, and cross-lingual transfer aims to build a system for the target language solely relying on the training data from other (source) languages.

The last common type of supervision is **general-purpose cross-lingual resources** (type II supervision), also referred to as *cross-lingual supervision*. Examples of this kind of supervision include generic-domain parallel corpora or a Machine Translation (MT) system. These cross-lingual resources are not task-

specific and can be used for transferring models in multiple NLP tasks, which are especially important in the absence of type I supervision. For instance, a commonly used approach for cross-lingual transfer without type I supervision is to use a Machine Translation system to translate the target language text into the source language, and employ the supervised source language NLP model to make predictions.

Whether a cross-lingual transfer learning system leverages type II supervision often dictates the high-level *methodology*. In the absence of type I supervision, most prior work employs *resource-based* methods (see Chapter 2) that resort to type II supervision to replace the need for target language training data. On the other hand, such type II supervision is also difficult to obtain for many low-resource languages. Therefore, this dissertation concentrates on eliminating the dependence on both type I and type II supervision, by proposing a series of deep neural network models that learn better feature representations for cross-lingual transfer.

Feature Representation Learning for Cross-Lingual Transfer In this dissertation, we focus on learning deep feature representations (see Chapter 2 for more background information) that are better suited for cross-lingual transfer. In particular, two sub-problems need to be addressed when pursuing this deep representation learning methodology. First, a unique challenge faced by cross-lingual transfer learning, compared to other transfer learning problems such as domain adaptation, is the *disparate input space* problem. For instance, different languages such as English and Chinese barely share any words at all, leading to almost disjoint vocabulary sets. As a first step, the model needs to build a **cross-lingual lexical representation** in order to create a common input space. In addition, the

divergence between languages is beyond the word level, and it further calls for an effective **model transfer** method for transferring the NLP models from the source language(s) to the target.

This dissertation consists of several works that progressively eliminate the need for both type I and type II supervision, by addressing key challenges in both problems of model transfer and cross-lingual lexical representation. It extends the frontier of cross-lingual transfer learning and enables the application of modern deep neural NLP models to an increasing number of low-resource languages.

1.2 Contributions

Cross-Lingual Model Transfer without Parallel Training Data. Unlike most traditional work for cross-lingual model transfer that directly rely on type II supervision (parallel data), we propose *Language-Adversarial Training*, a model-based approach that only requires unlabeled monolingual texts from each language during training. This is achieved by learning a language-invariant hidden feature space in which the distribution of the feature vectors of samples from the source and target languages are similar to each other, so that the knowledge learned on the source language training data can be better transferred to the target language. Experiments show that our model achieves state-of-the-art performance on cross-lingual text classification (Chapter 3).

Model Transfer for Multiple Languages and Domains. We further propose a theoretically sound generalization of Language-Adversarial Training for learn-

ing an invariant feature space among multiple *populations* (e.g. languages or domains). In particular, we introduce the Multinomial Adversarial Network (MAN), a general machine learning framework for minimizing the divergence among multiple probability distributions. MAN is applied to multi-domain text classification to validate its effectiveness where it outperforms previous models (Chapter 4), and additional experiments can be found in Chapter 5 when MAN is applied to the multilingual model transfer task.

Multilingual Model Transfer with more than Language-Invariant Features. We take one more step forward and devise a novel multilingual model transfer approach, which, unlike most prior work, can utilize not only the language-invariant features, but also language-specific features from each individual source language for transferring to the target language. This is extremely helpful in the multilingual transfer setting, especially when the target language is similar to a subset of the source languages. Strong performance is attained in multiple multilingual model transfer experiments on various NLP tasks ranging from text classification to sequence tagging (Chapter 5).

Unsupervised Multilingual Lexical Representation. We then study the other fundamental sub-problem in cross-lingual transfer learning, the cross-lingual lexical representation problem, and propose the first method for learning unsupervised multilingual word embeddings (See Chapter 2 for more background information). It is shown that our model outperforms unsupervised and even supervised baseline methods in two separate experiments (Chapter 6).

Zero-Resource Cross-Lingual Transfer Learning. Finally, by combining these efforts in eliminating type II supervision for both the model transfer and lexical representation problems, we are able to perform *zero-resource* cross-lingual transfer where neither type I nor type II supervision is needed. It is referred to as zero-resource in a sense that we are able to transfer the type 0 supervision to a different language using no additional resources. This makes cross-lingual transfer methods more widely applicable to an increasing number of low-source languages, where both annotated data and cross-lingual resources are scarce. Experiments in Chapter 3 first show promising results, and more experiments of zero-resource cross-lingual transfer can be found in Chapter 5.

1.3 Roadmap

The remaining of this dissertation is organized as follows.

We first introduce, in Chapter 2, the relevant background information on cross-lingual transfer learning and deep neural network models for NLP.

We proceed to tackle the cross-lingual model transfer problem in Chapter 3, 4 and 5, aiming to remove the requirement of type I and type II supervision. In Chapter 3, we present language-adversarial training, a pioneering effort towards eliminating the need for parallel data when performing cross-lingual model transfer. We give a theoretically sound generalization of language-adversarial training to support multiple populations (languages or domains) in Chapter 4, and propose a further improved approach for multilingual model transfer in Chapter 5.

Furthermore, we study the cross-lingual lexical representation problem in Chapter 6, and present a model for learning unsupervised multilingual word embeddings.

Finally, we conclude in Chapter 7 by summarizing the contributions of this dissertation and outline possible directions for future work.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, we present background knowledge and review existing research on the key problem that this dissertation tackles, cross-lingual transfer learning, as well as the key methodology that this dissertation takes, deep representation learning for natural language processing.

2.1 Cross-Lingual Transfer Learning

The diversity of human languages poses a critical challenge on Natural Language Processing, especially in the modern deep learning age, as it is prohibitive to obtain sufficient training data for many NLP tasks in most of the world languages. *Cross-lingual NLP*, or more technically *cross-lingual transfer learning* (CLTL), aims at this problem and offers a possibility of learning models for a target language using labeled training data from other (source) languages, and has been applied to a wide variety of NLP tasks such as part-of-speech tagging (Yarowsky et al., 2001), syntactic parsing (Hwa et al., 2005), text classification (Bel et al., 2003), named entity recognition (Täckström et al., 2012), semantic role labeling (Padó and Lapata, 2009), and many more.

In this section, we review the background of cross-lingual transfer learning along three dimensions.

2.1.1 Supervision for Cross-Lingual Transfer Learning

As summarized in Chapter 1, there are three common types of supervision exploited in cross-lingual transfer learning. Namely, *training data in the source language(s)* (**type 0 supervision**), *training data in the target language* (**type I supervision**), and *general-purpose cross-lingual resources* (**type II supervision**). Here, type 0 and type I supervision is task-specific, meaning different training data is needed for different NLP tasks. Type II supervision, however, can be general-purpose in that the same cross-lingual resource can be used to transfer NLP models for multiple tasks, since type II supervision (such as Machine Translation systems) provides general knowledge for connecting the semantics of two or more languages.

Cross-lingual transfer learning by definition requires type 0 supervision. On the other hand, the objective and approaches of cross-lingual transfer learning vary in the literature depending on the availability of type I and type II supervision. In particular, given that the motivation of cross-lingual transfer learning is to address the scarcity of annotated data in the target language, many previous works assume no availability of type I supervision (other than that used for validating and evaluating the models). For simplicity, we refer to this setting as the **unsupervised CLTL** setting, for the CLTL system is unsupervised in the target language. Most prior work relies on type II supervision in this case to bridge the language barrier between the source and target languages, such as parallel corpora (sentence translation pairs) (Yarowsky et al., 2001; Hwa et al., 2005), bilingual dictionaries (word translation pairs) (Prettenhofer and Stein, 2010), or Machine Translation systems (Wan, 2009). On the other hand, when type I supervision is available for training a monolingual supervised model for

a certain NLP task in the target language (the **supervised CLTL** setting), the goal of CLTL is usually to utilize the additional training data available in the source language(s) in order to improve the performance on top of the monolingual target language NLP model. This direction is relatively less investigated as i) it requires annotated training data in the target language and can thus be only applied to a few languages in practice, and ii) the addition of source language training data is usually beneficial only if the target language is close to the source. Many CLTL approaches designed for the more challenging unsupervised setting can be applied to the supervised setting as well, while other methods are specifically targeting the supervised setting with techniques such as multi-task learning, fine-tuning, etc (Swietojanski et al., 2012; Yang et al., 2017).

2.1.2 Resource-based and Model-based Cross-Lingual Transfer Learning

While the existence of type I supervision determines the motivation and objective of CLTL, whether a CLTL method leverages type II supervision, in contrast, often dictates its high-level methodology. Based on the role of type II supervision, CLTL methods can be loosely divided into two categories: *resource-based* CLTL and *model-based* CLTL. Resource-based methods center around a particular cross-lingual resource for connecting the source and target languages, while model-based approaches seek to directly transfer the source language NLP model into the target language.

Resourced-based CLTL One common type of resource-based CLTL is achieved via cross-lingual *projection*. For instance, in annotation projection, the source language NLP model can be projected through a parallel corpus to generate pseudo-supervision for the target language (Yarowsky et al., 2001; Hwa et al., 2005) by making predictions on the source side of the parallel corpus and assume the same labels apply to the target side of the bitext. To reduce the noise generated in this projection process, some works propose to project the expectation over the labels (Wang and Manning, 2014), or to project soft labels (Xu and Yang, 2017). Another family of methods rely on bilingual lexica (word translation pairs) to bridge the language gap (Bel et al., 2003; Mihalcea et al., 2007; Prettenhofer and Stein, 2010). Finally, one line of research focuses on the use of machine translation (MT) systems to achieve CLTL (Wan, 2009; Amini et al., 2009; Zhou et al., 2016).

Model-based CLTL In contrast, model-based CLTL aims to directly transfer the NLP model from the source language to the target. This can be done, for example, through sharing the model parameters, or learning language-agnostic features. A key challenge, though, is the *disparate input space* problem, where the source and the target languages may have considerably dissimilar vocabulary sets, making building a successful model-based transfer system highly challenging. This problem contributes to the fact that model-based CLTL is much less investigated in the past compared to resource-based ones, despite model-based CLTL being more appealing in that it does not require additional type II supervision. Some prior work circumvents the disparate input space problem by relying only on delexicalized features (e.g. part-of-speech tags) so that the source model can be directly applied to the target language (McDonald et al.,

2011). However, such methods are only applicable to certain NLP tasks such as parsing, and generally do not work well for distant language pairs.

Another paradigm that can lead to model-based CLTL is to explicitly addressing the disparate input space problem first by utilizing **cross-lingual lexical representation**, which provides a shared input representation for two or more languages. For instance, one can use *characters* as the shared input representation (Yang et al., 2017) if the source and target languages have the same alphabet. Alternatively, one can build an interlingual lexical (or document) representation using techniques such as latent semantic indexing (Littman et al., 1998), kernel canonical correlation analysis (Vinokourov et al., 2003), or more recently cross-lingual word embeddings (Klementiev et al., 2012; Mikolov et al., 2013b). Using these cross-lingual lexical representations, the remainder of the CLTL task, the **cross-lingual model transfer** part, can be reduced to a *domain adaptation* problem, where abundant literature exists on model-based transfer learning methods (Pan and Yang, 2010) such as parameter sharing, fine-tuning, and learning shared features. One caveat, though, is that most of these traditional approaches for learning cross-lingual lexical representation are resource-based that require type II supervision, and more recent advances in this field are discussed in Section 2.2.3.

2.1.3 Bilingual and Multilingual Transfer Learning

Historically, most research on cross-lingual transfer learning focuses on the simplest and most fundamental setting, the bilingual transfer setting, where the training data comes from a single source language. On the other hand, the mul-

tilingual transfer setting, also known as the multi-source CLTL setting, refers to the scenario where type 0 supervision in multiple source languages is leveraged simultaneously when transferring to the target language. Previous work shows that it can result in significant performance boost over the bilingual setting due to the use of training data in additional source languages (McDonald et al., 2011). While the multi-source CLTL setting can sometimes be viewed as a straightforward generalization of the bilingual transfer setting, and some methods designed for bilingual transfer can be adapted to support multiple source languages, it is still desirable to study dedicated approaches for the multilingual transfer (Hajmohammadi et al., 2014; Guo et al., 2016), because i) not all bilingual transfer methods can be applied to the multilingual setting, and ii) it is often beneficial to devise a more sophisticated mechanism specifically designed for the multilingual transfer that can take into account the relation between all source languages and the target (instead of treating each source-target pair as an independent transfer task or a simple aggregation of all source languages).

Another factor that contributes to the fact that multilingual transfer is less studied is that resource-based methods are less suitable for the multi-source transfer setting, as it is prohibitive to obtain type II supervision (e.g. parallel corpora) between all source-target language pairs. On the other hand, the model-based CLTL approaches proposed in this dissertation, are free of this constraint and can be readily employed for the multilingual transfer task to enjoy the performance boost over the standard bilingual transfer case without the need for additional resources (Chapter 4 and Chapter 5).

2.2 Deep Representation Learning

A wide range of Natural Language Processing (NLP) tasks can be formulated as *prediction* tasks, where the goal is to seek the most likely output $y \in \mathbb{Y}$ given an input $x \in \mathbb{X}$. Here $\mathbb{X}$ and $\mathbb{Y}$ are the sets of all possible inputs and outputs. A few examples of NLP tasks formulated this way include:

- Predicting the sentiment of a sentence, where $\mathbb{X}$ is the set of all possible English sentences and $\mathbb{Y} = \{\text{positive}, \text{neutral}, \text{negative}\}$.
- Predicting the most likely succeeding word given a partial sentence (language modeling), where $\mathbb{X}$ include all possible prefixes of English sentences, and $\mathbb{Y}$ is the set of all English words.
- Translating an English sentence into French, where $\mathbb{X}$ is the set of all English sentences, while $\mathbb{Y}$ is the set of all French sentences.

A model for solving a predictive NLP task can be viewed as a function $\hat{y} = f(x; \theta)$ that predicts an output $\hat{y}$ given the input x, and is parametrized by a set of parameters (weights) θ. A probabilistic model $f(x; \theta) = p(y|x; \theta)$ predicts the probability of the output y given the input x, where the model prediction $\hat{y}$ is determined by finding the most likely output label:

$$\hat{y} = \hat{f}(x; \theta) = \arg\max_{y} p(y|x; \theta) \tag{2.1}$$

There are multiple approaches for deriving a good model $f(x; \theta)$, such as manually writing a set of rules for predicting $\hat{y}$ given an input x. A more appealing method is to use Machine Learning to automatically learn the best set of parameters θ based on *training data*. In the supervised learning setting, we

assume access to a set of (x, y) pairs which can be used to train our model to predict the desired output for a given input. We denote such training data as

$$\mathbb{D} = \{(x_i, y_i) \in \mathbb{X} \times \mathbb{Y}\} \mid_{i=1}^{N} \tag{2.2}$$

The optimal parameters $\hat{\theta}$ are learned by minimizing a *loss function* L between the gold-standard labels y_i on the training data as well as the model predictions.

$$\hat{\theta} = \arg\min_{\theta} \sum_{(x_i, y_i) \in \mathbb{D}} L(y_i, f(x_i; \theta)) \tag{2.3}$$

For instance, in the commonly adopted Maximum Likelihood Estimation (MLE) paradigm, the likelihood of observing the training data under a probabilistic model is maximized in order to obtain $\hat{\theta}$. Please refer to the following book (Murphy, 2012) for more details on Machine Learning.

In a NLP model $f(x; \theta)$, the first step is often to decide how to represent the input x. For instance, how do we represent a word? Furthermore, given the word representations, how do we learn representations for higher-level language structures such as sentences or even documents? Traditionally, deciding on the data representation takes a tremendous amount of human expertise and feature engineering. For example, domain experts look at the NLP task at hand, and make a series of model decisions such as:

- Whether to use morphological features (word lemmas, capitalization, etc.)?
- Whether to use syntactical features (part-of-speech tags, parse trees, etc.)?
- Whether to use additional features from external knowledge bases (e.g. WordNet hypernyms, a lexicon of subjective words, etc.)?
- How much context to retain and how to represent it?

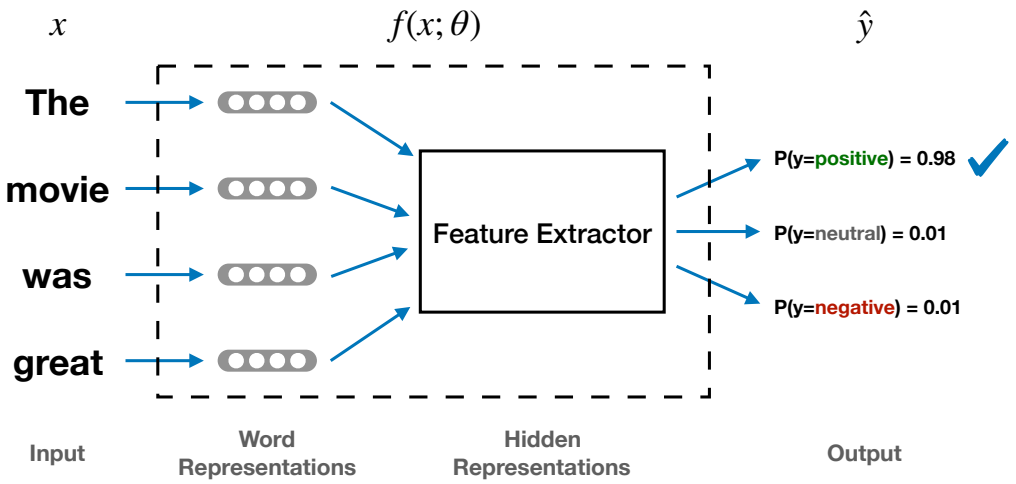

Figure 2.1: Abstract view of a sentiment classifier.

These decisions are difficult, and require expertise and extensive trials and errors. Moreover, models engineered this way do not generalize well to a different task, and one needs to start from scratch when facing a new task.

On the other hand, *(deep) representation learning* strives to automatically learn useful representations for a given task relying solely on the training data, and has enjoyed unmatched success in recent years in various Artificial Intelligence fields in and outside of NLP. To illustrate the high-level architecture of a modern deep representation learning system, Figure 2.1, as an example, shows an overview of a sentiment classifier model, which predicts the sentiment of given piece of text. Here each word in the input x is first represented by the **word representations** (Section 2.2.1) of choice, which is further passed to a *feature extractor* (usually a deep neural network) to learn higher-level **hidden representations** (Section 2.2.2) that retain useful features for solving the sentiment classification task.[1] Please refer to the book by Goodfellow et al. (2016) for a

[1]Note that not all systems learn representations following this two-step paradigm. For example, Zhang et al. (2015) directly model the input document as a sequence of characters, eliminating the step of word representation learning.

(much) more detailed introduction to Deep Learning.

Finally, despite not the focus of this dissertation, it is worth mentioning that the arg max operation in Equation 2.1 might not be straightforward to take depending on the task. For instance, when $\mathbb{Y}$ is a moderately sized set of predefined categories such as the sentiment polarities (Such tasks are referred as *classification tasks*.), one can simply iterate over all possible y values and find the most likely label. On the other hand, $\mathbb{Y}$ can be extremely large or even infinite (as in the translation case), and finding the most likely $\hat{y}$ (or an approximation of it) is a non-trivial challenge and an open research direction called *structured prediction* (Bakir et al., 2007). For simplicity, we will be mostly focusing on the non-structured output case where y is a scalar. Some of the techniques presented in this dissertation can be applied to the structured output case as well, though, as shown in Chapter 5.

The following three subsections will first introduce the two representation learning stages of learning word representations (Section 2.2.1) and hidden (document) representations (Section 2.2.2), and finally discuss a sub-task in word representation learning that is very relevant to cross-lingual NLP — the learning of cross-lingual word embeddings (Section 2.2.3).

2.2.1 Distributed Word Representations

Word representation provides a mathematical representation of the words (or *tokens*) in the input text, and is the first step of many NLP models. Traditionally, a widely-used word representation method is called *one-hot representation*, which represents each token as a one-hot vector, where each dimension in the

vector corresponds to a unique *word type* and the length of the vector equals to the size of the *vocabulary* (all word types in English). Only the dimension corresponding to the token is set to 1, while all other dimensions are set to 0. A major problem with this one-hot representation is that it represents each word type as an atomic and independent categorical feature, disregarding any similarity or relationship between different words. As the vectors for any two word types are orthogonal to each other, it makes the model unable to generalize to unseen word types and the system may hence suffer from training data sparsity.

Distributed Word Representations (also known as *word embeddings*) aim to address this draw back by embedding words into a low-dimensional real-valued vector space[2] instead of a high-dimensional sparse one-hot vector space. The resulted embedding space captures certain syntactic and semantic relationships between various word types, and can provide much better generalization across different words. For instance, in a sentiment classification model where the training data has a positive sample of "`The movie was great.`". Using the one-hot representation, the system could not generalize to "`The movie was awesome.`" assuming that it has never seen it during training, because the words "`great`" and "`awesome`" are treated as two separate and independent words orthogonal to each other. On the other hand, with distributed word representations that can capture similarities between words, the model will hopefully embed the two words close to each other in the vector space, providing improved generalization to unseen word types.

One commonly adopted principle for inducing such distributed word representations is the *distributional hypothesis* (Harris, 1954): words occurring in

[2]Also known as vector space model. There are distributed word representations that adopt alternative forms of representation such as *matrix space* models that represent a word using a matrix instead of a vector (Rudolph and Giesbrecht, 2010).

similar contexts share similar meaning. Therefore, such co-occurrence statistics, which is readily available from the enormous unlabeled text data, can be leveraged to learn word embeddings. There are many successful methods proposed in the literature (Bengio et al., 2003; Turian et al., 2010; Collobert et al., 2011; Mikolov et al., 2013a; Pennington et al., 2014; Bojanowski et al., 2017), and word embeddings have become the de facto standard for word representation in modern deep neural NLP models.

2.2.2 Hidden Representations for Sentences and Documents

As shown in Figure 2.1, for a predictive NLP task such as text classification, it is often necessary to go beyond word representations and learn higher-level representations that take into account the contexts in the input sentence or document[3] (the Feature Extractor box in Figure 2.1). These representations are known as *hidden* representations (or features), because they are often the outputs of some *hidden layers*[4] inside a deep neural network (Goodfellow et al., 2016).

In this section, we present several commonly adopted approaches for learning hidden text representations, using the text classification task as an example. They are referred to as feature extractors in Figure 2.1 since they take the word representations of a various-length input sequence of tokens, and output a fixed-length feature vector that are learned automatically and can be used for predicting the task label y.

[3]For simplicity, we in this dissertation denote the input x as a *sentence* unless otherwise noted, despite that x can sometimes have multiple sentences in practice.

[4]Intermediate network layers that are not the input or output layer.

Bag-of-Words Composition One of the simplest way of combining all the word presentations into a single feature vector is the (orderless) bag-of-words representation. Iyyer et al. (2015) propose Deep Averaging Networks that first take the arithmetic mean of all the word embeddings in the input sequence, and pass it through a number of fully-connected hidden layers to form the final feature vector. This representation does not consider the order of the words at all and inevitably loses many potentially useful information. It nonetheless shows relatively competitive performance (Iyyer et al., 2015) and is extremely simple and efficient.

Convolutional Neural Networks (CNN) One clear limitation of the bag-of-words composition is that language far more than a simple aggregation of words. For instance, in the translation task, there are many idioms and set phrases in each language whose meaning cannot be directly deduced from those of the individual words. Convolutional Neural Networks (Lecun et al., 1998) can extract local features with convolving filters. In the case of NLP, it means that the context around each word is taken into account to produce a local feature vector for that word, and a pooling mechanism (such as max pooling) is then used to find the most salient local features (Kim, 2014).

Recurrent Neural Networks (RNN) Despite CNN being able to take context into account, it still does not consider one of the key factors in human languages: word ordering. Therefore, CNN works well on tasks where finding salient local features generally suffices, such as the sentiment classification task where identifying key words and phrases expressing strong sentiment can help solve many cases. On the other hand, more sophisticated tasks that involve a compre-

hensive understanding of the entire sentence or document such as the machine translation task would require a *sequential composition* method for learning hidden representations that can incorporate word order information. Recurrent Neural Networks is an autoregressive model that composes the hidden representation starting from the beginning of the sentence, and recursively processes each token one at a time to form a hidden feature vector. The processing of each token is called a *timestamp*, and the parameters of a RNN are shared across all timestamps. The RNN output at each timestamp can be viewed as a *contextualized word representation* that takes into account the past utterance (left context) of the token. In order to generate a fixed-length feature vector for the entire input, a pooling mechanism is again needed. For instance, one can use the last hidden output as the representation for the entire input, or one can take the average of all hidden outputs. A more sophisticated pooling technique is the self-attention (Cheng et al., 2016) mechanism that automatically learns a linear combination of all hidden outputs as the final feature vector. There are many architectures for the RNN unit (the neural model that processes each timestamp), and the ones widely used in practice are mostly *gated* ones such as LSTM (Hochreiter and Schmidhuber, 1997) or GRU (Cho et al., 2014). In addition, to overcome the drawback that a RNN only takes into account the context on the left, one can combine a left-to-right RNN with a right-to-left one to form a bidirectional RNN (Schuster and Paliwal, 1997) that considers contexts in both directions.

Other Structural Compositions In RNN, the sentence representation is composed in the natural order (left to right). It is possible, however, to adopt alternative composition strategies when learning the hidden representations. For instance, one idea is to leverage the *syntactic structure* (parse tree) of the input

sentence, and compose the sentence representation following the tree structure using a Recursive Neural Network (Socher et al., 2013; Tai et al., 2015).

Recently there is a new *Transformer* model that composes the hidden representations solely based on self-attention (Vaswani et al., 2017), which has shown strong performance on multiple NLP tasks.

2.2.3 Cross-Lingual Word Embeddings

One subfield of word representation learning that is pertinent to this dissertation is the learning of cross-lingual word representation. With the rise of distributed word representations, it has also become popular to learn distributed cross-lingual lexical representation, namely cross-lingual word embeddings. Cross-lingual word embeddings project words from two or more languages into a single semantic space so that words with similar meanings reside closer to each other regardless of language.

Traditionally, most methods for learning cross-lingual word embeddings were supervised, relying on cross-lingual supervision such as bilingual dictionaries (Klementiev et al., 2012; Mikolov et al., 2013b), or parallel corpora (Zou et al., 2013; Gouws et al., 2015). Some methods attempted to alleviate the dependence on such cross-lingual supervision by only requiring comparable sentences or even documents (Vulić and Moens, 2015).

On the other hand, there are efforts focusing on the multilingual case to learn a shared embedding space across more than two languages (Ammar et al., 2016; Duong et al., 2017), resulting in a set of *multilingual word embeddings* (MWEs).

In comparison, the standard cross-lingual embeddings between two languages are referred to as *bilingual word embeddings* (BWEs). Please refer to this survey (Ruder et al., 2017) for a more detailed coverage on existing research of cross-lingual word embedding induction.

CHAPTER 3

LANGUAGE-ADVERSARIAL TRAINING FOR CROSS-LINGUAL MODEL TRANSFER

In this chapter, we propose the Language-Adversarial Training technique for the cross-lingual model transfer problem, which learns a language-invariant hidden feature space to achieve better cross-lingual generalization using only *unlabeled monolingual texts* from the source language and the target. It is a pioneering effort towards removing type II supervision (cross-lingual resources) from cross-lingual model transfer.

This chapter is based on (Chen et al., 2016; Chen et al., 2018b).

