Title
Cross-lingual Representation Learning for Natural Language Processing

Author
Ahmad, Wasi Uddin

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Cross-lingual Representation Learning for Natural Language Processing

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Wasi Uddin Ahmad

2021

ABSTRACT OF THE DISSERTATION

Cross-lingual Representation Learning for Natural Language Processing

by

Wasi Uddin Ahmad
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2021
Professor Kai-Wei Chang, Chair

In the modern era of deep learning, developing natural language processing (NLP) systems require large-scale annotated data. However, it is unfortunate that most large-scale labeled datasets are only available in a handful of languages; for the vast majority of languages, either a few or no annotations are available to empower automated NLP applications. Hence, one of the focuses of cross-lingual NLP research is to develop computational approaches by leveraging resource-rich language corpora and utilize them in low-resource language applications via transferable representation learning. Cross-lingual representation learning has emerged as an indispensable ingredient for cross-lingual natural language understanding that learns to embed notions, such as meanings of words, how the words are combined to form a concept, etc., in shared representation space. In recent years, cross-lingual representation learning and transfer learning together have redefined low-resource NLP and enabled us to build models for a broad spectrum of languages.

This dissertation discusses the fundamental challenges and proposes several approaches for *cross-lingual* representation learning that (1) utilize universal syntactic dependencies to bridge the typological differences across languages and (2) effectively use unlabeled resources to learn robust and generalizable representations. The proposed approaches in this dissertation effectively transfer across a wide range of languages across different NLP applications, including dependency parsing, named entity recognition, text classification, question answering, and more.

The dissertation of Wasi Uddin Ahmad is approved.

Guy Van den Broeck

Yizhou Sun

Junghoo Cho

Kai-Wei Chang, Committee Chair

University of California, Los Angeles

2021

Dedicated to my Ma and Baba

TABLE OF CONTENTS

1 Introduction **1**

1.1 Overview 1

1.2 Thesis Statement 3

1.3 Outline of This Thesis 3

2 Background: Word Representation Learning **6**

2.1 Introduction 6

2.2 Monolingual Word Representations 7

2.2.1 Contextualized Word Representations 9

2.3 Cross-lingual Word Representations 11

2.3.1 Resources for Learning Cross-lingual Representations 11

2.3.2 Techniques for Cross-lingual Representations Learning 12

2.3.3 Multilingual Word Representations 13

3 Cross-Lingual Transfer with Order Differences **14**

3.1 Introduction 14

3.2 Quantifying Language Distance 16

3.3 Models 18

3.3.1 Contextual Encoders 18

3.3.2 Structured Decoders 20

3.4 Experiments and Analysis 20

3.4.1 Setup 21

3.4.2 Results 23

3.4.3 Analysis 23

3.5 Related Work . . . 30
3.6 Summary . . . 31

4 Cross-lingual Representation Learning for Information Extraction . . . 33

4.1 Introduction . . . 33
4.2 Background . . . 36
4.3 Approach . . . 37
4.3.1 Transformer Encoder . . . 38
4.3.2 Graph Attention Transformer Encoder . . . 39
4.3.3 Relation Extractor . . . 41
4.3.4 Event Argument Role Labeler . . . 42
4.4 Experiment Setup . . . 42
4.4.1 Dataset . . . 43
4.4.2 Evaluation Criteria . . . 43
4.4.3 Baseline Models . . . 44
4.4.4 Implementation Details . . . 45
4.5 Results and Analysis . . . 46
4.5.1 Single-source transfer . . . 46
4.5.2 Multi-source transfer . . . 47
4.5.3 Encoding dependency structure . . . 48
4.5.4 Sensitivity towards source language . . . 49
4.5.5 Ablation study . . . 51
4.5.6 Error Analysis . . . 51
4.6 Related Work . . . 52
4.7 Summary . . . 53

5 Syntax-augmented Pre-trained Encoders for Cross-lingual Transfer . 58

5.1 Introduction . . . 58

5.2 Syntax-augmented Multilingual BERT . . . 61

5.2.1 Transformer Encoder . . . 61

5.2.2 Graph Attention Network . . . 62

5.2.3 Syntax-augmented Transformer Encoder . . . 64

5.2.4 Fine-tuning . . . 65

5.3 Experiment Setup . . . 67

5.3.1 Evaluation Tasks . . . 67

5.3.2 Implementation Details . . . 68

5.4 Experiment Results . . . 70

5.4.1 Cross-lingual Transfer . . . 70

5.4.2 Generalized Cross-lingual Transfer . . . 71

5.4.3 Analysis & Discussion . . . 72

5.4.4 Limitations and Challenges . . . 73

5.5 Related Work . . . 74

5.6 Summary . . . 75

6 Representation Learning using Unlabeled Data . . . 76

6.1 Introduction . . . 76

6.2 Language-agnostic Representation Learning . . . 77

6.2.1 Training Language-agnostic Encoders . . . 78

6.2.2 Proposed Method . . . 79

6.2.3 Experiments and Analysis . . . 82

6.2.4 Related Work . . . 87

6.3 Representation Learning for Programming Languages 89
6.3.1 Denoising Pre-training 92
6.3.2 Experiments Setup 97
6.3.3 Results & Analysis 100
6.3.4 Related Work 105
6.4 Summary 106

7 Conclusion and Future Work 107
7.1 Summary of Contributions 107
7.2 Future Work 110

References 112

LIST OF TABLES

3.1 The selected languages grouped by language families. "IE" is the abbreviation of Indo-European. 16

3.2 Results (UAS%/LAS%, excluding punctuation) on the test sets. Languages are sorted by the word-ordering distance to English, as shown in the second column. '*' refers to results of delexicalized models, '†' means that the best transfer model is statistically significantly better (by paired bootstrap test, $p < 0.05$) than all other transfer models. Models are marked with their encoder and decoder order sensitivity, OF denotes order-free and OS denotes order-sensitive. 22

3.3 Comparisons of different encoders (averaged results over all languages on the original training sets). 24

3.4 Relative frequencies (%) of dependency distances. English differs from the Average at d=1. 29

4.1 Average sequential and syntactic (shortest path) distance between relation mentions and event mentions and their candidate arguments in ACE05 dataset. Distances are computed by ignoring the order of mentions. 42

4.2 Statistics of the ACE 2005 dataset. 42

4.3 Single-source transfer results (F-score % on the test set) using perfect event triggers and entity mentions. The language on top and bottom of ⇓ denotes the source and target languages, respectively. 43

4.4 Multi-source transfer results (F-score % on the test set) using perfect event triggers and entity mentions. The language on top and bottom of ⇓ denotes the source and target languages, respectively. 44

4.5 GATE vs. Wang et al. (2019b) results (F-score %) on event argument role labeling (EARL) and relation extraction (RE); using English as source and Chinese, Arabic as the target languages, respectively. To limit the maximum relative position, the clipping distance is set to 10 and 5 for EARL and RE tasks, respectively. 47

4.6 Event argument role labeling (EARL) and relation extraction (RE) results (F-score %); using Chinese as the source and English as the target language. * indicates the English examples are translated into Chinese using `Google Cloud Translate`. 49

4.7 Contribution of multilingual word embeddings (Multi-WE) Joulin et al. (2018), M-BERT Devlin et al. (2019), and XLM-R Conneau et al. (2019) as a source of word features; using English as source and Chinese, Arabic as the target languages, respectively. 50

4.8 Ablation on the use of language-universal features (part-of-speech (POS) tag, dependency relation label, and entity type) in GATE (F-score (%); using English as source and Chinese, Arabic as the target languages, respectively. 51

4.9 Comparing GATE and Self-Attention on the EARL task using English and Chinese as the source and target languages, respectively. The rates are aggregated from confusion matrices shown in Figure 4.4 and 4.5. 52

5.1 Statistics of the evaluation datasets. |Train|, |Dev| and |Test| are the numbers of examples in the training, dev and test sets, respectively. For train set, the number is for the source language, English, while for dev and test set, the number is for each target language. |Lang| is the number of target languages we consider for each task. 66

5.2 Cross-lingual transfer results for all the evaluation tasks (on test set) across 17 languages. We report F1 score for the question answering (QA) datasets (for other datasets, see Table 5.1). We train and evaluate mBERT on the same pre-processed datasets and considers its performance as the *baseline* (denoted by "mBERT" rows in the table) for syntax-augmented mBERT (denoted by "+ Syn." rows in the table). Bold-faced values indicate that the syntax-augmented mBERT is statistically significantly better (by paired bootstrap test, $p < 0.05$) than the *baseline*. We include results from published works ([1]: Hu et al. (2020), [2]: Liang et al. (2020), and [3]: Lewis et al. (2020b)) as a reference. Except for the QA datasets, all our results are averaged over three different seeds. 69