In this chapter, we focus on a simple yet fundamental NLP task, text classification, and present our language-adversarial training approach in the context of cross-lingual text classification (CLTC). In the text classification task, the input is a piece of text (a sentence or document), and the output is chosen from a set of predetermined categories. For instance, one may want to classify a product review into five categories corresponding to its star rating (1-5).

Similar to all cross-lingual transfer learning tasks, the goal of CLTC is to leverage the abundant resources of a *source* language (likely English, denoted as SOURCE) in order to build text classifiers for a low-resource *target* language (TARGET). Our model is able to tackle the more challenging *unsupervised* CLTC setting, where no target language annotations (type I supervision) are available. On the other hand, our method remains superior in the semi-supervised setting

with a small amount of type I supervision available (See Section 3.3.3).

3.1 Related Work

Cross-lingual Text Classification is motivated by the lack of high-quality labeled data in many non-English languages (Bel et al., 2003; Mihalcea et al., 2007; Wan, 2008; Banea et al., 2008, 2010; Prettenhofer and Stein, 2010). Our work is comparable to these in objective but very different in method. Most previous works are resource-based methods that are directly centered around some kind of type II supervision (cross-lingual resources), such as machine translation system, parallel corpora, or bilingual lexica, in order to transfer the knowledge learned from the source language into the target. For instance, some recent efforts make direct use of a parallel corpus either to learn a bilingual document representation (Zhou et al., 2016) or to conduct cross-lingual distillation (Xu and Yang, 2017).

Domain Adaptation tries to learn effective classifiers for which the training and test samples are from different underlying distributions (Blitzer et al., 2007; Pan et al., 2011; Glorot et al., 2011; Chen et al., 2012; Liu et al., 2015). This can be thought of as a generalization of cross-lingual text classification. However, one main difference is that, when applied to text classification tasks, most of these domain adaptation work assumes a common feature space such as a bag-of-words representation, which is not available in the cross-lingual setting. See Section 3.3.2 for experiments on this. In addition, most works in domain adaptation evaluate on adapting product reviews across domains (e.g. books to electronics), where the divergence in distribution is less significant than that between two languages.

Adversarial Networks are a family of neural network models that have two or more components with competing objectives, and have enjoyed much success in computer vision (Goodfellow et al., 2014; Ganin et al., 2016). A series of work in image generation has used architectures similar to ours, by pitting a neural image generator against a discriminator that learns to classify real versus generated images (Goodfellow et al., 2014). More relevant to this work, adversarial architectures have produced the state-of-the-art in unsupervised domain adaptation for image object recognition: Ganin et al. (2016) train with many labeled source images and unlabeled target images, similar to our setup. In addition, other recent work (Arjovsky et al., 2017; Gulrajani et al., 2017) proposes improved methods for training Generative Adversarial Nets. In Chen et al. (2016), we proposed *language-adversarial training*, the first adversarial neural net for cross-lingual NLP, which will be described in this chapter. As of the writing of this dissertation, there are many more recent works that adopt adversarial training for cross-lingual NLP tasks, such as cross-lingual text classification (Xu and Yang, 2017), cross-lingual word embedding induction (Zhang et al., 2017; Lample et al., 2018) and cross-lingual question similarity reranking (Joty et al., 2017).

3.2 Language-Adversarial Networks (LAN)

The central hypothesis of language-adversarial training is that an ideal model for CLTC should learn features that both perform well on text classification for the SOURCE language, and are invariant with respect to the shift in language. Therefore, as shown in Figure 3.1, our proposed model, Language-Adversarial Network (LAN), has a joint *feature extractor* $\mathcal{F}$ which aims to learn features that

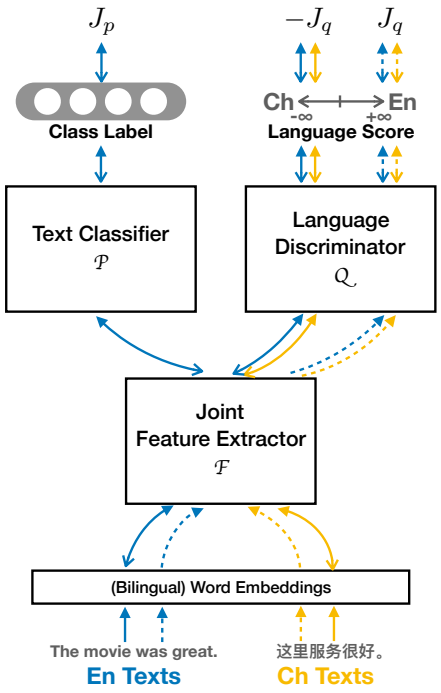

Figure 3.1: LAN with Chinese as the target language. The lines illustrate the training flows and the arrows indicate forward/backward passes. Blue lines show the flow for English samples while Yellow ones are for Chinese. J_p and J_q are the training objectives of $\mathcal{P}$ and $\mathcal{Q}$, respectively (Section 3.2.2). The parameters of $\mathcal{F}$, $\mathcal{P}$ and the embeddings are updated together (solid lines). The parameters of $\mathcal{Q}$ are updated using a separate optimizer (dotted lines) due to its adversarial objective.

aid prediction of the *text classifier* $\mathcal{P}$, and **hamper** the *language discriminator* $\mathcal{Q}$, whose goal is to identify whether an input text is from SOURCE or TARGET. The intuition is that if a well-trained $\mathcal{Q}$ cannot tell the language of a given input using the features extracted by $\mathcal{F}$, those features are effectively language-invariant. $\mathcal{Q}$ is hence *adversarial* since it does its best to identify language from learned features, yet good performance from $\mathcal{Q}$ indicates that LAN is not successful in learning language-invariant features. Upon successful language-adversarial training, $\mathcal{F}$ should have learned features discriminative for text classification, and at the same time providing no information for the adversarial $\mathcal{Q}$ to guess the

language of a given input.

As seen in Figure 3.1, LAN is exposed to both SOURCE and TARGET texts during training. Unlabeled SOURCE (blue lines) and TARGET (yellow lines) data go through the language discriminator, while only the labeled SOURCE data pass through the text classifier[1]. The feature extractor and the text classifier are then used for TARGET texts at test time. In this manner, we can train LAN with labeled SOURCE data and only unlabeled TARGET text. When some labeled TARGET data exist, LAN could naturally be extended to take advantage of that for improved performance (Section 3.3.3).

3.2.1 Network Architecture

As illustrated in Figure 3.1, LAN has two branches. There are three main components in the network, a joint *feature extractor* $\mathcal{F}$ that maps an input sequence x to a fixed-length feature vector in the shared feature space, a *text classifier* $\mathcal{P}$ that predicts the label for x given the feature representation $\mathcal{F}(x)$, and a *language discriminator* $\mathcal{Q}$ that also takes $\mathcal{F}(x)$ but predicts a scalar score indicating whether x is from SOURCE or TARGET.

An input document is modeled as a sequence of words $x = w_1, \ldots, w_n$, where each w is represented by its word embedding v_w (Turian et al., 2010). For improved performance, pre-trained bilingual word embeddings (BWEs, Zou et al., 2013; Gouws et al., 2015) can be employed to induce bilingual distributed word representations so that similar words are closer in the embedded space regardless of language.

[1] "Unlabeled" and "labeled" refer to task labels; all texts are assumed to have the correct language label.

A parallel corpus is often required to train high-quality BWEs, making LAN implicitly dependent on the bilingual corpus. However, compared to the MT systems used in other CLTC methods, training BWEs requires one or two orders of magnitude less parallel data, and some methods only take minutes to train on a consumer CPU (Gouws et al., 2015), while state-of-the-art MT systems need days to weeks for training on multiple GPUs. Moreover, even with randomly initialized embeddings, LAN can still outperform some baseline methods that use pre-trained BWEs (Section 3.3.3). Another possibility is to take advantage of the recent work that trains BWEs with no bilingual supervision (Lample et al., 2018).

We adopt the Deep Averaging Network (DAN) by Iyyer et al. (2015) for the feature extractor $\mathcal{F}$. We choose DAN for its simplicity to illustrate the effectiveness of our language-adversarial training framework, but other architectures can also be used for the feature extractor (Section 3.3.3). For each document, DAN takes the arithmetic mean of the word vectors as input, and passes it through several fully-connected layers until a softmax for classification. In LAN, $\mathcal{F}$ first calculates the average of the word vectors in the input sequence, then passes the average through a feed-forward network with ReLU nonlinearities. The activations of the last layer in $\mathcal{F}$ are considered the extracted features for the input and are then passed on to $\mathcal{P}$ and $\mathcal{Q}$. The text classifier $\mathcal{P}$ and the language discriminator $\mathcal{Q}$ are standard feed-forward networks. $\mathcal{P}$ has a softmax layer on top for text classification and $\mathcal{Q}$ ends with a linear layer of output width 1 to assign a language identification score[2].

[2] $\mathcal{Q}$ simply tries to maximize scores for SOURCE texts and minimize for TARGET, and the scores are not bounded.

3.2.2 Language-Adversarial Training

Before describing the adversarial training approach employed in LAN, we introduce a pre-existing formulation of its adversarial component in which training is done using a Gradient Reversal Layer (Ganin et al., 2016). We refer to this version of LAN as LAN-GRL.

In LAN-GRL, Q is a binary classifier with a sigmoid layer on top so that the language identification score is always between 0 and 1 and is interpreted as the probability of whether an input text x is from SOURCE or TARGET given its hidden features $\mathcal{F}(x)$. For training, Q is connected to $\mathcal{F}$ via a Gradient Reversal Layer (Ganin and Lempitsky, 2015), which preserves the input during the a forward pass but multiplies the gradients by $-\lambda$ during a backward pass. λ is a hyperparameter that balances the effects that $\mathcal{P}$ and Q have on $\mathcal{F}$ respectively. This way, the entire network can be trained in its entirety using standard backpropagation.

Unfortunately, researchers have found that the training of $\mathcal{F}$ and Q in LAN-GRL might not be fully in sync (Ganin and Lempitsky, 2015), and efforts need to be made to coordinate the adversarial training. This is achieved by setting λ to a non-zero value only once out of k batches as in practice we observe that $\mathcal{F}$ trains faster than Q. Here, k is another hyperparameter that coordinates the training of $\mathcal{F}$ and Q. When $\lambda = 0$, the gradients from Q will not be back-propagated to $\mathcal{F}$. This allows Q more iterations to adapt to $\mathcal{F}$ before $\mathcal{F}$ makes another adversarial update.

To illustrate the limitations of LAN-GRL and motivate the formal introduction of our LAN model, consider the distribution of the joint hidden features $\mathcal{F}$

for both SOURCE and TARGET instances:

$$P_{\mathcal{F}}^{src} \triangleq P(\mathcal{F}(x)|x \in \text{SOURCE}) \tag{3.1}$$

$$P_{\mathcal{F}}^{tgt} \triangleq P(\mathcal{F}(x)|x \in \text{TARGET}) \tag{3.2}$$

In order to learn language-invariant features, LAN trains $\mathcal{F}$ to make these two distributions as close as possible for better cross-lingual generalization. In particular, as argued by Arjovsky et al. (2017), previous approaches to training adversarial networks such as LAN-GRL are equivalent to minimizing the Jensen-Shannon divergence between two distributions, in our case $P_{\mathcal{F}}^{src}$ and $P_{\mathcal{F}}^{tgt}$. And because the Jensen-Shannon divergence suffers from discontinuities, providing less useful gradients for training $\mathcal{F}$, Arjovsky et al. (2017) propose instead to minimize the Wasserstein distance and demonstrate its improved stability for hyperparameter selection.

As a result, the LAN training algorithm (see Algorithm 3.1) departs from the earlier LAN-GRL training method. In LAN, we instead minimize the Wasserstein distance W between $P_{\mathcal{F}}^{src}$ and $P_{\mathcal{F}}^{tgt}$ according to the Kantorovich-Rubinstein duality (Villani, 2008):

$$W(P_{\mathcal{F}}^{src}, P_{\mathcal{F}}^{tgt}) = \sup_{\|g\|_L \leq 1} \mathop{\mathbb{E}}_{f(x)\sim P_{\mathcal{F}}^{src}} [g(f(x))] - \mathop{\mathbb{E}}_{f(x')\sim P_{\mathcal{F}}^{tgt}} [g(f(x'))] \tag{3.3}$$

where the supremum (maximum) is taken over the set of all 1-Lipschitz[3] functions g. In order to (approximately) calculate $W(P_{\mathcal{F}}^{src}, P_{\mathcal{F}}^{tgt})$, we use the language discriminator Q as the function g in (3.3), whose objective is then to seek the supremum in (3.3). To make Q a Lipschitz function (up to a constant), the parameters of Q are always clipped to a fixed range $[-c, c]$. Let Q be parameterized

[3] A function g is 1-Lipschitz iff $|g(x) - g(y)| \leq |x - y|$ for all x and y.

Require: labeled SOURCE corpus $\mathbb{X}_{src}$; unlabeled TARGET corpus $\mathbb{X}_{tgt}$; Hyperpamameter $\lambda > 0$, $k \in \mathbb{N}$, $c > 0$.

1: **repeat**
2: ▷ Q iterations
3: **for** $qiter = 1$ to k **do**
4: Sample unlabeled batch $\boldsymbol{x}_{src} \sim \mathbb{X}_{src}$
5: Sample unlabeled batch $\boldsymbol{x}_{tgt} \sim \mathbb{X}_{tgt}$
6: $\boldsymbol{f}_{src} = \mathcal{F}(\boldsymbol{x}_{src})$
7: $\boldsymbol{f}_{tgt} = \mathcal{F}(\boldsymbol{x}_{tgt})$ ▷ feature vectors
8: $loss_q = -Q(\boldsymbol{f}_{src}) + Q(\boldsymbol{f}_{tgt})$ ▷ Eqn (3.4)
9: Update Q parameters to minimize $loss_q$
10: $ClipWeights(Q, -c, c)$
11: ▷ Main iteration
12: Sample labeled batch $(\boldsymbol{x}_{src}, \boldsymbol{y}_{src}) \sim \mathbb{X}_{src}$
13: Sample unlabeled batch $\boldsymbol{x}_{tgt} \sim \mathbb{X}_{tgt}$
14: $\boldsymbol{f}_{src} = \mathcal{F}(\boldsymbol{x}_{src})$
15: $\boldsymbol{f}_{tgt} = \mathcal{F}(\boldsymbol{x}_{tgt})$
16: $loss = L_p(\mathcal{P}(\boldsymbol{f}_{src}); \boldsymbol{y}_{src}) + \lambda(Q(\boldsymbol{f}_{src}) - Q(\boldsymbol{f}_{tgt}))$ ▷ Eqn (3.6)
17: Update $\mathcal{F}$, $\mathcal{P}$ parameters to minimize $loss$
18: **until** convergence

Algorithm 3.1: LAN Training

by θ_q, then the objective J_q of Q becomes:

$$J_q(\theta_f) \equiv \max_{\theta_q} \mathop{\mathbb{E}}_{\mathcal{F}(x)\sim P^{src}_{\mathcal{F}}} [Q(\mathcal{F}(x))] - \mathop{\mathbb{E}}_{\mathcal{F}(x')\sim P^{tgt}_{\mathcal{F}}} [Q(\mathcal{F}(x'))] \tag{3.4}$$

Intuitively, Q tries to output higher scores for SOURCE instances and lower scores for TARGET (as shown in Line 8 of Algorithm 3.1). More formally, J_q is an approximation of the Wasserstein distance between $P^{src}_{\mathcal{F}}$ and $P^{tgt}_{\mathcal{F}}$ in (3.3).

For the text classifier $\mathcal{P}$ parameterized by θ_p, we use the traditional cross-entropy loss, denoted as $L_p(\hat{y}, y)$, where $\hat{y}$ and y are the predicted label distribution and the true label, respectively. L_p is the negative log-likelihood that $\mathcal{P}$ predicts the correct label. We therefore seek the minimum of the following loss function for $\mathcal{P}$:

$$J_p(\theta_f) \equiv \min_{\theta_p} \mathop{\mathbb{E}}_{(x,y)} \Big[L_p(\mathcal{P}(\mathcal{F}(x)), y) \Big] \tag{3.5}$$

Finally, the joint feature extractor $\mathcal{F}$ parameterized by θ_f strives to minimize both the text classifier loss J_p and $W(P^{src}_{\mathcal{F}}, P^{tgt}_{\mathcal{F}}) = J_q$:

$$J_f \equiv \min_{\theta_f} J_p(\theta_f) + \lambda J_q(\theta_f) \tag{3.6}$$

where λ is a hyper-parameter that balances the two branches $\mathcal{P}$ and $\mathcal{Q}$. (See Line 16 in Algorithm 3.1.)

As proved by Arjovsky et al. (2017) and observed in our experiments (Section 3.3.3), minimizing the Wasserstein distance is much more stable w.r.t. hyperparameter selection compared to LAN-GRL, saving the hassle of carefully varying λ during training (Ganin and Lempitsky, 2015). In addition, LAN-GRL needs to laboriously coordinate the alternating training of the two competing components by setting the hyperparameter k, which indicates the number of iterations one component is trained before training the other. The performance can degrade substantially if k is not properly set. In our case, however, delicate tuning of k is no longer necessary since $W(P^{src}_{\mathcal{F}}, P^{tgt}_{\mathcal{F}})$ is approximated by maximizing (3.4); thus, training $\mathcal{Q}$ to optimum using a large k can provide better performance (but is slower to train). In our experiments, we fix $\lambda = 0.1$ and $k = 5$ for all experiments (train 5 $\mathcal{Q}$ iterations per $\mathcal{F}$ and $\mathcal{P}$ iteration), and the performance is stable over a large set of hyperparameters (Section 3.3.3).

3.3 Experiments and Discussions

To demonstrate the effectiveness of our model, we experiment on Chinese and Arabic text classification, using English as the SOURCE for both. For all data used in experiments, tokenization is done using Stanford CoreNLP (Manning et al., 2014).

3.3.1 Data

Labeled English Data. We use a balanced dataset of 700*k* Yelp reviews from Zhang et al. (2015) with their ratings as labels (scale 1-5). We also adopt their train-validation split: 650*k* reviews for training and 50*k* form a validation set.

Labeled Chinese Data. Since LAN does not require labeled Chinese data for training, this annotated data is solely used to validate the performance of our model. 10*k* balanced Chinese hotel reviews from Lin et al. (2015) are used as validation set for model selection and parameter tuning. The results are reported on a separate test set of another 10*k* hotel reviews. For Chinese, the data are annotated with 5 labels (1-5).

Unlabeled Chinese Data. For the unlabeled TARGET data used in training LAN, we use another 150*k* unlabeled Chinese hotel reviews.

English-Chinese Bilingual Word Embeddings. For Chinese, we used the pre-trained bilingual word embeddings (BWE) by Zou et al. (2013). Their work provides 50-dimensional embeddings for 100*k* English words and another set of 100*k* Chinese words. See Section 3.3.3 for more experiments and discussions.

Labeled Arabic Data. We use the BBN Arabic dataset (Mohammad et al., 2016) for Arabic text classification. The dataset contains 1200 sentences (600 validation + 600 test) from social media posts annotated with 3 labels (−, 0, +). The dataset also provides machine translated text to English. Since the label set does not match with the English dataset, we map all the rating 4 and 5 English instances to + and the rating 1 and 2 instances to −, while the rating 3 sentences are converted to 0.

Methodology	Approach	Accuracy	
		Chinese	Arabic
Train-on-SOURCE-only	Logistic Regression	30.58%	45.83%
	DAN	29.11%	48.00%
Domain Adaptation	mSDA (Chen et al., 2012)	31.44%	48.33%
Machine Translation	Logistic Regression + MT	34.01%	51.67%
	DAN + MT	39.66%	52.50%
CLD-based CLTC	CLD-KCNN†	40.96%	52.67%‡
	CLDFA-KCNN†	41.82%	53.83%‡
Ours	LAN	**42.49%**±0.19%	**54.54%**±0.34%

† Xu and Yang (2017).
‡ As Xu and Yang (2017) did not report results for Arabic, these numbers are obtained based on our reproduction using their code.

Table 3.1: LAN performance for Chinese (5-class) and Arabic (3-class) text classification without using labeled TARGET data. All systems but the CLD ones use BWE to map SOURCE and TARGET words into the same space. CLD-based CLTC represents cross-lingual text classification methods based on cross-lingual distillation (Xu and Yang, 2017) and is explained in Section 3.3.2. For LAN, average accuracy and standard errors over five runs are shown. Bold numbers indicate statistical significance over all baseline systems with $p < 0.05$ under a One-Sample T-Test.

Unlabeled Arabic Data. For Arabic, no additional unlabeled data is used. We only use the text from the validation set (without labels) during training.

English-Arabic Bilingual Word Embeddings. For Arabic, we train a 300d BilBOWA BWE (Gouws et al., 2015) on the United Nations corpus (Ziemski et al., 2016).

3.3.2 Cross-Lingual Text Classification

Our main results are shown in Table 3.1, which shows very similar trends for Chinese and Arabic. Before delving into discussions on the performance of LAN

compared to various baseline systems in the following paragraphs, we begin by clarifying the bilingual resources used in all the methods. Note first that in all of our experiments, traditional features like bag of words cannot be directly used since SOURCE and TARGET have completely different vocabularies. Therefore, unless otherwise specified, BWEs are used as the input representation for all systems to map words from both SOURCE and TARGET into the same feature space. (The only exceptions are the CLD-based CLTC systems of Xu and Yang (2017) explained later in this section, which directly make use of a parallel corpus instead of relying on BWEs.) The same BWEs are adopted in all systems that utilize BWEs.

Train-on-SOURCE-only baselines We start by considering two baselines that train only on the SOURCE language, English, and rely solely on the BWEs to classify the TARGET. The first variation uses a standard supervised learning algorithm, Logistic Regression (LR), shown in Row 1 in Table 3.1. In addition, we evaluate a non-adversarial variation of LAN, just the DAN portion of our model (Row 2), which is one of the modern neural models for text classification. We can see from Table 3.1 that, in comparison to LAN (bottom line), the train-on-SOURCE-only baselines perform poorly. This indicates that BWEs by themselves do not suffice to transfer knowledge of English text classification to TARGET.

Domain Adaptation baselines We next compare LAN with domain adaptation baselines, since domain adaptation can be viewed as a generalization of the cross-lingual task. Nonetheless, the divergence between languages is much more significant than the divergence between two domains, which are typically two product categories in practice. Among domain adaptation meth-

ods, the widely-used TCA (Pan et al., 2011) did not work since it required quadratic space in terms of the number of samples (650k). We thus compare to mSDA (Chen et al., 2012), a very effective method for cross-domain text classification on Amazon reviews. However, as shown in Table 3.1 (Row 3), mSDA did not perform competitively. We speculate that this is because many domain adaptation models including mSDA were designed for the use of bag-of-words features, which are ill-suited in our task where the two languages have completely different vocabularies. In summary, this suggests that even strong domain adaptation algorithms cannot be used out of the box with BWEs for the CLTC task.

Machine Translation baselines We then evaluate LAN against Machine Translation baselines (Rows 4-5) that (1) translate the TARGET text into English and then (2) use the better of the train-on-SOURCE-only models for text classification. Previous studies (Banea et al., 2008; Salameh et al., 2015) on Arabic and European languages claim this MT approach to be very competitive and find that it can sometimes match the state-of-the-art system trained on that language. For Chinese, where translated text was not provided, we use the commercial Google Translate engine[4], which is highly engineered, trained on enormous resources, and arguably one of the best MT systems currently available. As shown in Table 3.1, our LAN model substantially outperforms the MT baseline on both languages, indicating that our adversarial model can successfully perform cross-lingual text classification without any annotated data in the target language.

[4] https://translate.google.com

Cross-lingual Text Classification baselines Finally, we conclude LAN's effectiveness by comparing against a state-of-the-art CLTC method (Xu and Yang, 2017). They propose a cross-lingual distillation (CLD) method that makes use of soft SOURCE predictions on a parallel corpus to train a TARGET model (CLD-KCNN). They further propose an improved variant (CLDFA-KCNN) that utilizes adversarial training to bridge the *domain* gap between the labeled and unlabeled texts within the source and the target language, similar to the adversarial domain adaptation by Ganin et al. (2016). In other words, CLDFA-KCNN consists of three conceptual adaptation steps: (i) Domain adaptation from source-language labeled texts to source-language unlabeled texts using adversarial training; (ii) Cross-lingual adaptation using distillation; and (iii) Domain adaptation in the target language from unlabeled texts to the test set. Note, however, Xu and Yang (2017) use adversarial training for domain adaptation within a single language vs. our work that uses adversarial training directly for cross-lingual generalization.

As shown in Table 3.1, LAN significantly outperforms both variants of CLD-KCNN and achieves a new state of the art performance, indicating that our direct use of adversarial neural nets for cross-lingual adaptation can be more effective than chaining three adaptation steps as in CLDFA-KCNN. This is the case in spite of the fact that LAN does not explicitly separate language variation from domain variation. In fact, the monolingual data we use for the source and target languages is indeed from different domains. LAN's performance suggests that it could potentially bridge the divergence introduced by both sources of variation in one shot.

Supervised SOURCE accuracy By way of comparison, it is also instructive to compare LAN's "transferred" accuracy on the TARGET with its (supervised) performance on the SOURCE. LAN achieves 58.7% accuracy on English for the 5-class English-Chinese setting, and 75.6% for the 3-class English-Arabic setting. The SOURCE accuracy for the DAN baselines (Rows 2 and 5) is similar to the SOURCE accuracy of LAN.

3.3.3 Analysis and Discussion

Since the Arabic dataset is small, we choose Chinese as an example for our further analysis.

Semi-supervised Learning

In practice, it is usually not very difficult to obtain at least a small amount of annotated data. LAN can be readily adapted to exploit such extra labeled data in the target language, by letting those labeled instances pass through the text classifier $\mathcal{P}$ as the English samples do during training. We simulate this semi-supervised scenario by adding labeled Chinese reviews for training. We start by adding 100 labeled reviews and keep doubling the number until 12800. As shown in Figure 3.2, when adding the same number of labeled reviews, LAN can better utilize the extra supervision and outperform the DAN baseline trained with combined data, as well as the supervised DAN using only labeled Chinese reviews. The margin is naturally decreasing as more supervision is incorporated, but LAN is still superior when adding 12800 labeled reviews. On the other hand, the DAN with translation baseline seems unable to effectively uti-

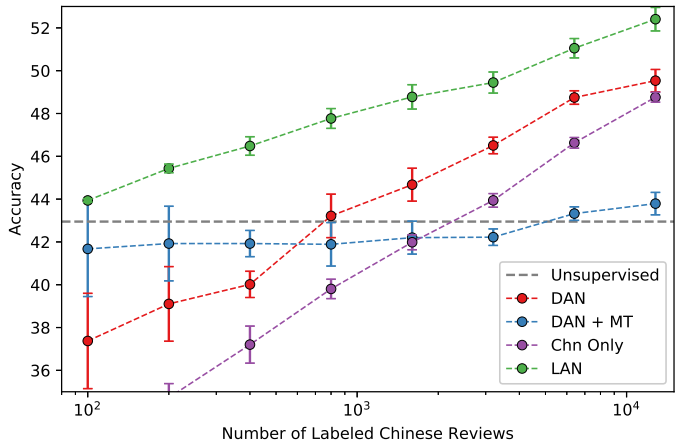

Figure 3.2: LAN performance and standard deviation for Chinese in the semi-supervised setting when using various amount of labeled Chinese data.

lize the added supervision in Chinese, and the performance only starts to show a slightly increasing trend when adding 6400 or more labeled reviews. One possible reason is that when adding to the training data a small number of English reviews translated from the labeled Chinese data, the training signals they produce might be lost in the vast number of English training samples, and thus not effective in improving performance. Another potentially interesting find is that it seems a very small amount of supervision (e.g. 100 labels) could significantly help DAN. However, with the same number of labeled reviews, LAN still outperforms the DAN baseline.

Qualitative Analysis and Visualizations

To qualitatively demonstrate how LAN bridges the distributional discrepancies between English and Chinese instances, t-SNE (Van der Maaten and Hinton, 2008) visualizations of the activations at various layers are shown in Figure 3.3. We randomly select 1000 reviews from the Chinese and English validation sets

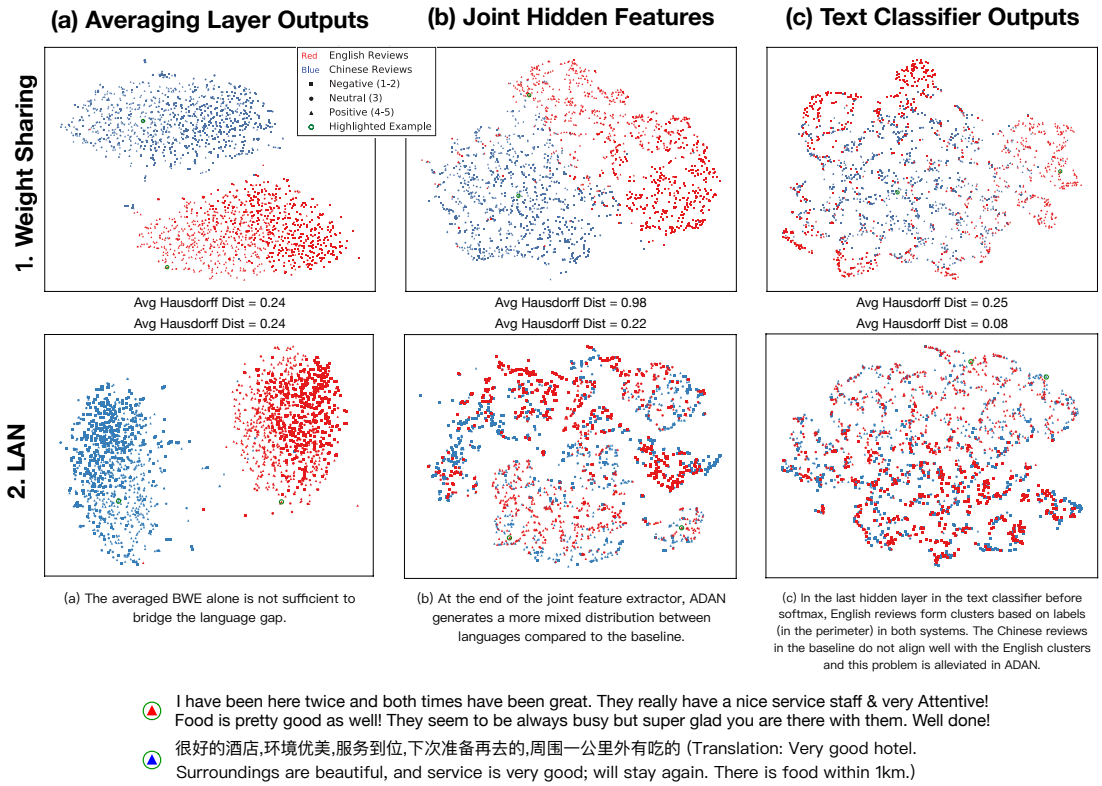

Figure 3.3: t-SNE visualizations of activations at various layers for the weight sharing (train-on-SOURCE-only) baseline model (top) and LAN (bottom). The distributions of the two languages are brought much closer in LAN as they are represented deeper in the network (left to right) measured by the Averaged Hausdorff Distance (see text). The green circles are two 5-star example reviews (shown below the figure) that illustrate how the distribution evolves (zoom in for details).

respectively, and plot the t-SNE of the hidden node activations at three locations in our model: the averaging layer, the end of the joint feature extractor, and the last hidden layer in the text classifier just prior to softmax. The train-on-English model is the DAN baseline in Table 3.1. Note that there is actually only one "branch" in this baseline model, but in order to compare to LAN, we conceptually treat the first three layers as the feature extractor.

Figure 3.3a shows that BWEs alone do not suffice to bridge the gap between the distributions of the two languages. To shed more light on the surprisingly clear separation given that individual words have a mixed distribution in both languages (not shown in figure), we first try to isolate the content divergence from the language divergence. In particular, the English and Chinese reviews are not translations of each other, and in fact may even come from different domains. Therefore, the separation could potentially come from two sources: the content divergence between the English and Chinese reviews, and the language divergence of how words are used in the two languages. To control for content divergence, we tried plotting (not shown in figure) the average word embeddings of 1000 random Chinese reviews and their machine translations into English using t-SNE, and surprisingly the clear separation was still present. There are a few relatively short reviews that reside close to their translations, but the majority still form two language islands. (The same trend persists when we switch to a different set of pre-trained BWEs, and when we plot a similar graph for English-Arabic.) When we remove stop words (the most frequent word types in both languages), the two islands finally start to become slightly closer with less clean boundaries, but the separation remains clear. We think this phenomenon is interesting, and a thorough investigation is out of the scope of this work. We hypothesize that at least in certain distant language pairs such as English-Chinese[5], the divergence between languages may not only be determined by word semantics, but also largely depends on how words are used.

Furthermore, we can see in Figure 3.3b that the distributional discrepancies between Chinese and English are significantly reduced after passing through the joint feature extractor ($\mathcal{F}$). The learned features in `LAN` bring the distribu-

[5]In a personal correspondence with Ahmed Elgohary, he did not observe the same phenomenon between English and French.

tions in the two languages dramatically closer compared to the monolingually trained baseline. This is shown via the Averaged Hausdorff Distance (AHD, Shapiro and Blaschko, 2004), which measures the distance between two sets of points. The AHD between the English and Chinese reviews is provided for all sub-plots in Figure 3.3.

Finally, when looking at the last hidden layer activations in the text classifier of the baseline model (Figure 3.3c), there are several notable clusters of red dots (English data) that roughly correspond to the class labels. These English clusters are the areas where the classifier is the most confident in making decisions. However, most Chinese samples are not close to one of those clusters due to the distributional divergence and may thus cause degraded classification performance in Chinese. On the other hand, the Chinese samples are more in line with the English ones in `LAN`, which results in the accuracy boost over the baseline model. In Figure 3.3, a pair of similar English and Chinese 5-star reviews is highlighted to visualize how the distribution evolves at various points of the network. We can see in 3.3c that the highlighted Chinese review gets close to the "positive English cluster" in `LAN`, while in the baseline, it stays away from dense English clusters where the text classifier trained on English data is not confident to make predictions.

Impact of Bilingual Word Embeddings

In this section we discuss the effect of the bilingual word embeddings. We start by initializing the systems with random word embeddings (WEs), shown in Table 3.2. `LAN` with random WEs outperforms the `DAN` and mSDA baselines using BWEs and matches the performance of the LR+MT baseline (Table 3.1),

Model	Random	BilBOWA	Zou et al.
DAN	21.66%	28.75%	29.11%
DAN+MT	37.78%	38.17%	39.66%
LAN	34.44%	40.51%	42.95%

Table 3.2: Model performance on Chinese with various (B)WE initializations.

suggesting that LAN successfully extracts features that could be used for cross-lingual classification tasks without *any* bitext. This impressive result vindicates the power of adversarial training to reduce the distance between two complex distributions without any direct supervision, which is also observed in more recent works for other tasks (Zhang et al., 2017; Lample et al., 2018).

With the introduction of BWEs (Column 2 and 3), the performance of LAN is further boosted. Therefore, the quality of the BWEs plays an important role in CLTC. To investigate the impact of the specific choice of BWEs, we also trained 100d BilBOWA BWEs (Gouws et al., 2015) using the UN parallel corpus for Chinese. All systems achieve slightly lower performance compared to the pre-trained BWEs from Zou et al. (2013), yet LAN still outperforms other baseline methods (Table 3.2), demonstrating that LAN's effectiveness is relatively robust with respect to the choice of BWEs. We conjecture that all systems show inferior results with BilBOWA, because it does not require word alignments during training as Zou et al. (2013) do. By only training on a sentence-aligned corpus, BilBOWA requires less resources and is much faster to train, potentially at the expense of quality.

Model	Accuracy	Run time
DAN	42.95%	0.127 (s/iter)
CNN	46.24%	0.554 (s/iter)
BiLSTM	44.55%	1.292 (s/iter)
BiLSTM + dot attn	46.41%	1.898 (s/iter)

Table 3.3: Performance and speed for various feature extractor architectures on Chinese.

Feature Extractor Architectures

As mentioned in Section 3.2.1, the architecture of LAN's feature extractor is not limited to a Deep Averaging Network (DAN), and one can choose different feature extractors to suit a particular task or dataset. While an extensive study of alternative architectures is beyond the scope of this work, we in this section present a brief experiment illustrating that our adversarial framework works well with other $\mathcal{F}$ architectures. In particular, we consider two popular choices: i) a CNN (Kim, 2014) that has a 1d convolutional layer followed by a single fully-connected layer to extract a fixed-length vector; and ii) a Bi-LSTM with two variants: one that takes the average of the hidden outputs of each token as the feature vector, and one with the dot attention mechanism (Luong et al., 2015) that learns a weighted linear combination of all hidden outputs.

As shown in Table 3.3, LAN's performance can be improved by adopting more sophisticated feature extractors, at the expense of slower running time. This demonstrates that LAN's language-adversarial training framework can be successfully used with other $\mathcal{F}$ choices.

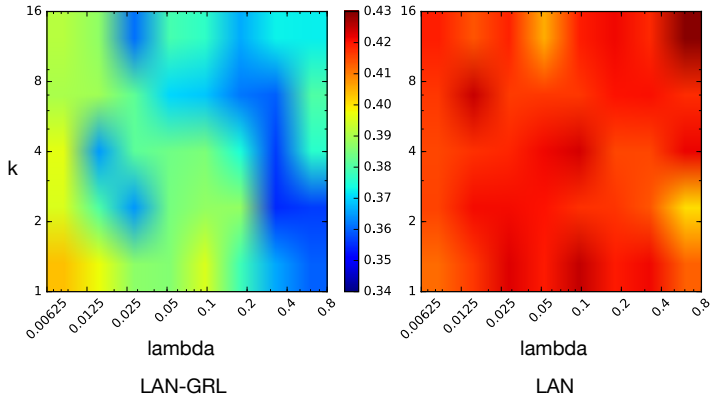

Figure 3.4: A grid search on k and *lambda* for LAN (right) and the LAN-GRL variant (left). Numbers indicate the accuracy on the Chinese development set.

LAN Hyperparameter Stability

In this section, we show that the training of LAN is stable over a large set of hyperparameters, and provides improved performance compared to the standard LAN-GRL.

To verify the superiority of LAN, we conduct a grid search over k and λ, which are the two hyperparameters shared by LAN and LAN-GRL. We experiment with $k \in \{1, 2, 4, 8, 16\}$, and $\lambda \in \{0.00625, 0.0125, 0.025, 0.05, 0.1, 0.2, 0.4, 0.8\}$. Figure 3.4 reports the accuracy on the Chinese dev set for both LAN variants, and shows that LAN attains higher accuracy and greater stability. This suggests that LAN overcomes the well-known problem that adversarial training is sensitive to hyperparameter tuning.

3.3.4 Implementation Details

For all our experiments on both languages, the feature extractor $\mathcal{F}$ has three fully-connected layers with ReLU non-linearities, while both $\mathcal{P}$ and $\mathcal{Q}$ have two. All hidden layers contain 900 hidden units. Batch Normalization (Ioffe and Szegedy, 2015) is used in each hidden layer in $\mathcal{P}$ and $\mathcal{Q}$. $\mathcal{F}$ does not use batch normalization. $\mathcal{F}$ and $\mathcal{P}$ are optimized jointly using `Adam` (Kingma and Ba, 2015) with a learning rate of 0.0005. $\mathcal{Q}$ is trained with another `Adam` optimizer with the same learning rate. The weights of $\mathcal{Q}$ are clipped to $[-0.01, 0.01]$. We train `LAN` for 30 epochs and use early stopping to select the best model on the validation set. `LAN` is implemented in `PyTorch` (Paszke et al., 2017)[6].

3.4 Chapter Summary

In this chapter, we described Language-Adversarial Training, a model-based method for cross-lingual model transfer using only monolingual data for training. It learns a language-invariant hidden feature space better suited for cross-lingual transfer by attempting to fool an adversarially trained language discriminator whose goal is to identify the language of a sample by looking at its feature vector. We further proposed `LAN`, a language-adversarial network for cross-lingual text classification. `LAN` leverages the abundant labeled data from a resource-rich source language such as English to help text classification on other languages where little or no annotated data exist.

We validate `LAN`'s effectiveness by experiments on Chinese and Arabic text

[6]The source code of `LAN` is available at: `https://github.com/ccsasuke/adan`

classification, where we have labeled English data and only *unlabeled* data in the target language. Experiments show that LAN outperforms several baselines including domain adaptation models, a competitive MT baseline, and previous state-of-the-art cross-lingual text classification methods. We further show that even without *any* bilingual resources, LAN trained with randomly initialized embeddings can still achieve encouraging performance. In addition, we show that in the presence of labeled data in the target language, LAN can naturally incorporate this additional supervision and yields even more competitive results.

CHAPTER 4

MULTINOMIAL ADVERSARIAL NETWORKS

In the previous chapter, we introduced language-adversarial training that could learn a language-invariant hidden feature space between two different languages without parallel training data. Formally speaking, adversarial training can serve as a tool for minimizing the divergence between two probabilistic distributions without paired supervision (Nowozin et al., 2016). In this chapter, we propose the Multinomial Adversarial Network (`MAN`), a theoretically sound generalization of standard adversarial networks into a more realistic scenario that can handle *multiple* populations (e.g. languages or domains) by leveraging a *multinomial discriminator* to directly minimize the divergence among multiple probability distributions. As the standard binomial adversarial networks have been successfully applied to numerous tasks including image generation (Goodfellow et al., 2014), domain adaptation (Ganin et al., 2016) and cross-lingual text classification (Chen et al., 2016, Chapter 3), we anticipate that `MAN`s will make a versatile machine learning framework with applications beyond the ones studied in this chapter.

This chapter is based on Chen and Cardie (2018a).

To isolate the impact of the quality of cross-lingual lexical representation, we in this chapter focus on the task of domain adaptation (or cross-domain model transfer) instead of cross-lingual model transfer.[1] In particular, we again focus on the fundamental NLP task in this chapter: text classification. Similar to the cross-lingual text classification case, text classification also faces the

[1]The techniques introduced in this chapter can be (and have been) applied to cross-lingual model transfer when combined with cross-lingual word embeddings as shown in Chapter 5.

data scarcity problem in the multi-domain setting. In fact, many text classification tasks are highly domain-dependent in that a text classifier trained using labeled data from one domain is likely to perform poorly on another. In the task of sentiment classification, for example, the phrase "runs fast" is usually associated with positive sentiment in the sports domain; not so when a user is reviewing the battery of an electronic device. In real applications, therefore, an adequate amount of training data from each domain of interest is typically required, which is expensive to obtain.

Two major lines of work attempt to tackle this challenge: **domain adaptation** (Blitzer et al., 2007) and **multi-domain text classification** (MDTC) (Li and Zong, 2008). In domain adaptation, the assumption is that there is some domain with abundant training data (the source domain), and the goal is to utilize knowledge learned from the source domain to help perform classifications on another lower-resourced target domain.[2] Our focus, MDTC, instead simulates an arguably more realistic scenario, where labeled data may exist for multiple domains, but in insufficient amounts to train an effective classifier for one or more of the domains. Worse still, some domains may have *no* labeled data at all. The objective of MDTC is to leverage *all* the available resources in order to improve the system performance over all domains simultaneously.

In the rest of this chapter, we first introduce the general Multinomial Adversarial Networks architecture in Section 4.2 and prove in Section 4.3 that it directly minimizes the (generalized) f-divergence among multiple distributions so that they are indistinguishable upon successful training. Specifically for MDTC, `MAN` is used to overcome a limitation in prior art where domain-specific features may sneak into the domain-agnostic feature space (Section 4.1). This is accom-

[2]See Section 4.1 for other variants of domain adaptation.

plished by relying on MAN's power of minimizing the divergence among the feature distributions of each domain. The high-level idea is that MAN will make the extracted feature distributions of each domain indistinguishable from one another, thus learning features that are invariant across domains.

We then validate the effectiveness of MAN in experiments on two MDTC data sets. We find first that MAN significantly outperforms the state-of-the-art CMSC method (Wu and Huang, 2015) on the widely used multi-domain Amazon review dataset, and does so without relying on external resources such as sentiment lexica (Section 4.4.1). When applied to the second dataset, FDU-MTL (Section 4.4.3), we obtain similar results: MAN achieves substantially higher accuracy than the previous top-performing method, ASP-MTL (Liu et al., 2017). Finally, while many MDTC methods such as CMSC require labeled data for each domain, MANs can be applied in cases where no labeled data exists for a subset of domains. To evaluate MAN in this semi-supervised setting, we compare MAN to a method that can accommodate unlabeled data for (only) one domain (Zhao et al., 2018), and show that MAN achieves performance comparable to the state of the art (Section 4.4.2).

4.1 Related Work

Multi-Domain Text Classification The MDTC task was first examined by Li and Zong (2008), who proposed to fuse the training data from multiple domains either at the feature level or the classifier level. One state-of-the-art system for MDTC, the CMSC system of Wu and Huang (2015), combines a classifier that is shared across all domains (for learning domain-invariant knowledge) with a set of classifiers, one per domain, each of which captures domain-specific text clas-

sification knowledge. This paradigm is sometimes known as the Shared-Private model (Bousmalis et al., 2016). CMSC, however, lacks an explicit mechanism to ensure that the shared classifier captures only domain-independent knowledge: the shared classifier may well also acquire some domain-specific features that are useful for a subset of the domains. We hypothesize that better performance can be obtained if this constraint were explicitly enforced.

Domain Adaptation Domain Adaptation attempts to transfer the knowledge from a source domain to a target one, and the traditional form is the *single-source, single-target* (**SS,ST**) adaptation (Blitzer et al., 2006). Another variant is the **SS,MT** adaptation (Yang and Eisenstein, 2015), which tries to simultaneously transfer the knowledge to multiple target domains from a single source. However, it cannot fully take advantage the training data if it comes from multiple source domains. **MS,ST** adaptation (Mansour et al., 2009; Zhao et al., 2018) can deal with multiple source domains but only transfers to a single target domain. Therefore, when multiple target domains exist, they need to treat them as independent tasks, which is more expensive and cannot utilize the additional unlabeled data in these domains. Finally, MDTC can be viewed as **MS,MT** adaptation, which is arguably more general and realistic.

Adversarial Networks The idea of adversarial networks was proposed by Goodfellow et al. (2014) for image generation, and has been applied to various NLP tasks as well (Chen et al., 2016; Yu et al., 2017). Ganin et al. (2016) first used it for the **SS,ST** domain adaptation followed by many others. Bousmalis et al. (2016) utilized adversarial training in a shared-private model for domain adaptation to learn domain-invariant features, but still focused on the **SS,ST**

setting. Finally, the idea of using adversarial nets to discriminate over multiple distributions was empirically explored by a very recent work (Liu et al., 2017) under the multi-task learning setting, and can be considered as a special case of our MAN framework with the NLL domain loss. We in this chapter propose MAN as a more general framework with alternative architectures for the adversarial component, and for the first time provide theoretical justifications the multinomial adversarial nets. Moreover, Liu et al. (2017) used a LSTM without attention as their feature extractor, which we found to perform sub-optimal in the experiments. We instead chose Convolutional Neural Nets as our feature extractor that achieves higher accuracy while running an order of magnitude faster (see Section 4.4.3).

4.2 Multinomial Adversarial Networks (MAN)

In this chapter, we tackle the text classification problem in the real-world setting in which texts come from a variety of domains, each with a varying amount of labeled data. Specifically, assume we have a total of N domains, N_1 *labeled domains* (denoted as Δ_L) for which there is some labeled data, and N_2 *unlabeled domains* (Δ_U) for which no annotated training instances are available. Denote $\Delta = \Delta_L \cup \Delta_U$ as the collection of all domains, with $N = N_1 + N_2$. The goal of this work, and of MDTC in general, is to improve the overall classification performance across all N domains, measured in this paper as the average[3] classification accuracy across the N domains in Δ.

[3]In this work, we use macro-average over domains, but MAN can be readily adapted for micro-average or other (weighted) averaging schemes.

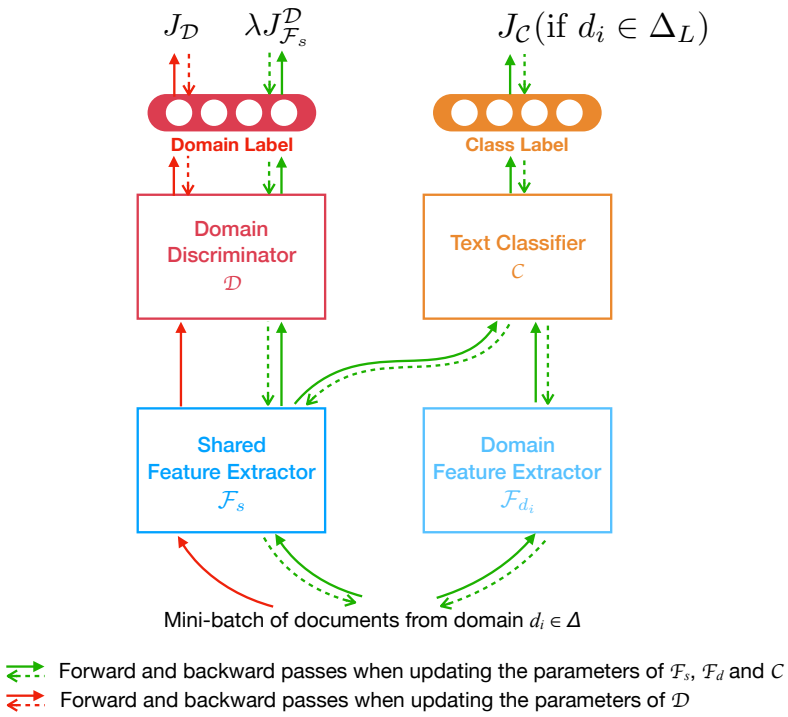

Figure 4.1: MAN for MDTC. The figure demonstrates the training on a mini-batch of data from one domain. One training iteration consists of one such mini-batch training from each domain. The parameters of $\mathcal{F}_s, \mathcal{F}_d, C$ are updated together, and the training flows are illustrated by the green arrows. The parameters of $\mathcal{D}$ are updated separately, shown in red arrows. Solid lines indicate forward passes while dotted lines are backward passes. $J^{\mathcal{D}}_{\mathcal{F}_s}$ is the domain loss for $\mathcal{F}_s$, which is anticorrelated with $J_{\mathcal{D}}$ (e.g. $J^{\mathcal{D}}_{\mathcal{F}_s} = -J_{\mathcal{D}}$). (See Section 4.2 and Section 4.3)

4.2.1 Model Architecture

As shown in Figure 4.1, the Multinomial Adversarial Network (MAN) adopts the Shared-Private paradigm of Bousmalis et al. (2016) and consists of four components: a *shared feature extractor* $\mathcal{F}_s$, a *domain feature extractor* $\mathcal{F}_{d_i}$ for each labeled domain $d_i \in \Delta_L$, a *text classifier* C, and a *domain discriminator* $\mathcal{D}$. The main idea of MAN is to explicitly model the domain-invariant features that are beneficial to the main classification task across all domains (i.e. the *shared features*, extracted

by $\mathcal{F}_s$), as well as the domain-specific features that mainly contribute to the classification in its own domain (the *domain features*, extracted by $\mathcal{F}_d$). Here, the adversarial domain discriminator $\mathcal{D}$ has a multinomial output that takes a shared feature vector and predicts the likelihood of that sample coming from each domain. As seen in Figure 4.1, during the training of $\mathcal{F}_s$ (green arrows denote the training flow), $\mathcal{F}_s$ aims to confuse $\mathcal{D}$ by minimizing $J^{\mathcal{D}}_{\mathcal{F}_s}$, which is anticorrelated to $J_{\mathcal{D}}$ (detailed in Section 4.2.2), so that $\mathcal{D}$ cannot predict the domain of a sample given its shared features. The intuition is that if even a strong discriminator $\mathcal{D}$ cannot tell the domain of a sample from the extracted features, those features $\mathcal{F}_s$ learned are essentially domain invariant. By enforcing domain-invariant features to be learned by $\mathcal{F}_s$, when trained jointly via backpropagation, the set of domain feature extractors $\mathcal{F}_d$ will each learn domain-specific features beneficial within its own domain.

The architecture of each component is relatively flexible, and can be decided by the practitioners to suit their particular classification tasks. For instance, the feature extractors can adopt the form of Convolutional Neural Nets (CNN), Recurrent Neural Nets (RNN), or a Multi-Layer Perceptron (MLP), depending on the input data (see Section 4.4). The input of `MAN` will also be dependent on the feature extractor choice. The output of a (shared/domain) feature extractor is a fixed-length vector, which is considered the (shared/domain) hidden features of some given input text. On the other hand, the outputs of $\mathcal{C}$ and $\mathcal{D}$ are label probabilities for class and domain prediction, respectively. For example, both $\mathcal{C}$ and $\mathcal{D}$ can be MLPs with a softmax layer on top. In Section 4.3, we provide alternative architectures for $\mathcal{D}$ and their mathematical implications. We now present a detailed description of the `MAN` training in Section 4.2.2 as well as the theoretical grounds in Section 4.3.

Require: labeled corpus $\mathbb{X}$; unlabeled corpus $\mathbb{U}$; Hyperpamameter $\lambda > 0$, $k \in \mathbb{N}$

repeat
  ▷ $\mathcal{D}$ *iterations*
  for *diter* = 1 to k **do**
    $l_{\mathcal{D}} = 0$
    for all $d \in \Delta$ **do** ▷ For all N domains
      Sample a mini-batch $\boldsymbol{x} \sim \mathbb{U}_d$
      $\boldsymbol{f}_s = \mathcal{F}_s(\boldsymbol{x})$ ▷ Shared feature vector
      $l_{\mathcal{D}}$ += $J_{\mathcal{D}}(\mathcal{D}(\boldsymbol{f}_s); d)$ ▷ Accumulate $\mathcal{D}$ loss
    Update $\mathcal{D}$ parameters using $\nabla l_{\mathcal{D}}$
  ▷ *Main iteration*
  $loss = 0$
  for all $d \in \Delta_L$ **do** ▷ For all labeled domains
    Sample a mini-batch $(\boldsymbol{x}, \boldsymbol{y}) \sim \mathbb{X}_d$
    $\boldsymbol{f}_s = \mathcal{F}_s(\boldsymbol{x})$
    $\boldsymbol{f}_d = \mathcal{F}_d(\boldsymbol{x})$ ▷ Domain feature vector
    $loss$ += $J_C(C(\boldsymbol{f}_s, \boldsymbol{f}_d); \boldsymbol{y})$ ▷ Compute C loss
  for all $d \in \Delta$ **do** ▷ For all N domains
    Sample a mini-batch $\boldsymbol{x} \sim \mathbb{U}_d$
    $\boldsymbol{f}_s = \mathcal{F}_s(\boldsymbol{x})$
    $loss$ += $\lambda \cdot J^{\mathcal{D}}_{\mathcal{F}_s}(\mathcal{D}(\boldsymbol{f}_s); d)$ ▷ Domain loss of $\mathcal{F}_s$
  Update $\mathcal{F}_s$, $\mathcal{F}_d$, C parameters using $\nabla loss$
until convergence

Algorithm 4.1: MAN Training

4.2.2 MAN Training

Denote the annotated corpus in a labeled domain $d_i \in \Delta_L$ as $\mathbb{X}_i$; and $(x, y) \sim \mathbb{X}_i$ is a sample drawn from the labeled data in domain d_i, where x is the input and y is the task label. On the other hand, for any domain $d_{i'} \in \Delta$, denote the unlabeled corpus as $\mathbb{U}_{i'}$. Note for the choice of unlabeled data of a labeled domain, one can use a separate unlabeled corpus or simply use the labeled data (or use both).

In Figure 4.1, the arrows illustrate the training flows of various components. Due to the adversarial nature of the domain discriminator $\mathcal{D}$, it is trained with a separate optimizer (red arrows), while the rest of the networks are updated

with the main optimizer (green arrows). C is only trained on the annotated data from labeled domains, and it takes as input the concatenation of the shared and domain feature vectors. At test time, for data from unlabeled domains with no $\mathcal{F}_d$, the domain features are set to the $\mathbf{0}$ vector for C's input. On the contrary, $\mathcal{D}$ only takes the shared features as input, for both labeled and unlabeled domains. The MAN training procedure is described in Algorithm 4.1.

In Algorithm 4.1, $\mathcal{L}_C$ and $\mathcal{L}_\mathcal{D}$ are the loss functions of the text classifier C and the domain discriminator $\mathcal{D}$, respectively. As mentioned in Section 4.2.1, C has a $softmax$ layer on top for classification. We hence adopt the canonical negative log-likelihood (NLL) loss:

$$\mathcal{L}_C(\hat{y}, y) = -\log P(\hat{y} = y) \tag{4.1}$$

where y is the true label and $\hat{y}$ is the $softmax$ predictions. For $\mathcal{D}$, we consider two variants of MAN. The first one is to use the NLL loss same as C which suits the classification task; while another option is to use the Least-Square (L2) loss that was shown to be able to alleviate the gradient vanishing problem when using the NLL loss in the adversarial setting (Mao et al., 2017):

$$\mathcal{L}_\mathcal{D}^{NLL}(\hat{d}, d) = -\log P(\hat{d} = d) \tag{4.2}$$

$$\mathcal{L}_\mathcal{D}^{L2}(\hat{d}, d) = \sum_{i=1}^{N} (\hat{d}_i - \mathbb{1}_{\{d=i\}})^2 \tag{4.3}$$

where d is the domain index of some sample and $\hat{d}$ is the prediction. Without loss of generality, we normalize $\hat{d}$ so that $\sum_{i=1}^{N} \hat{d}_i = 1$ and $\forall i : \hat{d}_i \geq 0$.

Therefore, the objectives of C and $\mathcal{D}$ that we are minimizing are:

$$J_C = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{X}_i} [\mathcal{L}_C(C(\mathcal{F}_s(x), \mathcal{F}_d(x)); y)] \tag{4.4}$$

$$J_\mathcal{D} = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{x\sim\mathbb{U}_i} [\mathcal{L}_\mathcal{D}(\mathcal{D}(\mathcal{F}_s(x)); d)] \tag{4.5}$$

For the feature extractors, the training of domain feature extractors is straightforward, as their sole objective is to help C perform better within their own domain. Hence, $J_{\mathcal{F}_d} = J_C$ for any domain d. Finally, the shared feature extractor $\mathcal{F}_s$ has two objectives: to help C achieve higher accuracy, and to make the feature distribution invariant across all domains. It thus leads to the following bipartite loss:

$$J_{\mathcal{F}_s} = J^{\mathcal{C}}_{\mathcal{F}_s} + \lambda \cdot J^{\mathcal{D}}_{\mathcal{F}_s} \tag{4.6}$$

where λ is a hyperparameter balancing the two parts. $J^{\mathcal{D}}_{\mathcal{F}_s}$ is the domain loss of $\mathcal{F}_s$ anticorrelated to $J_{\mathcal{D}}$:

$$(NLL) \quad J^{\mathcal{D}}_{\mathcal{F}_s} = -J_{\mathcal{D}} \tag{4.7}$$

$$(L2) \quad J^{\mathcal{D}}_{\mathcal{F}_s} = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{x \sim \mathbb{U}_i} \left[\sum_{j=1}^{N} \left(\mathcal{D}_j \left(\mathcal{F}_s(x) \right) - \frac{1}{N} \right)^2 \right] \tag{4.8}$$

If $\mathcal{D}$ adopts the NLL loss (4.7), the domain loss is simply $-J_{\mathcal{D}}$. For the L2 loss (4.8), $J^{\mathcal{D}}_{\mathcal{F}_s}$ intuitively translates to pushing $\mathcal{D}$ to make random predictions. See Section 4.3 for theoretical justifications.

4.3 Theoretical Analysis of Multinomial Adversarial Networks

Binomial adversarial nets are known to have theoretical connections to the minimization of various f-divergences[4] between *two* distributions (Nowozin et al., 2016). However, for adversarial training among multiple distributions, no theoretical justifications have been provided to our best knowledge.

In this section, we present a theoretical analysis showing the validity of `MAN`.

[4] An f-divergence (Ali and Silvey, 1966) is a function that measures the distance between two probability distributions. For example, KL divergence and Jensen-Shannon divergence are instances of f-divergence.

In particular, we show that MAN's objective is equivalent to minimizing the total f-divergence between each of the shared feature distributions of the N domains, and the centroid of the N distributions. The choice of loss function will determine which specific f-divergence is minimized. Furthermore, with adequate model capacity, MAN achieves its optimum for either loss function if and only if all N shared feature distributions are identical, hence learning an invariant feature space across all domains.

First, consider the distribution of the shared features $\mathbf{f}$ for instances in each domain $d_i \in \Delta$:

$$P_i(\mathbf{f}) \triangleq P(\mathbf{f} = \mathcal{F}_s(x) | x \in d_i) \tag{4.9}$$

Combining (4.5) with the two loss functions (4.2), (4.3), the objective of $\mathcal{D}$ can be written as:

$$J_{\mathcal{D}}^{NLL} = -\sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f} \sim P_i} [\log \mathcal{D}_i(\mathbf{f})] \tag{4.10}$$

$$J_{\mathcal{D}}^{L2} = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f} \sim P_i} \left[\sum_{j=1}^{N} (\mathcal{D}_j(\mathbf{f}) - \mathbb{1}_{\{i=j\}})^2 \right] \tag{4.11}$$

where $\mathcal{D}_i(\mathbf{f})$ is the i-th dimension of $\mathcal{D}$'s (normalized) output vector, which conceptually corresponds to the probability of $\mathcal{D}$ predicting that $\mathbf{f}$ is from domain d_i

We first derive the optimal $\mathcal{D}$ for any fixed $\mathcal{F}_s$.

Lemma 4.1. *For any fixed $\mathcal{F}_s$, with either NLL or L2 loss, the optimum domain discriminator $\mathcal{D}^*$ is:*

$$\mathcal{D}_i^*(\mathbf{f}) = \frac{P_i(\mathbf{f})}{\sum_{j=1}^{N} P_j(\mathbf{f})} \tag{4.12}$$

Proof. Without loss of generality, we normalize $\mathcal{D}$'s outputs to be on a simplex:

$$\sum_{i=1}^{N} \mathcal{D}_i(\mathbf{f}) = 1 \quad (\forall \mathbf{f}) \tag{4.13}$$

When $\mathcal{D}$ adopts the NLL loss, for a fixed $\mathcal{F}_s$, the optimum

$$\begin{aligned}
\mathcal{D}^* = \arg\min_{\mathcal{D}} J_{\mathcal{D}} &= \arg\min_{\mathcal{D}} -\sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} [\log \mathcal{D}_i(\mathbf{f})] \\
&= \arg\max_{\mathcal{D}} \sum_{i=1}^{N} \int_{\mathbf{f}} P_i(\mathbf{f}) \log \mathcal{D}_i(\mathbf{f}) d\mathbf{f} \\
&= \arg\max_{\mathcal{D}} \int_{\mathbf{f}} \sum_{i=1}^{N} P_i(\mathbf{f}) \log \mathcal{D}_i(\mathbf{f}) d\mathbf{f}
\end{aligned}$$

We employ the Lagrangian Multiplier to derive $\arg\max_{\mathcal{D}} \sum_{i=1}^{N} P_i(\mathbf{f}) \log \mathcal{D}_i(\mathbf{f})$ under the constraint of (4.13). Let

$$L(\mathcal{D}_1, \ldots, \mathcal{D}_N, \lambda) = \sum_{i=1}^{N} P_i \log \mathcal{D}_i - \lambda \left(\sum_{i=1}^{N} \mathcal{D}_i - 1 \right)$$

Let $\nabla L = 0$:

$$\begin{cases}
\nabla_{\mathcal{D}_i} \sum_{j=1}^{N} P_j \log \mathcal{D}_j = \lambda \nabla_{\mathcal{D}_i} \left(\sum_{j=1}^{N} \mathcal{D}_j - 1 \right) & (\forall i) \\
\sum_{i=1}^{N} \mathcal{D}_i - 1 = 0
\end{cases}$$

Solving the two equations, we have:

$$\mathcal{D}_i^*(\mathbf{f}) = \frac{P_i(\mathbf{f})}{\sum_{j=1}^{N} P_j(\mathbf{f})}$$

On the other hand, if $\mathcal{D}$ adopts the L2 loss, for a fixed $\mathcal{F}_s$, the optimum

$$\begin{aligned}
\mathcal{D}^* = \arg\min_{\mathcal{D}} J_{\mathcal{D}} &= \arg\min_{\mathcal{D}} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} [\mathcal{L}_{\mathcal{D}}(\mathcal{D}(\mathbf{f}), i)] \\
&= \arg\min_{\mathcal{D}} \sum_{i=1}^{N} \int_{\mathbf{f}} P_i(\mathbf{f}) \mathcal{L}_{\mathcal{D}}(\mathcal{D}(\mathbf{f}), i) d\mathbf{f} \\
&= \arg\min_{\mathcal{D}} \int_{\mathbf{f}} \sum_{i=1}^{N} P_i(\mathbf{f}) \sum_{j=1}^{N} \left(\mathcal{D}_j(\mathbf{f}) - \mathbb{1}_{\{i=j\}} \right)^2 d\mathbf{f}
\end{aligned}$$

Similar to MAN-NLL, we employ the Lagrangian Multiplier to derive

$$\arg\min_{\mathcal{D}} \sum_{i=1}^{N} P_i(\mathbf{f}) \sum_{j=1}^{N} \left(\mathcal{D}_j(\mathbf{f}) - \mathbb{1}_{\{i=j\}}\right)^2$$

under the constraint of (4.13). Let $\nabla L = 0$:

$$\begin{cases} 2\left(\left(\sum_{j=1}^{N} P_j\right)\mathcal{D}_i - P_i\right) = \lambda \quad (\forall i) \\ \sum_{i=1}^{N} \mathcal{D}_i - 1 = 0 \end{cases}$$

Solving the two equations, we have $\lambda = 0$ and again:

$$\mathcal{D}_i^*(\mathbf{f}) = \frac{P_i(\mathbf{f})}{\sum_{j=1}^{N} P_j(\mathbf{f})}$$

□

We then have the following main theorems for the domain loss for $\mathcal{F}_s$:

Theorem 4.1. *Let* $\overline{P} = \frac{\sum_{i=1}^{N} P_i}{N}$. *When* $\mathcal{D}$ *is trained to its optimality, if* $\mathcal{D}$ *adopts the* NLL loss*:*

$$\begin{aligned} J_{\mathcal{F}_s}^{\mathcal{D}} &= -\min_{\theta_{\mathcal{D}}} J_{\mathcal{D}} = -J_{\mathcal{D}^*} \\ &= -N \log N + N \cdot JSD\left(P_1, P_2, \ldots, P_N\right) \\ &= -N \log N + \sum_{i=1}^{N} KL\left(P_i \| \overline{P}\right) \end{aligned}$$

where $JSD(\cdot)$ is the generalized Jensen-Shannon Divergence (Lin, 1991) among multiple distributions, defined as the average Kullback-Leibler divergence of each P_i to the centroid $\overline{P}$ (Aslam and Pavlu, 2007).

Proof. Substituting the optimal discrminator $\mathcal{D}^*$ into $J^{\mathcal{D}}_{\mathcal{F}_s}$:

$$
\begin{aligned}
J^{\mathcal{D}}_{\mathcal{F}_s} = -J_{\mathcal{D}^*} &= \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\log \mathcal{D}^*_i(\mathbf{f})\right] \\
&= \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\log \frac{P_i(\mathbf{f})}{\sum_{j=1}^{N} P_j(\mathbf{f})}\right] \\
&= -N\log N + \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\log \frac{P_i(\mathbf{f})}{\sum_{j=1}^{N} P_j(\mathbf{f})} + \log N\right] \\
&= -N\log N + \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\log \frac{P_i(\mathbf{f})}{\frac{\sum_{j=1}^{N} P_j(\mathbf{f})}{N}}\right] \\
&= -N\log N + \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\log \frac{P_i(\mathbf{f})}{\overline{P}}\right] \\
&= -N\log N + \sum_{i=1}^{N} KL\left(P_i \| \overline{P}\right) \\
&= -N\log N + N \cdot JSD\left(P_1, P_2, \ldots, P_N\right)
\end{aligned}
$$

□

Theorem 4.2. *Let* $\overline{P} = \frac{\sum_{i=1}^{N} P_i}{N}$. *When* $\mathcal{D}$ *is trained to its optimality, if* $\mathcal{D}$ *uses the* L2 loss*:*

$$
\begin{aligned}
J^{\mathcal{D}}_{\mathcal{F}_s} &= \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\sum_{j=1}^{N} \left(\mathcal{D}^*_j(\mathbf{f}) - \frac{1}{N}\right)^2\right] \\
&= \frac{1}{N} \sum_{i=1}^{N} \chi^2_{Neyman}\left(P_i \| \overline{P}\right)
\end{aligned}
$$

where $\chi^2_{Neyman}(\cdot\|\cdot)$ is the Neyman χ^2 divergence (Nielsen and Nock, 2014).

Proof. Substituting $\mathcal{D}^*$ into $\mathcal{L}^{\mathcal{D}}_{\mathcal{F}_s}$:

$$
\begin{aligned}
J^{\mathcal{D}}_{\mathcal{F}_s} &= \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathbf{f}\sim P_i} \left[\sum_{j=1}^{N} (D^*_j(\mathbf{f}) - \frac{1}{N})^2 \right] \\
&= \sum_{i=1}^{N} \int_{\mathbf{f}} P_i \sum_{j=1}^{N} \left(\frac{P_j}{N\overline{P}} - \frac{1}{N} \right)^2 d\mathbf{f} \\
&= \int_{\mathbf{f}} \sum_{i=1}^{N} \sum_{j=1}^{N} P_i \left(\frac{P_j}{N\overline{P}} - \frac{1}{N} \right)^2 d\mathbf{f} \\
&= \frac{1}{N^2} \sum_{j=1}^{N} \int_{\mathbf{f}} \sum_{i=1}^{N} P_i \left(\frac{P_j}{\overline{P}} - 1 \right)^2 d\mathbf{f} \\
&= \frac{1}{N^2} \sum_{j=1}^{N} \int_{\mathbf{f}} N\overline{P} \left(\frac{P_j}{\overline{P}} - 1 \right)^2 d\mathbf{f} \\
&= \frac{1}{N} \sum_{j=1}^{N} \int_{\mathbf{f}} \frac{\left(P_j - \overline{P} \right)^2}{\overline{P}} d\mathbf{f} \\
&= \frac{1}{N} \sum_{i=1}^{N} \chi^2_{Neyman} \left(P_i \| \overline{P} \right)
\end{aligned}
$$

□

Given the Theorem 4.1 and 4.2, by the non-negativity and joint convexity of the f-divergence (Csiszar and Korner, 1982), we have:

Corollary 4.1. *The optimum of $J^{\mathcal{D}}_{\mathcal{F}_s}$ is $-N \log N$ when using NLL loss, and 0 for the L2 loss. The optimum value above is achieved if and only if $P_1 = P_2 = \cdots = P_N = \overline{P}$ for either loss.*

Therefore, the loss of $\mathcal{F}_s$ can be interpreted as simultaneously minimizing the classification loss J_C as well as the divergence among feature distributions of all domains. It can thus learn a shared feature mapping that is invariant across domains upon successful training while being beneficial to the main classification task.

	Book	DVD	Elec.	Kit.	Avg.
Domain-Specific Models Only					
LS	77.80	77.88	81.63	84.33	80.41
SVM	78.56	78.66	83.03	84.74	81.25
LR	79.73	80.14	84.54	86.10	82.63
MLP	81.70	81.65	85.45	85.95	83.69
Shared Model Only					
LS	78.40	79.76	84.67	85.73	82.14
SVM	79.16	80.97	85.15	86.06	82.83
LR	80.05	81.88	85.19	86.56	83.42
MLP	82.40	82.15	85.90	88.20	84.66
MAN-L2-MLP	82.05	83.45	86.45	88.85	85.20
MAN-NLL-MLP	81.85	83.10	85.75	89.10	84.95
Shared-Private Models					
RMTL[1]	81.33	82.18	85.49	87.02	84.01
MTLGraph[2]	79.66	81.84	83.69	87.06	83.06
CMSC-LS[3]	82.10	82.40	86.12	87.56	84.55
CMSC-SVM[3]	82.26	83.48	86.76	88.20	85.18
CMSC-LR[3]	81.81	83.73	86.67	88.23	85.11
SP-MLP	82.00	**84.05**	86.85	87.30	85.05
MAN-L2-SP-MLP	82.46 (±0.25)	83.98 (±0.17)	**87.22*** (±0.04)	88.53 (±0.19)	85.55* (±0.07)
MAN-NLL-SP-MLP	**82.98*** (±0.28)	84.03 (±0.16)	87.06 (±0.23)	**88.57*** (±0.15)	**85.66*** (±0.14)

[1] Evgeniou and Pontil (2004) [2] Zhou et al. (2011)
[3] Wu and Huang (2015)

Table 4.1: MDTC results on the Amazon dataset. Models in bold are ours while the performance of the rest are taken from Wu and Huang (2015). Numbers in parentheses indicate standard errors, calculated based on 5 runs. Bold numbers indicate the highest performance in each domain, and $*$ shows statistical significance ($p < 0.05$) over CMSC under a one-sample T-Test.

4.4 Experiments

4.4.1 Multi-Domain Text Classification

In this experiment, we compare `MAN` to state-of-the-art MDTC systems on the multi-domain Amazon review dataset (Blitzer et al., 2007), which is one of the most widely used MDTC datasets. Note that this dataset was already preprocessed into a bag of features (unigrams and bigrams), losing all word order information. This prohibits the use of CNNs or RNNs as feature extractors, limiting the potential performance of the system. Nonetheless, we adopt the same dataset for fair comparison and employ a MLP as our feature extractor. In particular, we take the 5000 most frequent features and represent each review as a 5000d feature vector, where feature values are raw counts of the features. Our MLP feature extractor would then have an input size of 5000 in order to process the reviews.

The Amazon dataset contains 2000 samples for each of the four domains: *book, DVD, electronics*, and *kitchen*, with binary labels (positive, negative). Following Wu and Huang (2015), we conduct 5-way cross validation. Three out of the five folds are treated as the training set, one serves as the validation set, while the remaining is the test set. The 5-fold average test accuracy is reported.

Table 4.1 shows the main results. Three types of models are shown: *Domain-Specific Models Only*, where only in-domain models are trained[5]; *Shared Model Only*, where a single model is trained with all data; and *Shared-Private Models*, a combination of the previous two. Within each category, various architectures

[5]For our models, it means $\mathcal{F}_s$ is disabled. Similarly, for Shared Model Only, no $\mathcal{F}_d$ is used.

are examined, such as Least Square (LS), SVM, and Logistic Regression (LR). As explained before, we use MLP as our feature extractors for all our models (bold ones). Among our models, the ones with the MAN prefix use adversarial training, and MAN-L2 and MAN-NLL indicate MAN with the L2 loss and the NLL loss, respectively.

From Table 4.1, we can see that by adopting modern deep neural networks, our methods achieve superior performance within the first two model categories even without adversarial training. This is corroborated by the fact that our SP-MLP model performs comparably to CMSC, while the latter relies on external resources such as sentiment lexica. Moreover, when our multinomial adversarial nets are introduced, further improvement is observed. With both loss functions, MAN outperforms all Shared-Private baseline systems on each domain, and achieves statistically significantly higher overall performance. For our MAN-SP models, we provide the mean accuracy as well as the standard errors over five runs, to illustrate the performance variance and conduct significance tests. It can be seen that MAN's performance is relatively stable, and consistently outperforms CMSC.

4.4.2 Experiments for Unlabeled Domains

As CMSC requires labeled data for each domain, their experiments were naturally designed this way. In reality, however, many domains may not have any annotated corpora available. It is therefore also important to look at the performance in these unlabeled domains for a MDTC system. Fortunately, as depicted before, MAN's adversarial training only utilizes unlabeled data from each

domain to learn the domain-invariant features, and can thus be used on unlabeled domains as well. During testing, only the shared feature vector is fed into C, while the domain feature vector is set to $\mathbf{0}$.

In order to validate MAN's effectiveness, we compare to state-of-the-art *multi-source domain adaptation* (MS-DA) methods (see Section 4.1). Compared to standard domain adaptation methods with one source and one target domain, MS-DA allows the adaptation from multiple source domains to a single target domain. Analogically, MDTC can be viewed as *multi-source multi-target* domain adaptation, which is superior when multiple target domains exist. With multiple target domains, MS-DA will need to treat each one as an independent task, which is more expensive and cannot utilize the unlabeled data in other target domains.

In this work, we compare MAN with one recent MS-DA method, MDAN (Zhao et al., 2018). Their experiments only have one target domain to suit their approach, and we follow this setting for fair comparison. However, it is worth noting that MAN is designed for the MDTC setting, and can deal with multiple target domains at the same time, which can potentially improve the performance by taking advantage of more unlabeled data from multiple target domains during adversarial training. We adopt the same setting as Zhao et al. (2018), which is based on the same multi-domain Amazon review dataset. Each of the four domains in the dataset is treated as the target domain in four separate experiments, while the remaining three are used as source domains.

In Table 4.2, the target domain is shown on top, and the test set accuracy is reported for various systems. It shows that MAN outperforms several baseline systems, such as a MLP trained on the source-domains, as well as single-source do-

Target Domain	Book	DVD	Elec.	Kit.	Avg.
MLP	76.55	75.88	84.60	85.45	80.46
mSDA[1]	76.98	78.61	81.98	84.26	80.46
DANN[2]	77.89	78.86	84.91	86.39	82.01
MDAN (H-MAX)[3]	78.45	77.97	84.83	85.80	81.76
MDAN (S-MAX)[3]	**78.63**	80.65	**85.34**	86.26	**82.72**
MAN-L2-SP-MLP	78.45	81.57	83.37	85.57	82.24
MAN-NLL-SP-MLP	77.78	**82.74**	83.75	**86.41**	82.67

[1] Chen et al. (2012) [2] Ganin et al. (2016)
[3] Zhao et al. (2018)

Table 4.2: Results on unlabeled domains. Models in bold are our models while the rest are taken from Zhao et al. (2018). Highest domain performance is shown in bold.

main adaptation methods such as mSDA (Chen et al., 2012) and DANN (Ganin et al., 2016), where the training data in the multiple source domains are combined and viewed as a single domain. Finally, when compared to MDAN, MAN and MDAN each achieves higher accuracy on two out of the four target domains, and the average accuracy of MAN is similar to MDAN. Therefore, MAN achieves competitive performance for the domains without annotated corpus. Nevertheless, unlike MS-DA methods, MAN can handle multiple target domains at one time.

4.4.3 Experiments on the MTL Dataset

To make fair comparisons, the previous experiments follow the standard settings in the literature, where the widely adopted Amazon review dataset is used. However, this dataset has a few limitations. First, it has only four domains. In addition, the reviews are already tokenized and converted to a bag of features consisting of unigrams and bigrams. Raw review texts are hence not

	books	elec.	dvd	kitchen	apparel	camera	health	music	toys	video	baby	magaz.	softw.	sports	IMDb	MR	Avg.
Domain-Specific Models Only																	
BiLSTM	81.0	78.5	80.5	81.2	86.0	86.0	78.7	77.2	84.7	83.7	83.5	91.5	85.7	84.0	85.0	74.7	82.6
CNN	85.3	87.8	76.3	84.5	86.3	89.0	87.5	81.5	87.0	82.3	82.5	86.8	87.5	85.3	83.3	75.5	84.3
Shared Model Only																	
FS-MTL	82.5	85.7	83.5	86.0	84.5	86.5	88.0	81.2	84.5	83.7	88.0	92.5	86.2	85.5	82.5	74.7	84.7
MAN-L2-CNN	**88.3**	88.3	87.8	88.5	85.3	90.5	**90.8**	85.3	89.5	89.0	89.5	91.3	88.3	89.5	**88.5**	73.8	87.7
MAN-NLL-CNN	88.0	87.8	87.3	88.5	86.3	90.8	89.8	84.8	89.3	89.3	87.8	91.8	90.0	**90.3**	87.3	73.5	87.6
Shared-Private Models																	
ASP-MTL	84.0	86.8	85.5	86.2	87.0	89.2	88.2	82.5	88.0	84.5	88.2	92.2	87.2	85.7	85.5	**76.7**	86.1
MAN-L2-SP-CNN	87.6* (0.2)	87.4 (1.0)	88.1* (0.4)	89.8* (0.4)	**87.6** (0.7)	**91.4*** (0.4)	89.8* (0.3)	**85.9*** (0.1)	90.0* (0.1)	89.5* (0.2)	90.0 (0.6)	92.5 (0.5)	90.4* (0.4)	89.0* (0.4)	86.6 (0.5)	76.1 (0.5)	88.2* (0.1)
MAN-NLL-SP-CNN	86.8* (0.4)	**88.8** (0.6)	**88.6*** (0.4)	**89.9*** (0.4)	**87.6** (0.4)	90.7 (0.4)	89.4 (0.3)	85.5* (0.1)	**90.4*** (0.2)	**89.6*** (0.3)	**90.2** (0.6)	**92.9** (0.4)	**90.9*** (0.7)	89.0* (0.2)	87.0* (0.1)	**76.7** (0.8)	**88.4*** (0.1)

Table 4.3: Results on the FDU-MTL dataset. Bolded models are ours, while the rest are taken from Liu et al. (2017). Highest performance is each domain is highlighted. For our full MAN models, standard errors are shown in parentheses and statistical significance ($p < 0.01$) over ASP-MTL is indicated by *.

available in this dataset, making it impossible to use certain modern neural architectures such as CNNs and RNNs. To provide more insights on how well MAN works with other feature extractor architectures, we provide a third set of experiments on the FDU-MTL dataset (Liu et al., 2017). This dataset is created as a multi-task learning dataset with 16 *tasks*, where each task is essentially a different domain of reviews. It has 14 Amazon domains: books, electronics, DVD, kitchen, apparel, camera, health, music, toys, video, baby, magazine, software, and sports, in addition to two movie review domains from the IMDb and the MR datasets. Each domain has a development set of 200 samples, and a test set of 400 samples. The amount of training and unlabeled data vary across domains but are roughly 1400 and 2000, respectively.

We compare MAN with ASP-MTL (Liu et al., 2017) on this FDU-MTL dataset. ASP-MTL also adopts adversarial training for learning a shared feature space, and can be viewed as a special case of MAN that adopts the NLL loss (MAN-NLL) and chooses LSTM as their feature extractor. In contrast, we found a CNN-based feature extractor (Kim, 2014) achieves much better accuracy while being $\sim$ 10 times faster. Indeed, as shown in Table 4.3, with or without adversarial training, our CNN models outperform LSTM ones by a large margin. When used in our MAN framework, we attain the state-of-the-art performance on every domain with a 88.4% overall accuracy, surpassing ASP-MTL by a significant margin of 2.3%.

We hypothesize the reason a LSTM performs much worse than a CNN is its lack of an attention mechanism. In ASP-MTL, only the last hidden unit is taken as the extracted features. While LSTMs are effective for representing the context for each token, it might not be powerful enough for directly encoding the en-

tire document (Bahdanau et al., 2015). Therefore, various attention mechanisms have been introduced on top of the vanilla LSTM to select words (and contexts) most relevant for making the predictions. In our preliminary experiments, we find that a Bi-directional LSTM with the dot-product attention (Luong et al., 2015) yields better performance than the vanilla LSTM in ASP-MTL. However, it still does not outperform CNN and is much slower. As a result, we conclude that, for text classification tasks, CNN is both effective and efficient in extracting local and higher-level features for making a single categorization.

Finally, we observe that MAN-NLL achieves slightly higher overall performance compared to MAN-L2, providing evidence for the claim in a recent study (Lucic et al., 2018) that the original GAN loss (NLL) may not be inherently inferior to the L2 loss. Moreover, the two variants excel in different domains, suggesting the possibility of further performance gain when using ensembled models.

4.4.4 Implementation Details

For all three of our experiments, we use $\lambda = 0.05$ and $k = 5$ (See Algorithm 4.1). For both optimizers, Adam (Kingma and Ba, 2015) is used with learning rate 0.0001. The size of the shared feature vector is set to 128 while that of the domain feature vector is 64. Dropout of $p = 0.4$ is used in all components. C and $\mathcal{D}$ each has one hidden layer of the same size as their input (128 + 64 for C and 128 for $\mathcal{D}$). ReLU is used as the activation function. Batch normalization (Ioffe and Szegedy, 2015) is used in both C and $\mathcal{D}$ but not $\mathcal{F}$. We use a batch size of 8.

For our first two experiments on the Amazon review dataset, the MLP fea-

ture extractor is used. As described in Section 4.4.1, it has an input size of 5000. Two hidden layers are used, with size 1000 and 500, respectively.

For the CNN feature extractor used in the FDU-MTL experiment, a single convolution layer is used. The kernel sizes are 3, 4, and 5, and the number of kernels are 200. The convolution layers take as input the 100d word embeddings of each word in the input sequence. We use *word2vec* word embeddings (Mikolov et al., 2013a) trained on a bunch of unlabeled raw Amazon reviews (Blitzer et al., 2007). After convolution, the outputs go through a ReLU layer before fed into a max pooling layer. The pooled output is then fed into a single fully connected layer to be converted into a feature vector of size either 128 or 64. More details of using CNN for text classification can be found in the original paper (Kim, 2014). `MAN` is implemented using PyTorch (Paszke et al., 2017)[6].

4.5 Chapter Summary

In this chapter, we propose a family of Multinomial Adversarial Networks (`MAN`s) that generalize the traditional binomial adversarial nets (as seen in Chapter 3) in the sense that `MAN` can simultaneously minimize the difference among multiple probability distributions instead of just two. We provide theoretical justifications for two instances of `MAN`, `MAN`-NLL and `MAN`-L2, showing they are minimizers of two different f-divergence metrics among multiple distributions, respectively. This indicates `MAN` can be used to make multiple distributions indistinguishable from one another. It can hence be applied to a variety of tasks, similar to the versatile binomial adversarial nets, which have been used in many

[6]The source code of `MAN` is available at `https://github.com/ccsasuke/man`.

areas for making *two* distributions alike.

In particular, we further devise a MAN model for the multi-domain task classification (MDTC) task, following the shared-private paradigm that has a shared feature extractor to learn domain-invariant features and domain feature extractors to learn domain-specific ones. MAN is used to enforce the shared feature extractor to learn only domain-invariant knowledge, by resorting to MAN's power of making indistinguishable the shared feature distributions of samples from each domain. We conduct extensive experiments, demonstrating our MAN model outperforms the prior art systems in MDTC, and achieves state-of-the-art performance on domains without labeled data when compared to multi-source domain adaptation methods.

CHAPTER 5

MULTILINGUAL MODEL TRANSFER: LEARNING WHAT TO SHARE

In Chapter 4, we proposed multinomial adversarial networks (MANs) that could serve as a tool to minimize divergence among multiple probability distributions. In the context of cross-lingual transfer learning (CLTL), it can be viewed as a generalization of language-adversarial training (Chapter 3) for tackling the multilingual transfer learning (MLTL) setting.

As mentioned in Section 2.1.3, the MLTL setting, also known as the **multi-source CLTL** setting, tackles the case where labeled training data (type 0 supervision) is available in multiple source languages, and we would like to be able to utilize all of them when transferring to other low-resource target languages. Previous work (McDonald et al., 2011) indeed showed that transferring from multiple source languages could result in significant performance improvement compared to transferring from a single source language as in the bilingual transfer (BLTL) case.

As shown in Section 4.4.2, MAN can be applied to this multi-source transfer learning setting where we have multiple "labeled" source languages and the goal is to improve the performance of an "unlabeled" target language[1]. In the MLTL setting, however, where multiple source languages exist, MAN will only use, for model transfer, the features that are common among all source languages and the target, which may be too restrictive in many cases. For example, when transferring from English, Spanish and Chinese to German, MAN will retain only features that are invariant across all four languages, which can be too

[1]Again, although we were focusing on "domains" rather than "languages" in Chapter 4, MAN can be readily applied to CLTL when coupled with cross-lingual word embeddings (See experiments in Section 5.3).

sparse to be informative. Furthermore, the fact that German is more similar to English than to Chinese is neglected because the transferred model is unable to utilize features that are shared only between English and German.

To address these shortcomings, we propose a new MLTL model that not only exploits language-invariant features, but also allows the target language to dynamically and selectively leverage language-specific features through a probabilistic attention-style mixture of experts mechanism (see Section 5.2). This allows our model to learn effectively what to share between various languages. Another contribution is that, when combined with the recent unsupervised cross-lingual word embeddings (Lample et al., 2018; Chen and Cardie, 2018b), our model is able to operate in a *zero-resource* setting where neither *task-specific target language annotations* (type I supervision) nor *general-purpose cross-lingual resources* (type II supervision) are available. This is an advantage over many existing CLTL works, making our model more widely applicable to many lower-resource languages.

We evaluate our model on multiple MLTL tasks ranging from text classification to named entity recognition and semantic slot filling, including a real-world industry dataset. Our model beats all baseline models trained, like ours, without type II supervision. More strikingly, in many cases, it can match or even outperform state-of-the-art models that have access to strong type II supervision (e.g. commercial Machine Translation systems).

This chapter is based on Chen et al. (2018a).

5.1 Related Work

The diversity of human languages is a critical challenge for natural language processing. In order to alleviate the need for obtaining annotated data for each task in each language, cross-lingual transfer learning (CLTL) has long been studied (Yarowsky et al., 2001; Bel et al., 2003, *inter alia*).

For *unsupervised* CLTL in particular, where no target language training data is available, most prior research investigates the **bilingual transfer** setting. Traditionally, research focuses on *resource-based* methods, where general-purpose cross-lingual resources such as MT systems or parallel corpora are utilized to replace task-specific annotated data (Wan, 2009; Prettenhofer and Stein, 2010). With the advent of deep learning, especially adversarial neural networks (Goodfellow et al., 2014; Ganin et al., 2016), progress has been made towards *model-based* CLTL methods. We (Chen et al., 2016, Chapter 3) proposed language-adversarial training that did not directly depend on parallel corpora, but instead only required a set of bilingual word embeddings (BWEs).

On the other hand, the **multilingual transfer** setting, although less explored, has also been studied (McDonald et al., 2011; Hajmohammadi et al., 2014; Guo et al., 2016), showing improved performance compared to using labeled data from one source language as in bilingual transfer.

Another important direction for CLTL is to learn cross-lingual word representations (Klementiev et al., 2012; Zou et al., 2013; Mikolov et al., 2013b). Recently, there have been several notable work for learning fully unsupervised

cross-lingual word embeddings, both for the bilingual (Zhang et al., 2017; Lample et al., 2018; Artetxe et al., 2018) and multilingual case (Chen and Cardie, 2018b). These efforts pave the road for performing CLTL without cross-lingual resources.

Finally, a related field to MLTL is multi-source domain adaptation (Zhao et al., 2018), where most prior work relies on the learning of domain-invariant features (Zhao et al., 2018; Chen and Cardie, 2018a). A very recent work (Guo et al., 2018) attempts to model the relation between the target domain and each source domain. Our model combines the strengths of prior art and is able to simultaneously utilize both the domain-invariant and domain-specific features in a coherent way.

5.2 The MAN-MoE Model

As we have seen in Chapter 4, one commonly adopted paradigm for neural cross-lingual transfer is the *shared-private* model (Bousmalis et al., 2016), where the features are divided into two parts: *shared* (language-invariant) features and *private* (language-specific) features. The shared features are enforced to be language-invariant via language-adversarial training, by attempting to fool a language discriminator. Furthermore, we in Chapter 4 propose a generalized shared-private model for the multi-source setting, where a *multinomial adversarial network* (MAN) is adopted to extract common features shared by all source languages as well as the target. On the other hand, the private features are learned by separate feature extractors, one for each source language, capturing the remaining features outside the shared ones. During training, the labeled

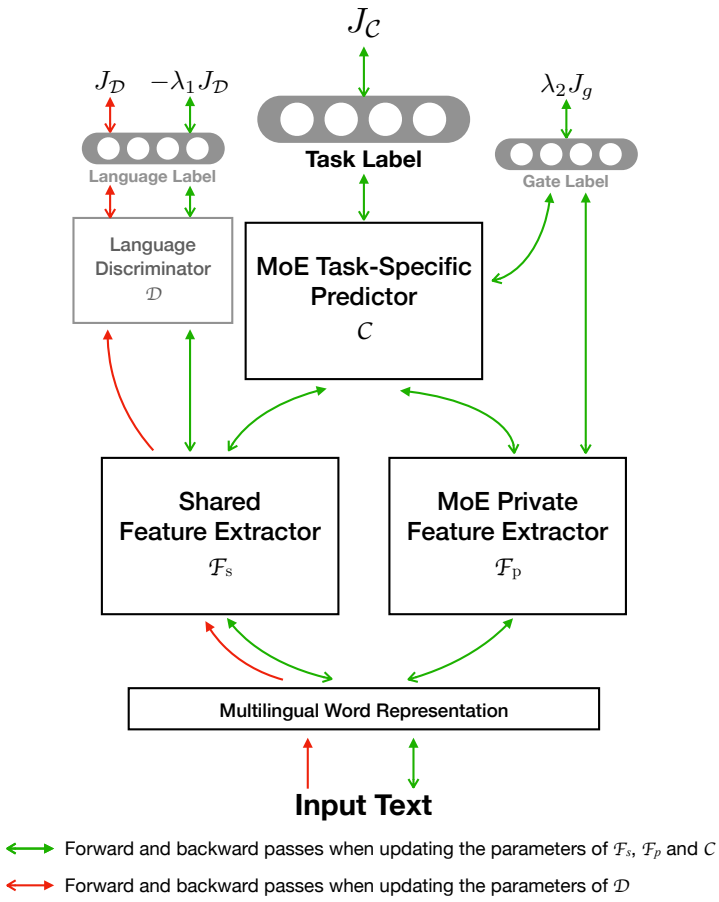

Figure 5.1: An overview of the MAN-MoE model.

samples from a certain source language go through the corresponding private feature extractor for that particular language. At test time, there is no private feature extractor for the target language; only the shared features are used for cross-lingual transfer.

As mentioned in the beginning of this chapter, using only the shared features for MLTL as in the MAN model imposes an overly strong constraint and many useful features may be wiped out by adversarial training if they are shared only between the target language and a subset of source languages. Therefore, we propose to use a mixture-of-experts (MoE) model (Shazeer et al., 2017; Gu et al.,

2018) to learn the private features. The idea is to have a set of language expert networks, one per source language, each responsible for learning language-specific features for that source language during training. However, instead of hard-switching between the experts, each sample uses a convex combination of all experts, dictated by an *expert gate*. Thus, at test time, the trained expert gate can decide the optimal expert weights for the unseen target language based on its similarity to the source languages.

Figure 5.1 shows an overview of our full MAN-MoE model for multilingual model transfer. The boxes illustrate various components of the MAN-MoE model (Section 5.2.1), while the arrows depict the training flow (Section 5.2.2).

5.2.1 Model Architecture

Figure 5.1 portrays an abstract view of the MAN-MoE model with four major components: the Multilingual Word Representation, the Shared Feature Extractor $\mathcal{F}_s$ (together with the Language Discriminator $\mathcal{D}$), the MoE Private Feature Extractor $\mathcal{F}_p$, and finally the MoE Predictor $\mathcal{C}$. Based on the actual task (e.g. sequence tagging, text classification, sequence to sequence, etc.), different architectures may be adopted, as explained below.

Multilingual Word Representation embeds words from all languages into a single semantic space so that words with similar meanings are close to each other regardless of language. In this work, we mainly rely on the MUSE embeddings (Lample et al., 2018), which are trained in a fully unsupervised manner. We map all other languages into English to obtain a multilingual embedding space. However, in certain experiments, MUSE yields 0 accuracy on one

or more language pairs (Søgaard et al., 2018), in which case the VecMap embeddings (Artetxe et al., 2017) are used. It uses *identical strings* as supervision, which does not require parallel corpus or human annotations. We further experiment with our recent unsupervised multilingual word embeddings (Chen and Cardie, 2018b, Chapter 6), which gives improved performance (Section 5.3.2).

In addition, for tasks where morphological features are important, one can add character-level word embeddings (Dos Santos and Zadrozny, 2014) that captures sub-word information. When character embeddings are used, we add a single CharCNN that is shared across all languages, and the final word representation is the concatenation of the word embedding and the char-level embedding. The CharCNN can then be trained end to end with the rest of the model.

`MAN` Shared Feature Extractor $\mathcal{F}_s$ is a multinomial adversarial network, which is an adversarial pair of a feature extractor (e.g. LSTM or CNN) and a *language discriminator* $\mathcal{D}$. $\mathcal{D}$ is a text classifier (Kim, 2014) that takes the shared features (extracted by $\mathcal{F}_s$) of an input sequence and predicts which language it comes from. On the other hand, $\mathcal{F}_s$ strives to fool $\mathcal{D}$ so that it cannot identify the language of a sample. The hypothesis is that if $\mathcal{D}$ cannot recognize the language of the input, the shared features then do not contain language information and are hence language-invariant. Note that $\mathcal{D}$ is trained only using unlabeled texts, and can therefore be trained on all languages including the target language.

`MoE` Private Feature Extractor $\mathcal{F}_p$ is a key difference from Chapter 4, shown in Figure 5.2. The figure shows the Mixture-of-Experts (Shazeer et al., 2017) model with three source languages, English, German and Spanish. $\mathcal{F}_p$ has a shared BiLSTM at the bottom that extracts contextualized word representations

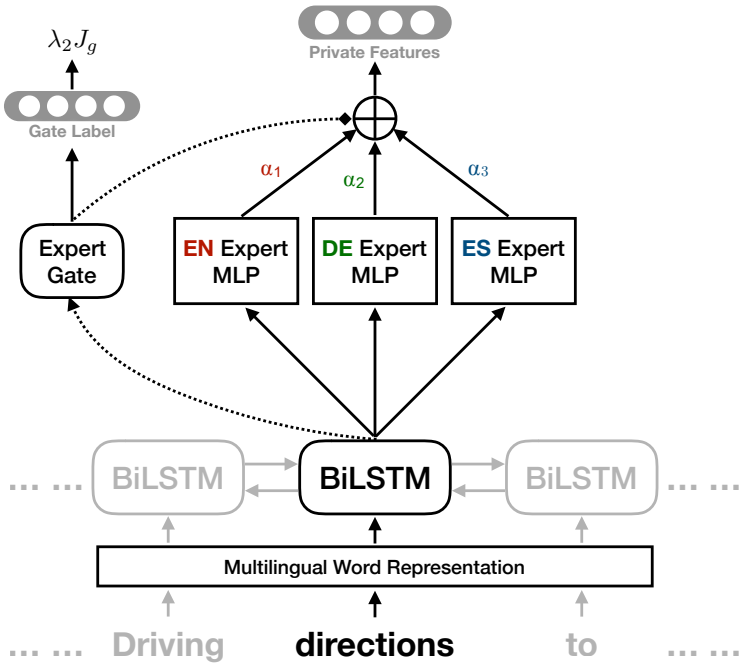

Figure 5.2: The MoE Private Feature Extractor $\mathcal{F}_p$.

for each token w in the input sentence. The LSTM hidden representation $\boldsymbol{h}_w$ is then fed into the MoE module, where each source language has a separate expert network (a MLP). In addition, the *expert gate* $\mathcal{G}$ is a linear transformation that takes $\boldsymbol{h}_w$ as input and outputs a softmax score α_i for each expert. The final private feature vector is a mixture of all expert outputs, dictated by the expert gate weights α.

During training, the expert gate is trained to predict the language of a sample using the gate loss J_g, where the expert gate output α is treated as the softmax probability of the predicted languages. In other words, the more accurate the language prediction is, the more the correct expert gets used. Therefore, J_g is used to encourage samples from a certain source language to use the correct expert (see Section 5.2.2 for more details), and each expert is hence learning language-specific features for that language. At test time, the trained expert gate

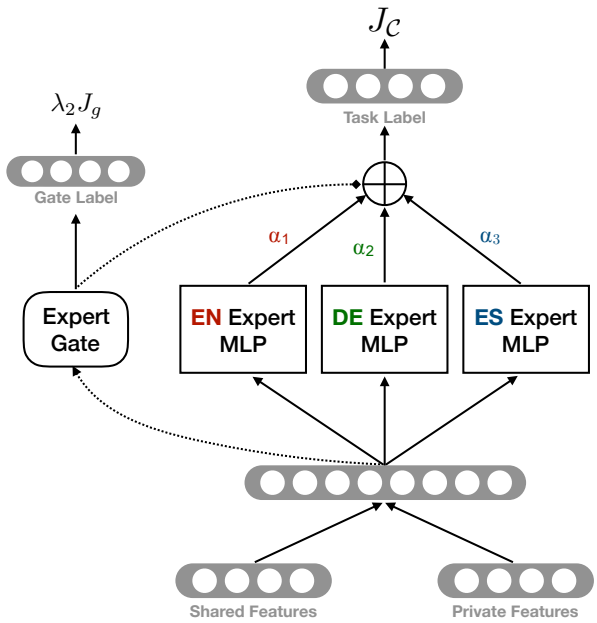

Figure 5.3: The MoE Predictor C for Sequence Tagging.

will examine the hidden representation of a token, and predicts the optimal expert weights α. For instance, if a sample is similar to the EN training samples, the trained gate will predict a higher α for the EN expert, resulting in a heavier use of it in the final feature vector. Therefore, even for the unseen target language, $\mathcal{F}_p$ is able to dynamically determine what knowledge to use from each individual source language at a token level.

MoE Task-Specific Predictor C is the final module that make predictions for the end task, and may take different forms depending on the task. For instance, Figure 5.3 shows the MoE predictor for sequence tagging, where one output label is predicted for each input token. The shared and private features are first concatenated for each token, and then past through a MoE module similar to $\mathcal{F}_p$. It is straightforward to adapt C to work for other tasks. For example, for text classification, a pooling layer such as dot-product attention (Luong et al., 2015) is added at the bottom to fuse token-level features into a single sentence feature

vector.

C first concatenates the shared and private features to form a single feature vector for each token. It then has another MoE module that outputs a softmax probability over all labels for each token. The idea is that it may be favorable to put different weights between the language-invariant and language-specific features for different target languages. Again consider the example of English, German, Spanish and Chinese. When transferring to Chinese from the other three, the source languages are similar to each other while all being rather distant from Chinese. Therefore, the adversarially learned shared features might be more important in this case. On the other hand, when transferring to German, which is much more similar to English than to Chinese, we might want to pay more attention to the MoE private features. Therefore, we adopt a MoE module in C, which provides more flexibility than using a single MLP[2].

5.2.2 Model Training

Denote the set of all N source languages as $\mathcal{S}$, where $|\mathcal{S}| = N$. Denote the target language as $\mathcal{T}$, and let $\Delta = \mathcal{S} \cup \mathcal{T}$ be the set of all languages. Denote the annotated corpus for a source language $l \in \mathcal{S}$ as $\mathbb{X}_l$, where $(x, y) \sim \mathbb{X}_l$ is a sample drawn from $\mathbb{X}_l$. In addition, unlabeled data is required for all languages to facilitate the MAN training. We hence denote as $\mathbb{U}_{l'}$ the unlabeled texts from a language $l' \in \Delta$.

The overall training flow of various components is illustrated in Figure 5.1, while the training algorithm is depicted in Algorithm 5.1. Similar to MAN, there

[2]We also experimented with an attention mechanism between the shared and private features, or a gating mechanism to modulate each feature channel, but got sub-optimal results.

Require: labeled corpus $\mathbb{X}$; unlabeled corpus $\mathbb{U}$; Hyperpamameter $\lambda_1, \lambda_2 > 0$, $k \in \mathbb{N}$

repeat
  ▷ $\mathcal{D}$ iterations
  for *diter* = 1 to k **do**
    $l_{\mathcal{D}} = 0$
    for all $l \in \Delta$ **do** ▷ For all languages
      Sample a mini-batch $\boldsymbol{x} \sim \mathbb{U}_l$
      $\boldsymbol{f}_s = \mathcal{F}_s(\boldsymbol{x})$ ▷ Shared features
      $l_{\mathcal{D}}$ += $L_{\mathcal{D}}(\mathcal{D}(\boldsymbol{f}_s); l)$ ▷ $\mathcal{D}$ loss
    Update $\mathcal{D}$ parameters using $\nabla l_{\mathcal{D}}$
  ▷ Main iteration
  $loss = 0$
  for all $l \in \mathcal{S}$ **do** ▷ For all source languages
    Sample a mini-batch $(\boldsymbol{x}, \boldsymbol{y}) \sim \mathbb{X}_l$
    $\boldsymbol{f}_s = \mathcal{F}_s(\boldsymbol{x})$ ▷ Shared features
    $\boldsymbol{f}_p, \boldsymbol{g}_1 = \mathcal{F}_p(\boldsymbol{x})$ ▷ Private features and gate outputs
    $\hat{\boldsymbol{y}}, \boldsymbol{g}_2 = C(\boldsymbol{f}_s, \boldsymbol{f}_p)$
    $loss$ += $L_C(\hat{\boldsymbol{y}}; \boldsymbol{y}) + \lambda_2(L_g(\boldsymbol{g}_1; l) + L_g(\boldsymbol{g}_2; l))$ ▷ C loss and gate loss
  for all $l \in \Delta$ **do** ▷ For all languages
    Sample a mini-batch $\boldsymbol{x} \sim \mathbb{U}_l$
    $\boldsymbol{f}_s = \mathcal{F}_s(\boldsymbol{x})$ ▷ Shared features
    $loss$ += $-\lambda_1 \cdot L_{\mathcal{D}}(\mathcal{D}(\boldsymbol{f}_s); l)$ ▷ Language loss to confuse $\mathcal{D}$
  Update $\mathcal{F}_s, \mathcal{F}_p, C$ parameters using $\nabla loss$
until convergence

Algorithm 5.1: `MAN-MoE` Training

are two separate optimizers to train `MAN-MoE`, one updating the parameters of $\mathcal{D}$ (red arrows), while the other updating the parameters of all other modules (green arrows). In Algorithm 5.1, L_C, $L_{\mathcal{D}}$ and L_g are the loss functions for the predictor C, the language discriminator $\mathcal{D}$, and the expert gates in $\mathcal{F}_p$ and C, respectively.

In practice, we adopt the NLL loss for L_C for text classification, and token-level NLL loss for sequence tagging:

$$L^{NLL}(\hat{y}; y) = -\log P(\hat{y} = y) \tag{5.1}$$

$$L^{T\text{-}NLL}(\hat{\boldsymbol{y}}; \boldsymbol{y}) = -\log P(\hat{\boldsymbol{y}} = \boldsymbol{y}) = -\sum_i \log P(\hat{y}_i = y_i) \tag{5.2}$$

where y is a scalar class label, and $\boldsymbol{y}$ is a vector of token labels. L_C is hence interpreted as the negative log-likelihood of predicting the correct task label. Similarly, $\mathcal{D}$ adopts the NLL loss in (5.1) for predicting the correct language of a sample. Finally, the expert gates $\mathcal{G}$ use token-level NLL loss in (5.2), which translates to the negative log-likelihood of using the correct language expert for each token in a sample.

Therefore, the objectives that C, $\mathcal{D}$ and $\mathcal{G}$ minimize are, respectively:

$$J_C = \sum_{l \in \mathcal{S}} \mathop{\mathbb{E}}_{(x,y) \in \mathbb{X}_l} \Big[L_C(C(\mathcal{F}_s(x), \mathcal{F}_p(x)); y) \Big] \tag{5.3}$$

$$J_{\mathcal{D}} = \sum_{l \in \Delta} \mathop{\mathbb{E}}_{x \in \mathbb{U}_l} [L_{\mathcal{D}}(\mathcal{D}(\mathcal{F}_s(x)); l)] \tag{5.4}$$

$$J_{\mathcal{G}} = \sum_{l \in \mathcal{S}} \mathop{\mathbb{E}}_{x \in \mathbb{X}_l} \left[\sum_{w \in x} L_{\mathcal{G}}(\mathcal{G}(h_w); l) \right] \tag{5.5}$$

where $\boldsymbol{h}_w$ in (5.5) is the BiLSTM hidden representation in $\mathcal{F}_p$ as shown in Figure 5.2. In addition, note that $\mathcal{D}$ is trained using unlabeled corpora over all languages (Δ), while the training of $\mathcal{F}_p$ and C (and hence $\mathcal{G}$) only take place on source languages ($\mathcal{S}$). Finally, the overall objective function is:

$$J = J_C - \lambda_1 J_{\mathcal{D}} + \lambda_2 (J_{\mathcal{G}}^{(1)} + J_{\mathcal{G}}^{(2)}) \tag{5.6}$$

where $J_{\mathcal{G}}^{(1)}$ and $J_{\mathcal{G}}^{(2)}$ are the two expert gates in $\mathcal{F}_p$ and C, respectively.

5.3 Experiments and Discussions

In this section, we present an extensive set of experiments across three datasets. The first experiment is on a real-world multilingual slot filling (sequence tagging) dataset, where the data is used in a commercial personal virtual assistant. In addition, we conduct experiments on two public academic datasets,

	English			German			Spanish			Chinese			
Domain	#Train	#Dev	#Test	#Train	#Dev	#Test	#Train	#Dev	#Test	#Train	#Dev	#Test	#Slot
Navigation	311045	23480	36625	13356	1599	2014	13862	1497	1986	7472	1114	1173	8
Calendar	64010	5946	8260	8261	1084	1366	6706	926	1081	2056	309	390	4
Files	30339	2058	5355	3005	451	480	6082	843	970	1289	256	215	5

Domain	Examples
Navigation	[*Driving*]$_{transportation\_type}$ directions to [*Walmart*]$_{place\_name}$ in [*New York*]$_{location}$.
Calendar	Add [*school meeting*]$_{title}$ to my calendar on [*Monday*]$_{start\_date}$ at [*noon*]$_{start\_time}$.
Files	Search for [*notes*]$_{data\_type}$ with [*grocery list*]$_{keyword}$.

Table 5.1: Statistics for the Multilingual Semantic Slot Filling dataset with examples from each domain.

namely the CoNLL multilingual named entity recognition (sequence tagging) dataset (Sang, 2002; Sang and Meulder, 2003), and the Multilingual Amazon Reviews (text classification) dataset (Prettenhofer and Stein, 2010).

5.3.1 Cross-Lingual Semantic Slot Filling

As shown in Table 5.1, we collect data for four languages: English, German, Spanish, and Chinese, over three domains: Navigation, Calendar, and Files. Each domain has a set of pre-determined slots (the slots are the same across languages), and the user utterances in each language and domain are annotated by crowd workers with the correct slots (see the examples in Table 5.1). We employ the standard BIO tagging scheme to formulate the slot filling problem as a sequence tagging task.

For each domain and language, the data is divided into a training, a validation, and a test set, with the number of samples in each split shown in Table 5.1. In our experiments, we treat each domain as a separate experiment, and consider each of German, Spanish and Chinese as the target language while the remaining three being source languages, which results in a total of 9 experiments.

Results

In Table 5.2, we report the performance of `MAN-MoE` compared to a number of baseline systems. All systems adopt the same base architecture, which is a multi-layer BiLSTM sequence tagger (İrsoy and Cardie, 2014) with a token-level

	German				Spanish				Chinese			
Domain	Navi.	Cal.	Files	*avg.*	Navi.	Cal.	Files	*avg.*	Navi.	Cal.	Files	*avg.*
Methods with cross-lingual resources												
MT (train-on-trans.)	59.95	63.53	38.68	54.05	64.37	**59.93**	67.55	**63.95**	**60.56**	**66.49**	61.01	**62.69**
MT (test-on-trans.)	54.49	51.74	55.87	54.03	52.13	58.10	55.00	55.08	54.23	22.71	**64.01**	46.98
Methods without cross-lingual resources												
BWE (1-to-1)	57.53	58.28	35.73	50.51	62.54	44.44	57.56	54.85	17.62	22.48	21.32	20.47
BWE (3-to-1)	61.03	67.66	51.30	60.00	63.74	45.10	64.47	57.77	20.91	13.70	28.47	21.03
MAN	59.07	60.24	39.35	52.89	58.86	37.90	46.75	47.84	34.45	13.53	40.63	29.54
MAN-MoE	**62.73**	**75.13**	**59.19**	**65.68**	**66.57**	50.21	**70.91**	62.56	34.18	29.36	41.70	35.08

Table 5.2: F1 scores on the Multilingual Semantic Slot Filling dataset. The highest performance is in bold, while the highest performance within method group (with vs. without cross-lingual supervision) is underlined.

	German				Spanish				Chinese			
Domain	Navi.	Cal.	Files	*avg*	Navi.	Cal.	Files	*avg*	Navi.	Cal.	Files	*avg*
MAN-MoE	62.73	75.13	**59.19**	**65.68**	**66.57**	**50.21**	**70.91**	**62.56**	34.18	**29.36**	41.70	**35.08**
- C MoE	**63.42**	**76.68**	55.68	65.26	65.50	47.51	69.67	60.89	27.71	21.75	**41.77**	30.41
- $\mathcal{F}_p$ MoE	58.33	48.85	37.35	48.18	58.99	36.67	48.39	48.02	**39.61**	14.64	38.08	30.78
- MoE	59.07	60.24	39.35	52.89	58.86	37.90	46.75	47.84	34.45	13.53	40.63	29.54
- MAN	60.64	67.69	55.10	61.14	65.38	46.71	68.25	60.11	18.43	10.82	28.90	19.38

Table 5.3: Ablation results on the Multilingual Semantic Slot Filling dataset.

MLP on top (no CRFs were used).

MT baselines employ machine translation (MT) for cross-lingual transfer. In particular, the *train-on-trans(lation)* method translates the entire English training set into each target language which are in turn used to train a supervised system on the target language. On the other hand, the *test-on-trans(lation)* method trains an English sequence tagger, and utilizes MT to translate the test set of each target language into English in order to make predictions. In this work, we adopt the Microsoft Translator[3], a strong commercial MT system. Note that for a MT system to work for sequence tagging tasks, *word alignment* information must be available, in order to project word-level annotations across languages. This rules out many MT systems such as Google Translate since they do not provide word alignment information through their APIs.

BWE baselines rely on Bilingual Word Embeddings (BWEs) and weight sharing for CLTL. Namely, the sequence tagger trained on the source language(s) are directly applied to the target language, in hopes that the BWEs could bridge the language gap. This simple method has been shown to yield strong results in recent work (Upadhyay et al., 2018). The MUSE (Lample et al., 2018) BWEs are used by all systems in this experiment. *1-to-1* indicates that we are only transferring from English, while *3-to-1* means the training data from all other three languages are leveraged.[4]

The final baseline is the `MAN` model, presented before our `MAN-MoE` approach. As shown in Table 5.2, `MAN-MoE` substantially outperforms all baseline systems that do not employ cross-lingual supervision on almost all domains and languages. Another interesting observation is that `MAN` performs strongly

[3]https://azure.microsoft.com/en-us/services/cognitive-services/translator-text-api/
[4]`MAN` and `MAN-MoE` results are always 3-to-1.

on Chinese while being much worse on German and Spanish compared to the BWE baseline. This corroborates our hypothesis that MAN only leverages features that are invariant across *all* languages for CLTL, and it learns such features better than weight sharing. Therefore, when transferring to German or Spanish, which is similar to a subset of source languages, the performance of MAN degrades significantly. On the other hand, when Chinese serves as the target language, where all source languages are rather distant from it, MAN has its merit in extracting language-invariant features that could generalize to Chinese. With MAN-MoE, however, this trade-off between close and distant language pairs is well addressed by the combination of MAN and MoE. By utilizing both language-invariant and language-specific features for transfer, MAN-MoE outperforms all cross-lingually unsupervised baselines on all languages.

Furthermore, even when compared with the MT baseline, which has access to hundreds of millions of parallel sentences, MAN-MoE performs competitively on German and Spanish. It even significantly beats both MT systems on German as MT sometimes fails to provide accurate word alignment for German. On Chinese, where the unsupervised BWEs are much less accurate (BWE baselines only achieve 20% F1), MAN-MoE is able to greatly improve over the BWE and MAN baselines and shows promising results for zero-resource CLTL even between distant language pairs.

Feature Ablation

In this section, we take a closer look at the various modules of MAN-MoE and their impacts on performance (Table 5.3). When the MoE in C is removed, moderate decrease is observed on all languages. The performance degrades the most

on Chinese, suggesting that using a single MLP in C is not ideal when the target language is not similar to the sources. When removing the private MoE, the MoE in C no longer makes much sense as C only has access to the shared features, and the performance is even slightly worse than removing both MoEs. With both MoE modules removed, it reduces to the MAN model, and we see a significant drop on German and Spanish. Finally, when removing MAN while keeping MoE, where the shared features are simply learned via weight-sharing, we see a slight drop on German and Spanish, but a rather great one on Chinese. The ablation results support our hypotheses and validate the merit of MAN-MoE.

5.3.2 Cross-Lingual Named Entity Recognition

In this section, we present experiments on the CoNLL 2002/2003 multilingual named entity recognition (NER) dataset (Sang, 2002; Sang and Meulder, 2003), with four languages: English, German, Spanish and Dutch. The task is also formulated as a sequence tagging problem, with four types of tags: PER, LOC, ORG, and MISC.

The results are summarized in Table 5.4. We observe that using only word embeddings does not yield satisfactory results, since the out-of-vocabulary problem is rather severe, and morphological features such as capitalization is crucial for NER. We hence add character-level word embeddings for this task (Section 5.2.1) to capture subword features and alleviate the OOV problem. For German, however, all nouns are capitalized, and the capitalization features learned on the other three languages would lead to poor results. Therefore, for German only, we lowercase all characters in systems that adopt CharCNN.

Target Language	de	es	nl	avg
Methods with cross-lingual resources				
Täckström et al. (2012)	40.4	59.3	58.4	52.7
Nothman et al. (2013)	55.8	61.0	64.0	60.3
Tsai et al. (2016)	48.1	60.6	61.6	56.8
Ni et al. (2017)	**58.5**	65.1	65.4	63.0
Mayhew et al. (2017)	57.5	66.0	64.5	62.3
Methods without cross-lingual resources				
MAN-MoE	55.1	59.5	61.8	58.8
BWE+CharCNN (1-to-1)	51.5	61.0	67.3	60.0
BWE+CharCNN (3-to-1)	55.8	70.4	69.8	65.3
Xie et al. (2018)*	56.9	71.0	71.3	66.4
MAN-MoE+CharCNN	56.7	71.0	70.9	66.2
MAN-MoE+CharCNN+UMWE	56.0	**73.5**	**72.4**	**67.3**

* Contemporaneous work

Table 5.4: F1 scores for the CoNLL NER dataset on German (de), Spanish (es) and Dutch (nl).

Table 5.4 also shows the performance of several state-of-the-art models in the literature[5]. Note that most of these systems are specifically designed for the NER task, and exploit many task-specific resources, such as multilingual gazetteers and metadata in Freebase or Wikipedia (such as entity categories). Among these, Täckström et al. (2012) rely on parallel corpora to learn cross-lingual word clusters that serve as features. Nothman et al. (2013); Tsai et al. (2016) both leverage information in external knowledge bases such as Wikipedia to learn useful features for cross-lingual NER. Ni et al. (2017) employ noisy parallel corpora (aligned sentence pairs, but not always translations) and bilingual dictionaries (5k words for each language pair) for model transfer. They further add external features such as entity types learned from Wikipedia for improved performance. Finally, Mayhew et al. (2017) propose a multi-source framework that utilizes large cross-lingual lexica. Despite using none of these resources,

[5]We also experimented with the MT baselines, but it often failed to produce word alignment, resulting in many empty predictions. The MT baselines attain only a F1 score of ~30%, and were thus excluded for comparison.

general or task-specific, MAN-MoE nonetheless outperforms all these methods. The only exception is German, where task-specific resources remain helpful due to its unique capitalization rules and high OOV rate.

In a contemporaneous work by (Xie et al., 2018), they propose a cross-lingual NER model using Bi-LSTM-CRF that achieves similar performance compared to MAN-MoE+CharCNN. However, our architecture is not specialized to the NER task, and we did not add task-specific modules such as a CRF decoding layer, etc.

Last but not least, we replace the MUSE embeddings with our recently proposed unsupervised multilingual word embeddings (Chen and Cardie, 2018b, Chapter 6), which further boosts the performance, achieving a new state-of-the-art performance.

5.3.3 Cross-Lingual Text Classification on Amazon Reviews

Finally, we report results on a multilingual text classification dataset (Prettenhofer and Stein, 2010). The dataset is a binary classification dataset where each review is classified into positive or negative sentiment. It has four languages: English, German, French and Japanese.

As shown in Table 5.5, MT-BOW uses machine translation to translate the bag of words of a target sentence into the source language, while CL-SCL learns a cross-lingual feature space via structural correspondence learning (Prettenhofer and Stein, 2010). CR-RL (Xiao and Guo, 2013) learns bilingual word representations where part of the word vector is shared among languages. Bi-

	German				French				Japanese			
Domain	books	dvd	music	*avg*	books	dvd	music	*avg*	books	dvd	music	*avg*
Methods with general-purpose cross-lingual resources												
MT-BOW[1]	79.68	77.92	77.22	78.27	80.76	78.83	75.78	78.46	70.22	71.30	72.02	71.18
CL-SCL[1]	79.50	76.92	77.79	78.07	78.49	78.80	77.92	78.40	73.09	71.07	75.11	73.09
CR-RL[2]	79.89	77.14	77.27	78.10	78.25	74.83	78.71	77.26	71.11	73.12	74.38	72.87
Bi-PV[3]	79.51	78.60	**82.45**	80.19	**84.25**	79.60	80.09	81.31	71.75	75.40	75.45	74.20
UMM[4]	81.65	81.27	81.32	81.41	80.27	80.27	79.41	79.98	71.23	72.55	75.38	73.05
Methods with task-specific cross-lingual resources												
CLDFA[5]	**83.95**	**83.14**	79.02	**82.04**	83.37	82.56	**83.31**	**83.08**	**77.36**	**80.52**	**76.46**	**78.11**
Methods without cross-lingual resources												
BWE (1-to-1)	76.00	76.30	73.50	75.27	77.80	78.60	78.10	78.17	55.93	57.55	54.35	55.94
BWE (3-to-1)	78.35	77.45	76.70	77.50	77.95	79.25	79.95	79.05	54.78	54.20	51.30	53.43
MAN-MoE	82.40	78.80	77.15	79.45	81.10	**84.25**	80.90	82.08	62.78	69.10	72.60	68.16

[1] Prettenhofer and Stein (2010) [2] Xiao and Guo (2013) [3] Pham et al. (2015)
[4] Xu and Wan (2017) [5] Xu and Yang (2017)

Table 5.5: Results for the Multilingual Amazon Reviews dataset. Numbers indicate binary classification accuracy. VecMap embeddings (Artetxe et al., 2017) are used for this experiment as MUSE training fails on Japanese (Section 4.2.1).

PV (Pham et al., 2015) extracts bilingual paragraph vector by sharing the representation between parallel documents. UMM (Xu and Wan, 2017) is a multilingual framework that could utilize parallel corpora between multiple language pairs, and pivot as needed when direct bitexts are not available for a specific source-target pair. Finally CLDFA (Xu and Yang, 2017) proposes cross-lingual distillation on parallel corpora for CLTL. Unlike other works listed, however, they adopt a task-specific parallel corpus (translated Amazon reviews) that are difficult to obtain in practice, making the numbers not directly comparable to others.

Among these methods, UMM is the only one that does not require direct parallel corpus between all source-target pairs. It can instead utilize pivot languages (e.g. English) to connect multiple languages. `MAN-MoE`, however, takes another giant leap forward to completely remove the necessity of parallel corpora while achieving similar results on German and French compared to UMM. On Japanese, the performance of `MAN-MoE` is again limited by the quality of BWEs. (BWE baselines are merely better than randomness.) Nevertheless, `MAN-MoE` remains highly effective and the performance is only a few points below most SoTA methods with cross-lingual supervision.

5.3.4 Visualization of Expert Gate Weights

For a better understanding of the model behavior, we in Figure 5.4 visualize the average expert gate weights for each of the three target languages in the Amazon dataset. For each sample, we first compute a sentence-level aggregation by averaging over the expert gate weights of all its tokens. These sentence-level ex-

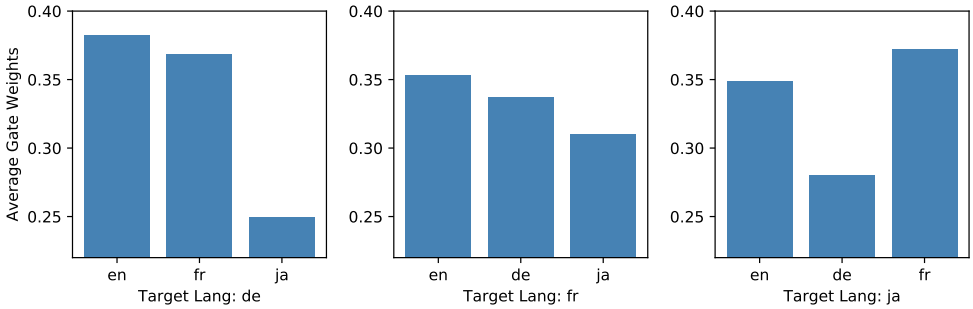

Figure 5.4: Average expert gate weights aggregated on a language level for the Amazon dataset.

pert gate weights are then further averaged across all samples in the validation set of all three domains (books, dvd, music), which forms a final language-level average expert gate weight for each target language.

The visualization further collaborates with our hypothesis that our model makes informed decisions when selecting what features to share to the target language. It can be seen that when transferring to German or French (from the remaining three), the Japanese expert is less utilized compared to the European languages. On the other hand, it is interesting that when transferring to Japanese, the French and English experts are used more than the German one, and the exact reason remains to be investigated. However, this phenomenon might be of less significance since the private features may not play a very important role when transferring to Japanese as the model is probably focusing more on the shared features, according to the ablation study in Section 5.3.1.

5.3.5 Implementation Details

In all experiments, Adam (Kingma and Ba, 2015) is used for both optimizers (main optimizer and $\mathcal{D}$ optimizer), with learning rate 0.001 and weight decay 10^{-8}. Batch size is 64 for the slot filling experiment and 16 for the NER and Amazon Reviews experiments, which is selected mainly due to memory concerns. CharCNN increases the GPU memory usage and NER hence could only use a batch size of 16 to fit in 12GB of GPU memory. The Amazon experiment does not employ character embeddings but the documents are much longer, and thus also using a smaller batch size. All embeddings are fixed during training. Dropout (Srivastava et al., 2014a) with $p = 0.5$ is applied in all components. Unless otherwise mentioned, ReLU is used as non-linear activation.

Bidirectional-LSTM is used in the feature extractors for all experiments. In particular, $\mathcal{F}_s$ is a two-layer BiLSTM of hidden size 128 (64 for each direction), and $\mathcal{F}_p$ is a two-layer BiLSTM of hidden size 128 stacked with a `MoE` module (see Figure 5.2). Each expert network in the `MoE` module of $\mathcal{F}_p$ is a two-layer MLP again of hidden size of 128. The final layer in the MLP has a *tanh* activation instead of ReLU to match the LSTM-extracted shared features (with *tanh* activations). The expert gate is a linear transformation (matrix) of size $128 \times N$, where N is the number of source languages.

On the other hand, the architecture of the task specific predictor C depends on the task. For sequence tagging experiments, the structure of C is shown in Figure 5.3, where each expert in the `MoE` module is a token-level two-layer MLP with a softmax layer on top for making token label predictions. For text classification tasks, a dot-product attention mechanism (Luong et al., 2015) is added after the shared and private features are concatenated. It has a length 256

	λ_1	λ_2	k
Slot Filling	0.01	1	5
CoNLL NER	0.0001	0.01	1
Amazon	0.002	0.1	1

Table 5.6: The hyperparameter choices for different experiments.

weight vector that attends to the feature vectors of each token and computes a softmax mixture that pools the token-level feature vectors into a single sentence-level feature vector. The rest of C remains the same for text classification.

For the language discriminator $\mathcal{D}$, a CNN text classifier (Kim, 2014) is adopted in all experiments. It takes as input the shared feature vectors of each token, and employs a CNN with max-pooling to pool them into a single fixed-length feature vector, which is then fed into a MLP for classifying the language of the input sequence. The number of kernels is 200 in the CNN, while the kernel sizes are 3, 4, and 5. The MLP has one hidden layer of size 128.

The MUSE, VecMap, and UMWE embeddings are trained with the monolingual 300d fastText Wikipedia embeddings (Bojanowski et al., 2017). When character-level word embeddings are used, a CharCNN is added that takes randomly initialized character embeddings of each character in a word, and passes them through a CNN with kernel number 200 and kernel sizes 3, 4, and 5. Finally, the character embeddings are max-pooled and fed into a single fully-connected layer to form a 128 dimensional character-level word embedding, which is concatenated with the pre-trained cross-lingual word embedding to form the final word representation of that word.

The remaining hyperparameters such as λ_1, λ_2 and k (see Algorithm 5.1) are tuned for each individual experiment, as shown in Table 5.6.

5.4 Chapter Summary

In this chapter, we propose MAN-MoE, a multilingual model transfer approach that exploits both language-invariant (shared) features and language-specific (private) features, which departs from most previous models that can only make use of shared features. Following chapters 3 and 4, the shared features are learned via language-adversarial training. On the other hand, the private features are extracted by a mixture-of-experts (MoE) module, which is able to dynamically capture the relation between the target language and each source language on a token level. This is extremely helpful when the target language is similar to a subset of the source languages, in which case traditional models that solely rely on shared features would perform poorly. Furthermore, MAN-MoE is a purely model-based transfer method, which does not require type II supervision for training, enabling zero-resource MLTL when combined with unsupervised cross-lingual word embeddings. This makes MAN-MoE more widely applicable to lower-resourced languages.

Our claim is supported by a wide range of experiments over multiple text classification and sequence tagging tasks, including a large-scale real-world industry dataset. MAN-MoE significantly outperforms all cross-lingually unsupervised baselines (trained without type II supervision) regardless of task or language. Furthermore, even considering methods with strong cross-lingual supervision, MAN-MoE is able to match or outperform these models on closer language pairs. When transferring to distant languages such as Chinese or Japanese, where the quality of cross-lingual word embeddings are unsatisfactory, MAN-MoE remains highly effective and substantially mitigates the performance gap introduced by cross-lingual supervision.

CHAPTER 6

UNSUPERVISED MULTILINGUAL WORD EMBEDDINGS

In chapters 3–5, we investigated the task of low-resource cross-lingual model transfer, focusing particularly on removing the need for type II supervision. While we proposed a series of successful models for both the bilingual and multilingual transfer cases, the performance of these methods depended on the quality of the cross-lingual word embeddings as shown in the previous chapters. As reviewed in Chapter 2, cross-lingual word representations (Klementiev et al., 2012; Mikolov et al., 2013b) create a shared embedding space for words across *two* (Bilingual Word Embeddings, BWEs) or *more* languages (Multilingual Word Embeddings, MWEs). Most earlier approaches for learning these cross-lingual word embeddings require type II supervision such as bilingual dictionaries or parallel corpora, which can potentially undermine the benefit brought by removing type II supervision dependence during model transfer. Therefore, we in this chapter continue onto looking at the task of learning *unsupervised* cross-lingual lexical representation without type II supervision[1].

There are attempts to reduce the dependence on type II supervision by requiring a very small parallel lexicon such as identical character strings (Smith et al., 2017), or numerals (Artetxe et al., 2017). Recently, a few papers (Zhang et al., 2017; Lample et al., 2018) started to look at applying our *language-adversarial training* technique to the learning of bilingual word embeddings, and proposed the first set of methods for inducing completely unsupervised bilingual word embeddings (UBWEs).

[1]Note that for the task of learning cross-lingual word embeddings, type 0 and I supervision is not relevant, and the only supervision required is type II supervision. Therefore, it can be simply referred to as unsupervised cross-lingual word embeddings without ambiguity if no type II supervision is employed during training.

In contrast to BWEs that only focus on a pair of languages, MWEs instead strive to leverage the interdependencies among multiple languages to learn a multilingual embedding space. MWEs are desirable when dealing with multiple languages simultaneously and have also been shown to improve the performance on some bilingual tasks thanks to its ability to acquire knowledge from other languages (Ammar et al., 2016; Duong et al., 2017). Despite these advantages, not much progress has been made on learning *unsupervised multilingual word embeddings* (UMWEs) that requires no type II supervision. The prior art obtains UMWEs simply through naïve combinations of the existing UBWEs, using methods such as mapping all the languages independently to the embedding space of a chosen target language[2] (usually English) (Lample et al., 2018). There are downsides, however, when using a single fixed target language with no interaction between any of the two source languages. For instance, French and Italian are very similar, and the fact that each of them is individually converted to a less similar language, English for example, in order to produce a shared embedding space will inevitably degrade the quality of the MWEs.

For certain multilingual tasks such as translating between any pair of N given languages, another option for obtaining UMWEs exists. One can directly train UBWEs for each of such language pairs (referred to as BWE-Direct). This is seldom used in practice, since it requires training $O(N^2)$ BWE models as opposed to only $O(N)$ in BWE-Pivot, and is too expensive for most use cases. Moreover, this method still does not fully exploit the language interdependence. For example, when learning embeddings between French and Italian, BWE-Direct only utilizes information from the pair itself, but other Romance languages such as Spanish may also provide valuable information that could improve perfor-

[2]Henceforth, we refer to this method as BWE-Pivot since the target language serves as a pivot to connect other languages.

mance.

In this chapter, we propose a novel unsupervised algorithm to train MWEs using *only* monolingual corpora (or equivalently, monolingual word embeddings). Our method exploits the interdependencies between any two languages and maps all monolingual embeddings into a shared multilingual embedding space via a two-stage algorithm consisting of (i) Multilingual Adversarial Training (`MAT`) and (ii) Multilingual Pseudo-Supervised Refinement (`MPSR`). As shown by experimental results on multilingual word translation and cross-lingual word similarity, our model is as efficient as BWE-Pivot yet outperforms both BWE-Pivot and BWE-Direct despite the latter being much more expensive. In addition, our model achieves a higher overall performance than state-of-the-art *supervised* methods in these experiments. Finally, as we have seen in Section 5.3.2, when applied to downstream tasks such as multilingual named entity recognition, our UMWEs yield higher performance compared to UBWEs, which serves as an extrinsic evaluation to further validate the effectiveness of our model.

This chapter is based on Chen and Cardie (2018b).

6.1 Related Work

There is a plethora of literature on learning cross-lingual word representations, focusing either on a pair of languages, or multiple languages at the same time (Klementiev et al., 2012; Zou et al., 2013; Mikolov et al., 2013b; Gouws et al., 2015; Coulmance et al., 2015; Ammar et al., 2016; Duong et al., 2017, *inter alia*). One shortcoming of these methods is the dependence on cross-lingual supervi-

sion (type II supervision) such as parallel corpora or bilingual lexica. Abundant research efforts have been made to alleviate such dependence (Vulić and Moens, 2015; Artetxe et al., 2017; Smith et al., 2017), but consider only the case of a single pair of languages (BWEs). Furthermore, fully unsupervised methods exist for learning BWEs (Zhang et al., 2017; Lample et al., 2018; Artetxe et al., 2018). For unsupervised MWEs, however, previous methods merely rely on a number of independent BWEs to separately map each language into the embedding space of a chosen target language (Smith et al., 2017; Lample et al., 2018).

Mikolov et al. (2013b) first propose to learn cross-lingual word representations by learning a linear mapping between the monolingual embedding spaces of a pair of languages. It has then been observed that enforcing the linear mapping to be orthogonal could significantly improve performance (Xing et al., 2015; Artetxe et al., 2016; Smith et al., 2017). These methods solve a linear equation called the orthogonal Procrustes problem for the optimal orthogonal linear mapping between two languages, given a set of word pairs as supervision. Artetxe et al. (2017) find that when using weak supervision (e.g. digits in both languages), applying this Procrustes process iteratively achieves higher performance. Lample et al. (2018) adopt the iterative Procrustes method with pseudo-supervision in a fully unsupervised setting and also obtain good results. In the MWE task, however, the multilingual mappings no longer have a closed-form solution, and we hence propose the `MPSR` algorithm (Section 6.2.2) for learning multilingual embeddings using gradient-based optimization methods.

6.2 Model

In this chapter, our goal is to learn a single multilingual embedding space for N languages, without relying on *any* type II supervision. We assume that we have access to monolingual embeddings for each of the N languages, which can be obtained using unlabeled monolingual corpora (Mikolov et al., 2013c; Bojanowski et al., 2017). We now present our unsupervised MWE (UMWE) model that jointly maps the monolingual embeddings of all N languages into a single space by explicitly leveraging the interdependencies between arbitrary language pairs, but is computationally as efficient as learning $O(N)$ BWEs (instead of $O(N^2)$).

Denote the set of languages as $\mathcal{L}$ with $|\mathcal{L}| = N$. Suppose for each language $l \in \mathcal{L}$ with vocabulary $\mathcal{V}_l$, we have a set of d-dimensional monolingual word embeddings $\mathcal{E}_l$ of size $|\mathcal{V}_l| \times d$. Let $\mathcal{S}_l$ denote the monolingual embedding space for l, namely the distribution of the monolingual embeddings of l. If a set of embeddings $\mathcal{E}$ are in an embedding space $\mathcal{S}$, we write $\mathcal{E} \vdash \mathcal{S}$ (e.g. $\forall l : \mathcal{E}_l \vdash \mathcal{S}_l$). Our models learns a set of encoders $\mathcal{M}_l$, one for each language l, and the corresponding decoders $\mathcal{M}_l^{-1}$. The encoders map all $\mathcal{E}_l$ to a single target space $\mathcal{T}$: $\mathcal{M}_l(\mathcal{E}_l) \vdash \mathcal{T}$. On the other hand, a decoder $\mathcal{M}_l^{-1}$ maps an embedding in $\mathcal{T}$ back to $\mathcal{S}_l$.

Previous research (Mikolov et al., 2013b) shows that there is a strong linear correlation between the vector spaces of two languages, and that learning a complex non-linear neural mapping does not yield better results. Xing et al. (2015) further show that enforcing the linear mappings to be orthogonal matrices achieves higher performance. Therefore, we let our encoders $\mathcal{M}_l$ be or-

thogonal linear matrices, and the corresponding decoders can be obtained by simply taking the transpose: $\mathcal{M}_l^{-1} = \mathcal{M}_l^\top$. Thus, applying the encoder or decoder to an embedding vector is accomplished by multiplying the vector with the encoder/decoder matrix.

Another benefit of using linear encoders and decoders (also referred to as *mappings*) is that we can learn $N - 1$ mappings instead of N by choosing the target space $\mathcal{T}$ to be the embedding space of a specific language (denoted as the *target language*) without losing any expressiveness of the model. Given a MWE with an arbitrary $\mathcal{T}$, we can construct an equivalent one with only $N - 1$ mappings by multiplying the encoders of each language $\mathcal{M}_l$ to the decoder of the chosen target language $\mathcal{M}_t^\top$:

$$\mathcal{M}_t' = \mathcal{M}_t^\top \mathcal{M}_t = I$$
$$\mathcal{M}_l' \mathcal{E}_l = (\mathcal{M}_t^\top \mathcal{M}_l)\mathcal{E}_l \vdash \mathcal{S}_t$$

where I is the identity matrix. The new MWE is isomorphic to the original one.

We now present the two major components of our approach, Multilingual Adversarial Training (Section 6.2.1) and Multilingual Pseudo-Supervised Refinement (Section 6.2.2).

6.2.1 Multilingual Adversarial Training

In this section, we introduce an adversarial training approach for learning multilingual embeddings without cross-lingual supervision. Figure 6.1 shows our Multilingual Adversarial Training (`MAT`) model and the training procedure is described in Algorithm 6.1. Note that as explained in Section 6.2, the encoders

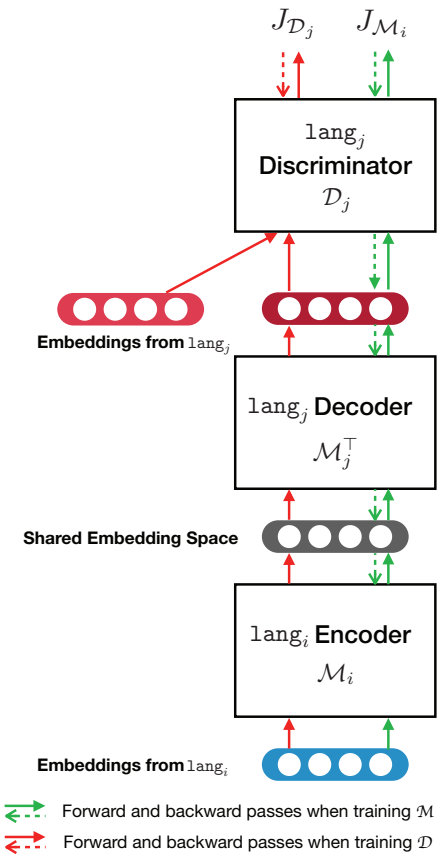

Figure 6.1: Multilingual Adversarial Training (Algorithm 6.1). lang$_i$ and lang$_j$ are two randomly selected languages at each training step. $J_{\mathcal{D}_j}$ and $J_{\mathcal{M}_i}$ are the objectives of $\mathcal{D}_j$ and $\mathcal{M}_i$, respectively (Equation 6.1 and 6.2).

and decoders adopted in practice are orthogonal linear mappings while the shared embedding space is chosen to be the same space as a selected target language.

In order to learn a multilingual embedding space without supervision, we employ a series of *language discriminators* $\mathcal{D}_l$, one for each language $l \in \mathcal{L}$. Each $\mathcal{D}_l$ is a binary classifier with a sigmoid layer on top, and is trained to identify how likely a given vector is from $\mathcal{S}_l$, the embedding space of language l. On the other hand, to train the mappings, we convert a vector from a random language

Require: Monolingual word embeddings $\mathcal{E}_i$ for each language $\texttt{lang}_i \in \mathcal{L}$. Hyperparameter $k \in \mathbb{N}$.

repeat
  ▷ $\mathcal{D}$ iterations
  for $\texttt{diter} = 1$ to k **do**
    $\texttt{loss}_d = 0$
    for all $\texttt{lang}_j \in \mathcal{L}$ **do**
      Select at random $\texttt{lang}_i \in \mathcal{L}$
      Sample a batch of word embeddings $x_i \sim \mathcal{E}_i$
      Sample a batch of word embeddings $x_j \sim \mathcal{E}_j$
      $\hat{x}_t = \mathcal{M}_i(x_i)$ ▷ encode to $\mathcal{T}$
      $\hat{x}_j = \mathcal{M}_j^\top(\hat{x}_t)$ ▷ decode to $\mathcal{S}_j$
      $y_j = \mathcal{D}_j(x_j)$ ▷ genuine embeddings
      $\hat{y}_j = \mathcal{D}_j(\hat{x}_j)$ ▷ converted embeddings
      $\texttt{loss}_d \mathrel{+}= L_d(1, y_j) + L_d(0, \hat{y}_j)$
    Update all $\mathcal{D}$ parameters to minimize $\texttt{loss}_d$
  ▷ $\mathcal{M}$ iteration
  $\texttt{loss} = 0$
  for all $\texttt{lang}_i \in \mathcal{L}$ **do**
    Select at random $\texttt{lang}_j \in \mathcal{L}$
    Sample a batch of word embeddings $x_i \sim \mathcal{E}_i$
    $\hat{x}_t = \mathcal{M}_i(x_i)$ ▷ encode to $\mathcal{T}$
    $\hat{x}_j = \mathcal{M}_j^\top(\hat{x}_t)$ ▷ decode to $\mathcal{S}_j$
    $\hat{y}_j = \mathcal{D}_j(\hat{x}_j)$
    $\texttt{loss} \mathrel{+}= L_d(1, \hat{y}_j)$
  Update all $\mathcal{M}$ parameters to minimize $\texttt{loss}$
  $\texttt{orthogonalize}(\mathcal{M})$ ▷ see Section 6.2.3
until convergence

Algorithm 6.1: Multilingual Adversarial Training

$\texttt{lang}_i$ to another random language $\texttt{lang}_j$ (via the target space $\mathcal{T}$ first). The objective of the mappings is to confuse $\mathcal{D}_j$, the language discriminator for $\texttt{lang}_j$, so the mappings are updated in a way that $\mathcal{D}_j$ cannot differentiate the converted vectors from the real vectors in $\mathcal{S}_j$. This multilingual objective enables us to explicitly exploit the relations between all language pairs during training, leading to improved performance.

Formally, for any language $\texttt{lang}_j$, the objective that $\mathcal{D}_j$ is minimizing is:

$$J_{\mathcal{D}_j} = \mathop{\mathbb{E}}_{i\sim\mathcal{L}} \mathop{\mathbb{E}}_{\substack{x_i\sim\mathcal{S}_i \\ x_j\sim\mathcal{S}_j}} \Big[\!\Big[L_d\left(1, \mathcal{D}_j(x_j)\right) + L_d\left(0, \mathcal{D}_j(\mathcal{M}_j^\top \mathcal{M}_i x_i)\right) \Big]\!\Big] \tag{6.1}$$

where $L_d(y, \hat{y})$ is the loss function of $\mathcal{D}$, which is chosen as the *cross entropy loss* in practice. y is the language label with $y = 1$ indicates a real embedding from that language.

Furthermore, the objective of $\mathcal{M}_i$ for $\texttt{lang}_i$ is:

$$J_{\mathcal{M}_i} = \mathop{\mathbb{E}}_{j\sim\mathcal{L}} \mathop{\mathbb{E}}_{\substack{x_i\sim\mathcal{S}_i \\ x_j\sim\mathcal{S}_j}} \Big[\!\Big[L_d\left(1, \mathcal{D}_j(\mathcal{M}_j^\top \mathcal{M}_i x_i)\right) \Big]\!\Big] \tag{6.2}$$

where $\mathcal{M}_i$ strives to make $\mathcal{D}_j$ believe that a converted vector to $\texttt{lang}_j$ is instead real. This adversarial relation between $\mathcal{M}$ and $\mathcal{D}$ stimulates $\mathcal{M}$ to learn a shared multilingual embedding space by making the converted vectors look as authentic as possible so that $\mathcal{D}$ cannot predict whether a vector is a genuine embedding from a certain language or converted from another language via $\mathcal{M}$.

In addition, we allow $\texttt{lang}_i$ and $\texttt{lang}_j$ to be the same language in (6.1) and (6.2). In this case, we are encoding a language to $\mathcal{T}$ and back to itself, essentially forming an adversarial autoencoder (Makhzani et al., 2015), which is reported to improve the model performance (Zhang et al., 2017). Finally, on Line 5 and 17 in Algorithm 6.1, a for loop is used instead of random sampling. This is to ensure that in each step, every discriminator (or mapping) is getting updated at least once, so that we do not need to increase the number of training iterations when adding more languages. Computationally, when compared to the BWE-Pivot and BWE-Direct baselines, one step of `MAT` training costs similarly to N BWE training steps, and in practice we train `MAT` for the same number of iterations as training the baselines. Therefore, `MAT` training scales linearly with the number of languages similar to BWE-Pivot (instead of quadratically as in BWE-Direct).

6.2.2 Multilingual Pseudo-Supervised Refinement

Using MAT, we are able to obtain UMWEs with reasonable quality, but they do not yet achieve state-of-the-art performance. Previous research on learning unsupervised BWEs (Lample et al., 2018) observes that the embeddings obtained from adversarial training do a good job aligning the frequent words between two languages, but performance degrades when considering the full vocabulary. They hence propose to use an iterative refinement method (Artetxe et al., 2017) to repeatedly refine the embeddings obtained from the adversarial training. The idea is that we can anchor on the more accurately predicted relations between frequent words to improve the mappings learned by adversarial training.

When learning MWEs, however, it is desirable to go beyond aligning each language with the target space individually, and instead utilize the relations between all languages as we did in MAT. Therefore, we in this section propose a generalization of the existing refinement methods to incorporate a multilingual objective.

In particular, MAT can produce an approximately aligned embedding space. As mentioned earlier, however, the training signals from $\mathcal{D}$ for rare words are noisier and may lead to worse performance. Thus, the idea of Multilingual Pseudo-Supervised Refinement (MPSR) is to induce a dictionary of accurately-predicted word pairs for every language pair, used as pseudo supervision to improve the embeddings learned by MAT. For a specific language pair $(\texttt{lang}_i, \texttt{lang}_j)$, the pseudo-supervised lexicon $\texttt{Lex}(\texttt{lang}_i, \texttt{lang}_j)$ is constructed from *mutual nearest neighbors* between $\mathcal{M}_i\mathcal{E}_i$ and $\mathcal{M}_j\mathcal{E}_j$, among the most frequent $15k$ words of both languages.

Require: A set of (pseudo-)supervised lexica of word pairs $\texttt{Lex}(\texttt{lang}_i, \texttt{lang}_j)$ between each pair of languages.

repeat
  $\texttt{loss} = 0$
  for all $\texttt{lang}_i \in \mathcal{L}$ **do**
    Select at random $\texttt{lang}_j \in \mathcal{L}$
    Sample $(x_i, x_j) \sim \texttt{Lex}(\texttt{lang}_i, \texttt{lang}_j)$
    $t_i = \mathcal{M}_i(x_i)$ ▷ encode x_i
    $t_j = \mathcal{M}_j(x_j)$ ▷ encode x_j
    $\texttt{loss} \mathrel{+}= L_r(t_i, t_j)$ ▷ refinement loss
  Update all $\mathcal{M}$ parameters to minimize $\texttt{loss}$
  $\texttt{orthogonalize}(\mathcal{M})$ ▷ see Section 6.2.3
until convergence

Algorithm 6.2: Multilingual Pseudo-Supervised Refinement

With the constructed lexica, the MPSR objective is:

$$J_r = \mathop{\mathbb{E}}_{(i,j)\sim\mathcal{L}^2} \mathop{\mathbb{E}}_{(x_i,x_j)\sim\texttt{Lex}(i,j)} [\![L_r(\mathcal{M}_i x_i, \mathcal{M}_j x_j)]\!] \tag{6.3}$$

where $L_r(x, \hat{x})$ is the loss function for MPSR, for which we use the *mean square loss*. The MPSR training is depicted in Algorithm 6.2.

Cross-Lingual Similarity Scaling (CSLS) When constructing the pseudo-supervised lexica, a distance metric between embeddings is needed to compute nearest neighbors. Standard distance metrics such as the Euclidean distance or the cosine similarity, however, can lead to the *hubness* problem in high-dimensional spaces when used to calculate nearest neighbors (Radovanović et al., 2010; Dinu and Baroni, 2015). Namely, some words are very likely to be the nearest neighbors of many others (hubs), while others are not the nearest neighbor of any word. This problem is addressed in the literature by designing alternative distance metrics, such as the inverted softmax (Smith et al., 2017) or the CSLS (Lample et al., 2018). In this work, we adopt the CSLS similarity as a drop-in replacement for cosine similarity whenever a distance metric is needed.

The CSLS similarity (whose negation is a distance metric) is calculated as follows:

$$\text{CSLS}(x, y) = 2\cos(x, y) - \frac{1}{n}\sum_{y' \in N_Y(x)} \cos(x, y') - \frac{1}{n}\sum_{x' \in N_X(y)} \cos(x', y) \tag{6.4}$$

where $N_Y(x)$ is the set of n nearest neighbors of x in the vector space that y comes from: $Y = \{y_1, ..., y_{|Y|}\}$, and vice versa for $N_X(y)$. In practice, we use $n = 10$.

6.2.3 Orthogonalization

As mentioned in Section 6.2, orthogonal linear mappings are the preferred choice when learning transformations between the embedding spaces of different languages (Xing et al., 2015; Smith et al., 2017). Therefore, we perform an orthogonalization update (Cisse et al., 2017) after each training step to ensure that our mappings $\mathcal{M}$ are (approximately) orthogonal:

$$\forall l : \mathcal{M}_l = (1 + \beta)\mathcal{M}_l - \beta\mathcal{M}_l\mathcal{M}_l^\top\mathcal{M}_l \tag{6.5}$$

where β is set to 0.001.

6.2.4 Unsupervised Multilingual Validation

In order to do model selection in the unsupervised setting, where no validation set can be used, a surrogate validation criterion is required that does not depend on bilingual data. Previous work shows promising results using such surrogate criteria for model validation in the bilingual case (Lample et al., 2018), and we

in this work adopt a variant adapted to our multilingual setting:

$$V(\mathcal{M}, \mathcal{E}) = \mathop{\mathbb{E}}_{(i,j)\sim P_{ij}} \left[\!\left[\texttt{mean\_csls}(\mathcal{M}_j^\top \mathcal{M}_i \mathcal{E}_i, \mathcal{E}_j) \right]\!\right]$$
$$= \sum_{i \neq j} p_{ij} \cdot \texttt{mean\_csls}(\mathcal{M}_j^\top \mathcal{M}_i \mathcal{E}_i, \mathcal{E}_j)$$

where p_{ij} forms a probability simplex. In this work, we let all $p_{ij} = \frac{1}{N(N-1)}$ so that $V(\mathcal{M}, \mathcal{E})$ reduces to the macro average over all language pairs. Using different p_{ij} values can place varying weights on different language pairs, which might be desirable in certain scenarios.

The `mean_csls` function is an unsupervised bilingual validation criterion proposed by Lample et al. (2018), which is the mean CSLS similarities between the most frequent $10k$ words and their translations (nearest neighbors).

6.3 Experiments

In this section, we present experimental results to demonstrate the effectiveness of our unsupervised MWE method on two benchmark tasks, the multilingual word translation task, and the SemEval-2017 cross-lingual word similarity task. We compare our `MAT+MPSR` method with state-of-the-art unsupervised and supervised approaches, and show that ours outperforms previous methods, supervised or not, on both tasks.

Pre-trained $300d$ fastText (monolingual) embeddings (Bojanowski et al., 2017) trained on the Wikipedia corpus are used for all systems that require monolingual word embeddings for learning cross-lingual embeddings.

6.3.1 Multilingual Word Translation

In this section, we consider the task of word translation between arbitrary pairs of a set of N languages. To this end, we use the recently released multilingual word translation dataset on six languages: English, French, German, Italian, Portuguese and Spanish (Lample et al., 2018). For any pair of the six languages, a ground-truth bilingual dictionary is provided with a train-test split of 5000 and 1500 unique source words, respectively. The $5k$ training pairs are used in training supervised baseline methods, while all unsupervised methods do not rely on any cross-lingual resources. All systems are tested on the 1500 test word pairs for each pair of languages.

For comparison, we adopt a state-of-the-art unsupervised BWE method (Lample et al., 2018) and generalize it to the multilingual setting using the two aforementioned approaches, namely BWE-Pivot and BWE-Direct, to produce baseline unsupervised MWE systems. English is chosen as the pivot language in BWE-Pivot. We further incorporate the supervised BWE-Direct (Sup-BWE-Direct) method as a baseline, where each BWE is trained on the $5k$ gold-standard word pairs via the orthogonal Procrustes process (Artetxe et al., 2017; Lample et al., 2018).

Table 6.1 and 6.2 present the evaluation results, wherein the numbers represent *precision@1*, namely how many times one of the correct translations of a source word is retrieved as the top candidate. All systems retrieve word translations using the CSLS similarity in the learned embedding space. Table 6.1 shows the detailed results for all 30 language pairs, while Table 6.2 summarizes the results in a number of ways. We first observe the training cost of all systems summarized in Table 6.2. #BWEs indicates the training cost of a certain method

	en-de	en-fr	en-es	en-it	en-pt	de-fr	de-es	de-it	de-pt	fr-es	fr-it	fr-pt	es-it	es-pt	it-pt
Supervised methods with cross-lingual supervision															
Sup-BWE-Direct	73.5	81.1	81.4	77.3	79.9	73.3	67.7	69.5	59.1	82.6	83.2	78.1	83.5	87.3	81.0
Unsupervised methods without cross-lingual supervision															
BWE-Pivot	74.0	82.3	81.7	77.0	80.7	71.9	66.1	68.0	57.4	81.1	79.7	74.7	81.9	85.0	78.9
BWE-Direct	74.0	82.3	81.7	77.0	80.7	73.0	65.7	66.5	58.5	83.1	83.0	77.9	83.3	87.3	80.5
MAT+MPSR	**74.8**	**82.4**	**82.5**	**78.8**	**81.5**	**76.7**	**69.6**	**72.0**	**63.2**	**83.9**	**83.5**	**79.3**	**84.5**	**87.8**	**82.3**
	de-en	fr-en	es-en	it-en	pt-en	fr-de	es-de	it-de	pt-de	es-fr	it-fr	pt-fr	it-es	pt-es	pt-it
Supervised methods with cross-lingual supervision															
Sup-BWE-Direct	72.4	**82.4**	82.9	76.9	**80.3**	69.5	68.3	67.5	63.7	85.8	87.1	84.3	87.3	91.5	81.1
Unsupervised methods without cross-lingual supervision															
BWE-Pivot	72.2	82.1	83.3	**77.7**	80.1	68.1	67.9	66.1	63.1	84.7	86.5	82.6	85.8	91.3	79.2
BWE-Direct	72.2	82.1	83.3	**77.7**	80.1	69.7	68.8	62.5	60.5	86	87.6	83.9	87.7	92.1	80.6
MAT+MPSR	**72.9**	81.8	**83.7**	77.4	79.9	**71.2**	**69.0**	**69.5**	**65.7**	**86.9**	**88.1**	**86.3**	**88.2**	**92.7**	**82.6**

Table 6.1: Detailed Results for Multilingual Word Translation.

	Training Cost		*Single Source*						*Single Target*						
	#BWEs	time	en-xx	de-xx	fr-xx	es-xx	it-xx	pt-xx	xx-en	xx-de	xx-fr	xx-es	xx-it	xx-pt	**Overall**
Supervised methods with cross-lingual supervision															
Sup-BWE-Direct	$N(N-1)$	$4h$	78.6	68.4	79.2	81.6	80.0	80.2	79.0	68.5	82.3	82.1	78.9	77.1	78.0
Unsupervised methods without cross-lingual supervision															
BWE-Pivot	$2(N-1)$	$8h$	79.1	67.1	77.1	80.6	79.0	79.3	**79.1**	67.8	81.6	81.2	77.2	75.3	77.0
BWE-Direct	$N(N-1)$	$23h$	79.1	67.2	79.2	81.7	79.2	79.4	**79.1**	67.1	82.6	82.1	78.1	77.0	77.6
MAT+MPSR	$N-1$	$5h$	**80.0**	**70.9**	**79.9**	**82.4**	**81.1**	**81.4**	**79.1**	**70.0**	**84.1**	**83.4**	**80.3**	**78.8**	**79.3**

Table 6.2: Summarized Results for Multilingual Word Translation.

measured by how many BWE models it is equivalent to train. BWE-Pivot needs to train $2(N-1)$ BWEs since a separate BWE is trained for each direction in a language pair for increased performance. BWE-Direct on the other hand, trains an individual BWE for all (again, directed) pairs, resulting a total of $N(N-1)$ BWEs. The supervised Sup-BWE-Direct method trains the same number of BWEs as BWE-Direct but is much faster in practice, for it does not require the unsupervised adversarial training stage. Finally, while our `MAT+MPSR` method does not train independent BWEs, as argued in Section 6.2.1, the training cost is roughly equivalent to training $N-1$ BWEs, which is corroborated by the real training time shown in Table 6.2.

We can see in Table 6.1 that our `MAT+MPSR` method achieves the highest performance on all but 3 language pairs, compared against both the unsupervised and supervised approaches. When looking at the overall performance across all language pairs, BWE-Direct achieves a +0.6% performance gain over BWE-Pivot at the cost of being much slower to train. When supervision is available, Sup-BWE-Direct further improves another 0.4% over BWE-Direct. Our `MAT+MPSR` method, however, attains an impressive 1.3% improvement against Sup-BWE-Direct, despite the lack of cross-lingual supervision.

To provide a more in-depth examination of the results, we first consider the Romance language pairs, such as fr-es, fr-it, fr-pt, es-it, it-pt and their reverse directions. BWE-Pivot performs notably worse than BWE-Direct on these pairs, which validates our hypothesis that going through a less similar language (English) when translating between similar languages will result in reduced accuracy. Our `MAT+MPSR` method, however, overcomes this disadvantage of BWE-Pivot and achieves the best performance on all these pairs through an explicit

multilingual learning mechanism without increasing the computational cost.

Furthermore, our method also beats the BWE-Direct approach, which supports our second hypothesis that utilizing knowledge from languages beyond the pair itself could improve performance. For instance, there are a few pairs where BWE-Pivot outperforms BWE-Direct, such as de-it, it-de and pt-de, even though it goes through a third language (English) in BWE-Pivot. This might suggest that for some less similar language pairs, leveraging a third language as a bridge could in some cases work better than only relying on the language pair itself. German is involved in all these language pairs where BWE-Pivot outperforms than BWE-Direct, which is potentially due to the similarity between German and the pivot language English. We speculate that if choosing a different pivot language, there might be other pairs that could benefit. This observation serves as a possible explanation of the superior performance of our multilingual method over BWE-Direct, since our method utilizes knowledge from all languages during training.

An additional observation is that for all systems, even when the predictions are incorrect, it is often the case that the top predictions are still semantically relevant to the query word. We therefore also present the *precision@5* in Table 6.3 and 6.4. It can be seen that all three systems achieve a huge performance improvement of about 10% compared to *precision@1*, reaching nearly 90% precision. Our `MAT+MPSR` method still outperforms all baseline methods and achieves the best performance.

	en-de	en-fr	en-es	en-it	en-pt	de-fr	de-es	de-it	de-pt	fr-es	fr-it	fr-pt	es-it	es-pt	it-pt
Supervised methods with cross-lingual supervision															
Sup-BWE-Direct	**88.7**	90.7	90.9	88.5	89.7	86.1	81.8	83.5	76.3	92.7	90.8	88.9	92.5	93.9	90.2
Unsupervised methods without cross-lingual supervision															
BWE-Pivot	**88.7**	90.8	91.3	88.3	89.5	85.1	80.9	82.5	74.8	91.2	90.3	88.2	91.5	93.0	89.1
BWE-Direct	**88.7**	90.8	91.3	88.3	89.5	85.5	79.3	80.1	72.8	92.7	**91.2**	87.8	92.6	**94.2**	88.8
MAT+MPSR	**88.7**	**91.1**	**91.4**	**89.3**	**90.3**	**87.3**	**82.3**	**85.4**	**77.7**	**93.1**	90.7	**90.0**	**92.7**	94.1	**91.0**

	de-en	fr-en	es-en	it-en	pt-en	fr-de	es-de	it-de	pt-de	es-fr	it-fr	pt-fr	it-es	pt-es	pt-it
Supervised methods with cross-lingual supervision															
Sup-BWE-Direct	84.8	90.8	92.0	87.8	89.6	85.1	82.9	83.0	80.5	93.5	94.0	91.3	93.7	96.1	89.3
Unsupervised methods without cross-lingual supervision															
BWE-Pivot	**85.5**	**91.3**	**92.2**	87.8	**89.9**	84.6	83.6	83.3	79.1	93.5	93.0	91.3	92.7	95.4	89.3
BWE-Direct	**85.5**	**91.3**	**92.2**	87.8	**89.9**	**85.7**	83.7	78.1	78.2	94.1	**94.3**	89.9	93.6	**96.3**	87.8
MAT+MPSR	85.3	90.7	91.7	**88.0**	89.2	85.6	**84.7**	**83.5**	**81.2**	**94.3**	94.1	**92.9**	**93.9**	96.2	**89.6**

Table 6.3: *Precision@5* for Multilingual Word Translation: Detailed Results

	Training Cost		*Single Source*						*Single Target*						
	#BWEs	time	en-xx	de-xx	fr-xx	es-xx	it-xx	pt-xx	xx-en	xx-de	xx-fr	xx-es	xx-it	xx-pt	**Overall**
Supervised methods with cross-lingual supervision															
Sup-BWE-Direct	$N(N-1)$	$4h$	89.7	82.5	89.7	91.0	89.7	89.4	89.0	84.0	91.1	91.0	88.9	87.8	88.7
Unsupervised methods without cross-lingual supervision															
BWE-Pivot	$2(N-1)$	$8h$	89.7	81.8	89.1	90.8	89.2	89.0	**89.3**	83.9	90.7	90.3	88.4	86.9	88.3
BWE-Direct	$N(N-1)$	$23h$	89.7	80.6	89.7	91.4	88.5	88.4	**89.3**	82.9	90.9	90.6	88.0	86.6	88.1
MAT+MPSR	$N-1$	$5h$	**90.2**	**83.6**	**90.0**	**91.5**	**90.1**	**89.8**	89.0	**84.7**	**91.9**	**91.4**	**89.5**	**88.6**	**89.2**

Table 6.4: *Precision@5* for Multilingual Word Translation: Summarized Results

			fr-es examples			
Source Word		littéraire			mattre	
Gold Translation(s)		**literario**			**poner**	
Model Predictions	BWE-Pivot	BWE-Direct	MAT+MPSR	BWE-Pivot	BWE-Direct	MAT+MPSR
	literaria	**literario**	**literario**	poniendo	**poner**	**poner**
	iterario	literaria	literaria	**poner**	ponerlo	ponerlo
	literaria»	literaria»	literaria»	ponerle	ponerle	ponerle
	literariamente	literariamente	literariamente	colocando	ponerles	ponerse
	literarios	novelístico	literatura	poniéndolo	ponerse	ponerla
Source Word		vérifier			plein	
Gold Translation(s)		**verificar**			**lleno**	
		comprobar				
Model Predictions	BWE-Pivot	BWE-Direct	MAT+MPSR	BWE-Pivot	BWE-Direct	MAT+MPSR
	verificando	**comprobar**	**verificar**	abierto	lonja	**lleno**
	comprobando	**verificar**	**comprobar**	ratos	abierto	ratos
	comprobar	verificarlo	verificarlo	sobresalir	repleto	diáfano
	verificar	comprobarlo	comprobarlo	quiebro	quiebro	abierto
	comprobarlo	verificarse	verificarse	salí	andador	quiebro
			de-it examples			
Source Word		deutschland			gewicht	
Gold Translation(s)		**germania**			**peso**	
Model Predictions	BWE-Pivot	BWE-Direct	MAT+MPSR	BWE-Pivot	BWE-Direct	MAT+MPSR
	germania	italia	**germania**	chilogrammi	chilogrammi	**peso**
	italia	**germania**	italia	pesare	pesare	pesare
	europa	italia,	svizzera	pesando	kilogrammi	chilogrammi
	austria	europa	europa	chilogrammo	pesano	pesando
	paesi	svizzera	italia,	pesano	chilogrammo	chilogrammo
Source Word		schließlich			bär	
Gold Translation(s)		**infine**			**orso**	
Model Predictions	BWE-Pivot	BWE-Direct	MAT+MPSR	BWE-Pivot	BWE-Direct	MAT+MPSR
	poi	poi	**infine**	**orso**	leoncino	**orso**
	successivamente	**infine**	poi	orsacchiotti	**orso**	leoncino
	infine	successivamente	dopodiché	orsacchiotto	cigno	lupo
	dopodiché	dopodiché	successivamente	leoncino	leone	leone
	nuovamente	dapprima	dapprima	bambi	unicorno	aquila

Table 6.5: Multilingual Word Translation Examples. Top 5 predictions are shown for each model. Correct predictions are highlighted.

Example System Predictions

Table 6.5 shows some examples for the multilingual word translation task. We pick two language pairs, French-Spanish and German-Italian, where in the former BWE-Direct achieves higher performance than BWE-Pivot while BWE-Pivot outperforms BWE-Direct in the latter. The top 5 predictions are shown for each model. Note that the MWT dataset we use can handle the case of multiple correct translations, and we show all the gold-standard translations for each source word in the table. For French-Spanish, we can see BWE-Pivot sometimes predicts the incorrect inflected form of the correct translation, such as wrong gender, etc. This could potentially be caused by having to go through English as a pivot, which has less inflection than the two Romance languages.

6.3.2 Cross-Lingual Word Similarity

In this section, we evaluate the quality of our MWEs on the cross-lingual word similarity (CLWS) task, which assesses how well the similarity in the cross-lingual embedding space corresponds to a human-annotated semantic similarity score. The high-quality CLWS dataset from SemEval-2017 (Camacho-Collados et al., 2017) is used for evaluation. The dataset contains word pairs from any two of the five languages: English, German, Spanish, Italian, and Farsi (Persian), annotated with semantic similarity scores.

In addition to the BWE-Pivot and BWE-Direct baseline methods, we also include the two best-performing systems on SemEval-2017, Luminoso (Speer and Lowry-Duda, 2017) and NASARI (Camacho-Collados et al., 2016) for comparison. Note that these two methods are supervised, and have access to the

	en-de	en-es	de-es	en-it	de-it	es-it	en-fa	de-fa	es-fa	it-fa	**Average**
Supervised methods with cross-lingual supervision											
Luminoso	**.769**	**.772**	**.735**	**.787**	**.747**	**.767**	.595	.587	.634	.606	.700
NASARI	.594	.630	.548	.647	.557	.592	.492	.452	.466	.475	.545
Unsupervised methods without cross-lingual supervision											
BWE-Pivot	.709	.711	.703	.709	.682	.721	.672	.655	.701	.688	.695
BWE-Direct	.709	.711	.703	.709	.675	.726	.672	.662	.714	.695	.698
MAT+MPSR	.711	.712	.708	.709	.684	.730	**.680**	**.674**	**.720**	**.709**	**.704**

Table 6.6: Results for the SemEval-2017 Cross-Lingual Word Similarity task. Spearman's ρ is reported. Luminoso (Speer and Lowry-Duda, 2017) and NASARI (Camacho-Collados et al., 2016) are the two top-performing systems for SemEval-2017 that reported results on all language pairs.

Europarl[3] (for all languages but Farsi) and the OpenSubtitles2016[4] parallel corpora.

Table 6.6 shows the results, where the performance of each model is measured by the Spearman correlation. When compared to the BWE-Pivot and the BWE-Direct baselines, MAT+MPSR continues to perform the best on all language pairs. The qualitative findings stay the same as in the word translation task, except the margin is less significant. This might be because the CLWS task is much more lenient compared to the word translation task, where in the latter one needs to correctly identify the translation of a word out of hundreds of thousands of words in the vocabulary. In CLWS though, one can still achieve relatively high correlation in spite of minor inaccuracies.

On the other hand, an encouraging result is that when compared to the state-of-the-art supervised results, our MAT+MPSR method outperforms NASARI by a very large margin, and achieves top-notch overall performance similar to the competition winner, Luminoso, without using any bitexts. A closer examination reveals that our unsupervised method lags a few points behind Luminoso on the European languages wherein the supervised methods have access to the large-scale high-quality Europarl parallel corpora. It is the low-resource language, Farsi, that makes our unsupervised method stand out. All of the unsupervised methods outperform the supervised systems from SemEval-2017 on language pairs involving Farsi, which is not covered by the Europarl bitexts. This suggests the advantage of learning unsupervised embeddings for lower-resourced languages, where the supervision might be noisy or absent. Furthermore, within the unsupervised methods, MAT+MPSR again performs the best,

[3] http://opus.nlpl.eu/Europarl.php
[4] http://opus.nlpl.eu/OpenSubtitles2016.php

and attains a higher margin over the baseline approaches on the low-resource language pairs, vindicating our claim of better multilingual performance.

6.3.3 Implementation Details

As the $300d$ fastText monolingual embeddings are employed, the dimension of our encoders $\mathcal{M}$ is 300×300. Since the vocabulary of the full fastText embeddings are very large, we only select $200k$ most frequent words for each language, following prior work on learning BWEs (Lample et al., 2018). The discriminator for each language is a fully connected neural network with two hidden layers of size 2048 and Leaky-ReLU activation (Maas et al., 2013). Dropout (Srivastava et al., 2014b) with $p = 0.1$ is used on the input to the discriminators. We also use (two-sided) label smoothing (Salimans et al., 2016) with $s = 0.1$ for the discriminators. We use a batch size of 32.

In the `MAT` training, two identical Stochastic Gradient Descent (SGD) optimizers are used for training $\mathcal{M}$ and $\mathcal{D}$, with a learning rate of 0.1 and a decay of 0.98. The learning rate is halved at every epoch in which the unsupervised multilingual validation score does not increase. We use $k = 5$ in the `MAT` Algorithm. Similar to Lample et al. (2018), we only feed the discriminators with the $75k$ most frequent words in each language. A total of 5 `MAT` epochs are trained, where each epoch consists of one million samples (i.e. $\sim 31k$ steps). On the other hand, `MPSR` is trained with Adam (Kingma and Ba, 2015) at a learning rate of 0.001. `MPSR` is also trained for 5 epochs, and each epoch has $30k$ steps. The Faiss library (Johnson et al., 2017) is used to accelerate the nearest neighbor computation. Our model is implemented using PyTorch (Paszke et al., 2017)[5].

[5]The source code is available at `https://github.com/ccsasuke/umwe`.

6.4 Chapter Summary

In this chapter, we propose a fully unsupervised model for learning multilingual word embeddings (MWEs). Although methods exist for learning high-quality unsupervised BWEs (Lample et al., 2018), little work has been done in the unsupervised multilingual setting. Previous work relies solely on a number of unsupervised BWE models to generate MWEs (e.g. BWE-Pivot and BWE-Direct), which does not fully leverage the interdependencies among all the languages. Therefore, we propose the `MAT+MPSR` method that explicitly exploits the relations between all language pairs without increasing the computational cost. In our experiments on multilingual word translation and cross-lingual word similarity (SemEval-2017), we show that `MAT+MPSR` outperforms existing unsupervised and even supervised models, achieving new state-of-the-art performance.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this dissertation, we proposed a series of models for learning deep hidden feature representations suited for performing cross-lingual natural language processing tasks. In particular, our focus is to eliminate the dependence on type I (task-specific target language annotations) and type II (general-purpose cross-lingual resources) supervision, so that our cross-lingual NLP models can be applied to a wider range of low-resource languages without such supervision.

7.1 Summary of Contributions

In Chapter 3, we present language-adversarial training, a pioneering effort on cross-lingual model transfer that does not require type II supervision during training. Language-adversarial training is a deep representation learning approach for learning language-invariant feature representations by striving to delude an adversarially trained language discriminator whose goal is to identify the language of a given sample. Experiments are conducted for cross-lingual text classification, and our method outperforms several baseline methods (with type II supervision) even when our method is not using any type II supervision at all. Furthermore, when combined with cross-lingual word embeddings (trained with type II supervision), it outperformed all existing methods including strong machine translation baseline as well as the previous state of the art.

In Chapter 4, we generalize language-adversarial training to support multiple languages. Moreover, we go beyond cross-lingual model transfer and propose the multinomial adversarial network (`MAN`), a general machine learning

framework that can serve as a tool to minimize the divergence among *multiple* probability distributions. For instance, when applied to cross-lingual model transfer, the divergence between the distributions of the feature vectors of the samples from the source and target languages are minimized. Similarly, the same technique can be used in domain adaptation, where the divergence between the feature distributions of various domains are minimized instead. We provide theoretical justification of MAN, showing it minimizes the (generalized) f-divergence among multiple distributions, and empirically validate its effectiveness on multiple multi-domain text classification tasks.

Chapter 5 presents an improved method for the multi-source cross-lingual model transfer (also known as multilingual model transfer) task, compared to the MAN model. In particular, MAN is only able to leverage the language-invariant features for model transfer, which is often too restrictive in the multi-source setting. For instance, when transferring from English, Chinese and Spanish to German, MAN will wipe out features that are not shared across all four languages, and the remaining features may be very sparse. On the other hand, the model is not able to utilize features that are only shared between German and more similar source languages such as English. Therefore, we propose a MAN-MoE model that can use both language-invariant and language-specific features for multilingual model transfer, which demonstrates impressive empirical performance.

Finally in Chapter 6, we look at the other important sub-problem within the task of cross-lingual transfer learning, the cross-lingual lexical representation task. While there have been previous attempts to apply our language-adversarial training technique to this problem, and obtained unsupervised cross-lingual word embeddings, these efforts focus only on the bilingual case.

We in Chapter 6 propose the first method for learning unsupervised multilingual word embeddings, which shows strong performance in the experiments. These unsupervised cross-lingual word embeddings can also be combined with the cross-lingual model transfer techniques in chapters 3–5 to achieve *zero-resource* cross-lingual NLP that requires neither type I nor type II supervision (see Chapter 5).

7.2 Future Work

To conclude this dissertation, we in this section outline several potential future work directions.

Model transfer between distant language pairs. This dissertation demonstrates that strong performance can be achieved for cross-lingual transfer learning even without type II supervision. On the other hand, as shown in Chapter 5, while zero-resource methods can often achieve comparable performance to the baselines methods with access to type II supervision (e.g. machine translation) *between closer languages* such as English - French, English - German, etc., the performance gap remains present between type-II-supervised and -unsupervised methods on more distant language pairs such as English - Chinese and English - Japanese. Moreover, Chapter 5 shows that the mixture of experts model is more effective when transferring to closer languages, while language-adversarial training plays an more important role when transferring to more distant languages. The closeness of languages, however, is not a binary attribute, which may suggest that it can be beneficial to perform cross-lingual

transfer in a more systematic way by leveraging language similarity. For instance, a hierarchical cross-lingual transfer can be performed based on language typology, which may be able to improve the performance on more distant languages.

Cross-lingual contextualized lexical representation In Chapter 6, we saw that the state of the art for inducing cross-lingual lexical representation still relies on learning a static linear transformation to connect the word embeddings spaces of two languages, which can be improved in several aspects. For instance, there are recent papers that attempt to go beyond linear transformations and seek to model the complex relation between the semantic spaces of two languages using more sophisticated mappings (Nakashole and Flauger, 2018). One can also substitute the static mapping that connects the two languages by representing a target language word as a dynamic combination of all source language words where the weights are calculated using an attention-style mechanism. This has been experimented in machine translation (Gu et al., 2018).

Finally, we can also take one step further to replace the word embeddings with contextualized lexical representation, which has demonstrated impressive performance gains over traditional word embeddings on a variety of monolingual NLP tasks (Peters et al., 2018; Devlin et al., 2018). Many techniques described in this dissertation can be applied to induce cross-lingual contextualized word embeddings, while not requiring any additional type II supervision.

Transfer learning for NLP beyond the cross-lingual case Another future work direction is to study transfer learning in a broader sense. Transfer learning is a powerful machine learning framework for learning with fewer human an-

notations, by leveraging *related* labeled data such as that in a different domain, language, or even for a different task. Most of this dissertation focuses on the cross-lingual transfer case, where the labeled data comes from a different language. Many techniques proposed in this dissertation, though, can be readily applied to other transfer learning scenarios as well. For example, we studied the domain adaptation problem in Chapter 4, where the labeled data comes from a different domain. As machine learning is being applied to more and more NLP tasks, the scarcity of labeled data is becoming an increasingly important problem. It is hence worth exploring more general transfer learning approaches to alleviate this problem and to enable learning modern deep neural models for more and more NLP problems.

BIBLIOGRAPHY

S. M. Ali and S. D. Silvey. 1966. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society. Series B (Methodological)*, 28(1):131–142.

Massih Amini, Nicolas Usunier, and Cyril Goutte. 2009. Learning from multiple partially observed views - an application to multilingual text categorization. In *Advances in Neural Information Processing Systems 22*, pages 28–36. Curran Associates, Inc.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A. Smith. 2016. Massively multilingual word embeddings. *CoRR*, abs/1602.01925.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 214–223, Sydney, Australia.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, Austin, Texas. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798. Association for Computational Linguistics.

Javed A. Aslam and Virgil Pavlu. 2007. Query hardness estimation using jensen-shannon divergence among multiple scoring functions. In *Advances in Information Retrieval*, pages 198–209, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR 2015)*.

Gükhan H. Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan. 2007. *Predicting Structured Data (Neural Information Processing)*. The MIT Press.

Carmen Banea, Rada Mihalcea, and Janyce Wiebe. 2010. Multilingual subjectivity: Are more languages better? In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 28–36, Beijing, China.

Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. 2008. Multilingual subjectivity analysis using machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 127–135, Honolulu, Hawaii.

Nuria Bel, Cornelis H. A. Koster, and Marta Villegas. 2003. Cross-lingual text

categorization. In *Research and Advanced Technology for Digital Libraries*, pages 126–139, Berlin, Heidelberg.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *Advances in Neural Information Processing Systems 29*, pages 343–351. Curran Associates, Inc.

Jose Camacho-Collados, Mohammad Taher Pilehvar, Nigel Collier, and Roberto Navigli. 2017. Semeval-2017 task 2: Multilingual and cross-lingual semantic word similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 15–26, Vancouver, Canada. Association for Computational Linguistics.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence*, 240:36–64.

Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 767–774, Edinburgh, Scotland, GB.

Xilun Chen, Ahmed Hassan Awadallah, Hany Hassan, Wei Wang, and Claire Cardie. 2018a. Zero-resource multilingual model transfer: Learning what to share. *CoRR*, abs/1810.03552.

Xilun Chen and Claire Cardie. 2018a. Multinomial adversarial networks for multi-domain text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1226–1240. Association for Computational Linguistics.

Xilun Chen and Claire Cardie. 2018b. Unsupervised multilingual word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 261–270, Brussels, Belgium. Association for Computational Linguistics.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2016. Adversarial deep averaging networks for cross-lingual sentiment classification. *ArXiv e-prints 1606.01614*.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018b. Adversarial deep averaging networks for cross-lingual sentiment clas-

sification. *Transactions of the Association for Computational Linguistics*, 6:557–570.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. 2017. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 854–863, International Convention Centre, Sydney, Australia. PMLR.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Jocelyn Coulmance, Jean-Marc Marty, Guillaume Wenzek, and Amine Benhalloum. 2015. Trans-gram, fast cross-lingual word-embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1109–1113, Lisbon, Portugal. Association for Computational Linguistics.

Imre Csiszar and Janos Korner. 1982. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Academic Press, Inc., Orlando, FL, USA.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*.

Georgiana Dinu and Marco Baroni. 2015. Improving zero-shot learning by mitigating the hubness problem. In *International Conference on Learning Representations, Workshop Track*.

Cícero Nogueira Dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1818–II–1826. JMLR.org.

Long Duong, Hiroshi Kanayama, Tengfei Ma, Steven Bird, and Trevor Cohn. 2017. Multilingual training of crosslingual word embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 894–904, Valencia, Spain. Association for Computational Linguistics.

Theodoros Evgeniou and Massimiliano Pontil. 2004. Regularized multi–task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 109–117, New York, NY, USA. ACM.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1180–1189, Lille, France.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 513–520, Bellevue, Washington, USA.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680, Montreal, Canada.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. BilBOWA: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 748–756, Lille, France.

Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O.K. Li. 2018. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 344–354. Association for Computational Linguistics.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and

Aaron Courville. 2017. Improved training of Wasserstein GANs. In *Advances in Neural Information Processing Systems 30*, pages 5767–5777, Long Beach, USA.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2016. A representation learning framework for multi-source transfer parsing. In *AAAI Conference on Artificial Intelligence*.

Jiang Guo, Darsh Shah, and Regina Barzilay. 2018. Multi-source domain adaptation with mixture of experts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4694–4703, Brussels, Belgium. Association for Computational Linguistics.

Mohammad Sadegh Hajmohammadi, Roliana Ibrahim, Ali Selamat, and Alireza Yousefpour. 2014. Combination of multi-view multi-source language classifiers for cross-lingual sentiment classification. In *Intelligent Information and Database Systems*, pages 21–30. Springer International Publishing.

Zellig S. Harris. 1954. Distributional structure. *WORD*, 10(2-3):146–162.

Sepp Hochreiter and JÃijrgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Nat. Lang. Eng.*, 11(3):311–325.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 448–456, Lille, France.

Ozan İrsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 720–728, Doha, Qatar.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Shafiq Joty, Preslav Nakov, Lluís Màrquez, and Israa Jaradat. 2017. Cross-language learning with adversarial neural networks: Application to community question answering. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 226–237, Vancouver, Canada.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, San Diego, California.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. In *Proceedings of COLING 2012*, pages 1459–1474, Mumbai, India. The COLING 2012 Organizing Committee.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*, Vancouver, Canada.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Shoushan Li and Chengqing Zong. 2008. Multi-domain sentiment classification. In *Proceedings of ACL-08: HLT, Short Papers*, pages 257–260, Columbus, Ohio. Association for Computational Linguistics.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.

Yiou Lin, Hang Lei, Jia Wu, and Xiaoyu Li. 2015. An empirical study on sentiment classification of chinese review using word embedding. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation: Posters*, pages 258–266, Shanghai, China.

Michael L. Littman, Susan T. Dumais, and Thomas K. Landauer. 1998. *Automatic Cross-Language Information Retrieval Using Latent Semantic Indexing*, pages 51–62. Springer US, Boston, MA.

Biao Liu, Minlie Huang, Jiashen Sun, and Xuan Zhu. 2015. Incorporating domain and sentiment supervision in representation learning for domain adaptation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 1277–1283, Buenos Aires, Argentina.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 1–10, Vancouver, Canada. Association for Computational Linguistics.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. 2018. Are GANs created equal? a large-scale study. In *Advances in Neural Information Processing Systems 31*, pages 700–709. Curran Associates, Inc.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal.

Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2009. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems 21*, pages 1041–1048. Curran Associates, Inc.

Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2813–2821.

Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. 2017. Cheap translation for cross-lingual named entity recognition. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2536–2545. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rada Mihalcea, Carmen Banea, and Janyce Wiebe. 2007. Learning multilingual subjective language via cross-lingual projections. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 976–983. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations 2013 Workshop*.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013c. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Informa-*

tion Processing Systems - Volume 2, pages 3111–3119, USA. Curran Associates Inc.

Saif M. Mohammad, Mohammad Salameh, and Svetlana Kiritchenko. 2016. How translation alters sentiment. *Journal of Artificial Intelligence Research*, 55(1):95–130.

Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Ndapa Nakashole and Raphael Flauger. 2018. Characterizing departures from linearity in word translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 221–227, Melbourne, Australia. Association for Computational Linguistics.

Jian Ni, Georgiana Dinu, and Radu Florian. 2017. Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1470–1480. Association for Computational Linguistics.

Frank Nielsen and Richard Nock. 2014. On the chi square and higher-order chi distances for approximating f-divergences. *IEEE Signal Processing Letters*, 21(1):10–13.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. 2013. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151–175.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. 2016. f-GAN: Training generative neural samplers using variational divergence minimization. In

Advances in Neural Information Processing Systems 29, pages 271–279. Curran Associates, Inc.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection of semantic roles. *Journal of Artificial Intelligence Research*, 36(1):307–340.

Sinno J. Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. 2011. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210.

Sinno J. Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS 2017 Autodiff Workshop*, Long Beach, USA.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Hieu Pham, Minh-Thang Luong, and Christopher Manning. 2015. Learning distributed representations for multilingual text sequences. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 88–94, Denver, Colorado. Association for Computational Linguistics.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1127. Association for Computational Linguistics.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. *J. Mach. Learn. Res.*, 11:2487–2531.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2017. A survey of cross-lingual word embedding models. *arXiv preprint arXiv:1706.04902*.

Sebastian Rudolph and Eugenie Giesbrecht. 2010. Compositional matrix-space models of language. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 907–916, Uppsala, Sweden. Association for Computational Linguistics.

Mohammad Salameh, Saif Mohammad, and Svetlana Kiritchenko. 2015. Sentiment after translation: A case-study on arabic social media posts. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 767–777, Denver, Colorado.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Proceedings of the*

30th International Conference on Neural Information Processing Systems, NIPS'16, pages 2234–2242, USA. Curran Associates Inc.

Erik F. Tjong Kim Sang. 2002. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Michael D Shapiro and Matthew B Blaschko. 2004. On hausdorff distance measures. Technical Report UM-CS-2004-071, University of Massachusetts Amherst.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *Proceedings of ICLR*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, D. Christopher Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013*

Conference on Empirical Methods in Natural Language Processing, pages 1631–1642, Seattle, Washington, USA.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788. Association for Computational Linguistics.

Robert Speer and Joanna Lowry-Duda. 2017. Conceptnet at semeval-2017 task 2: Extending word embeddings with multilingual relational knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 85–89.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014a. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014b. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Pawel Swietojanski, Arnab Ghoshal, and Steve Renals. 2012. Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 246–251.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487. Association for Computational Linguistics.

Sheng Kai Tai, Richard Socher, and D. Christopher Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016. Cross-lingual named entity recognition via wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 219–228. Association for Computational Linguistics.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden.

Shyam Upadhyay, Manaal Faruqui, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2018. (Almost) zero-shot cross-lingual spoken language understanding. In *Proceedings of the IEEE ICASSP*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Cédric Villani. 2008. *Optimal transport: old and new*, volume 338. Springer Science & Business Media.

Alexei Vinokourov, Nello Cristianini, and John Shawe-Taylor. 2003. Inferring a semantic representation of text via cross-language correlation analysis. In

Advances in Neural Information Processing Systems 15, pages 1497–1504. MIT Press.

Ivan Vulić and Marie-Francine Moens. 2015. Bilingual word embeddings from non-parallel document-aligned data applied to bilingual lexicon induction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 719–725, Beijing, China. Association for Computational Linguistics.

Xiaojun Wan. 2008. Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 553–561, Honolulu, Hawaii.

Xiaojun Wan. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 235–243, Suntec, Singapore. Association for Computational Linguistics.

Mengqiu Wang and Christopher D. Manning. 2014. Cross-lingual projected expectation regularization for weakly supervised learning. *Transactions of the Association for Computational Linguistics*, 2:55–66.

Fangzhao Wu and Yongfeng Huang. 2015. Collaborative multi-domain sentiment classification. In *2015 IEEE International Conference on Data Mining*, pages 459–468.

Min Xiao and Yuhong Guo. 2013. Semi-supervised representation learning for cross-lingual text classification. In *Proceedings of the 2013 Conference on Empir-*

ical Methods in Natural Language Processing, pages 1465–1475. Association for Computational Linguistics.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379. Association for Computational Linguistics.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, Denver, Colorado. Association for Computational Linguistics.

Kui Xu and Xiaojun Wan. 2017. Towards a universal sentiment classifier in multiple languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 511–520, Copenhagen, Denmark. Association for Computational Linguistics.

Ruochen Xu and Yiming Yang. 2017. Cross-lingual distillation for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Vancouver, Canada.

Yi Yang and Jacob Eisenstein. 2015. Unsupervised multi-domain adaptation with feature embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 672–682. Association for Computational Linguistics.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learn-

ing for sequence tagging with hierarchical recurrent networks. In *International Conference on Learning Representations*.

D. Yarowsky, G. Ngai, and R. Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the First International Conference on Human Language Technology Research*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI Conference on Artificial Intelligence*.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970, Vancouver, Canada.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.

Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, Joao P. Costeira, and Geoffrey J. Gordon. 2018. Adversarial multiple source domain adaptation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8568–8579. Curran Associates, Inc.

J. Zhou, J. Chen, and J. Ye. 2011. *MALSAR: Multi-tAsk Learning via StructurAl Regularization*. Arizona State University.

Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2016. Cross-lingual sentiment classification with bilingual document representation learning. In *Proceedings of*

the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1403–1412, Berlin, Germany.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The united nations parallel corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 3530–3534, Portorož, Slovenia.

Will Y. Zou, Richard Socher, Daniel Cer, and Christopher D. Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, Seattle, Washington, USA.