5.3 The performance difference between syntax-augmented mBERT and mBERT in the *generalized* cross-lingual transfer setting. The rows and columns indicate (a) language of the first and second sentences in the candidate pairs and (b) context and question languages. The gray cells have a value greater than or equal to the average performance difference, which is 3.9 and 3.1 for (a) and (b). 70

6.1 The selected languages grouped by language families. "IE" is the abbreviation of Indo-European. 82

6.2 Cross-lingual transfer performances (UAS%/LAS%, excluding punctuation) of the SelfAtt-Graph parser Ahmad et al. (2019c) on the test sets. In column 1, languages are sorted by the word-ordering distance to English. (en-fr) and (en-ru) denotes the source-auxiliary language pairs. '†' indicates that the adversarially trained model results are statistically significantly better (by permutation test, $p < 0.05$) than the model trained only on the source language (en). Results show that the utilization of unlabeled auxiliary language corpora improves cross-lingual transfer performance significantly. 85

6.3 Comparison between adversarial training (AT) and multi-task learning (MTL) of the contextual encoders. Columns 2–5 demonstrate the parsing performances (UAS%/LAS%, excluding punctuation) on the auxiliary languages and average of the 29 languages. Columns 6–7 present accuracy (%) of the language label prediction test. '†' indicates that the performance is higher than the baseline performance (shown in the 2nd column of Table 6.2). 86

6.4 Statistics of the data used to pre-train PLBART. "Nb of documents" refers to the number of functions in Java and Python collected from Github and the number of posts (questions and answers) in the natural language (English) from StackOverflow. 92

6.5 Example encoder inputs and decoder outputs during denoising pre-training of PLBART. We use three noising strategies: token masking, token deletion, and token infilling (shown in the three examples, respectively). 93

6.6 Example inputs to the encoder and decoder for fine-tuning PLBART on sequence generation tasks: source code summarization (S), generation (G), and translation (T). 94

6.7 Statistics of the downstream benchmark datasets. 96

6.8 Results on source code summarization, evaluated with smoothed BLEU-4 score. The baseline results are reported from Feng et al. (2020). 99

6.9 Results on text-to-code generation task using the CONCODE dataset (Iyer et al., 2018). 100

6.10 Results on source code translation using Java and C# language dataset introduced in (Lu et al., 2021). PBSMT refers to phrase-based statistical machine translation where the default settings of Moses decoder (Koehn et al., 2007) is used. The training data is tokenized using the RoBERTa (Liu et al., 2019c) tokenizer. 102

6.11 Results on program repair (in Java). 105

6.12 Results on the vulnerable code detection (accuracy) and clone detection (F1 score) tasks. 105

LIST OF FIGURES

1.1 Overview of the chapters in this thesis. The single-headed arrows indicate tasks involving transfer from high-resource languages to low-resource ones. . 4

3.1 Hierarchical clustering (with the Nearest Point Algorithm) dendrogram of the languages by their word-ordering vectors. 17

3.2 Evaluation score differences between Order-Free (OF) and Order Sensitive (OS) modules. We show results of both encoder (blue solid curve) and decoder (dashed red curve). Languages are sorted by their word-ordering distances to English from left to right. The position of English is marked with a green bar. 26

3.3 Analysis on specific dependency types. To save space, we merge the curves of encoders and decoders into one figure. The blue and red curves and left y-axis represent the differences in evaluation scores, the brown curve and right y-axis represents the relative frequency of left-direction (modifier before head) on this type. The languages (x-axis) are sorted by this relative frequency from high to low. 27

3.4 Evaluation differences of models on d=1 dependencies. Annotations are the same as in Figure 3.3, languages are sorted by percentages (represented by the brown curve and right y-axis) of d=1 dependencies. 28

3.5 Transfer performance of all source-target language pairs. The blue and red curves show the averages over columns and over rows of the source-target pair performance matrix (see text for details). The brown curve and the right y-axis legend represent the average language distance between one language and all others. 30

4.1 A relation (red dashed) between two entities and an event of type *Attack* (triggered by "firing") including two arguments and their role labels (blue) are highlighted. 34

4.2 Distance matrix showing the shortest path distances between all pairs of words. The dependency arc direction is ignored while computing pairwise distances. The diagonal value is set to 1, indicating a self-loop. If we set the values in white cells (with value > 1) to 0, the distance matrix becomes an adjacency matrix. 39

4.3 Models trained on the Chinese language perform on event argument role labeling in English and their parallel Chinese sentences. The parallel sentences have the same meaning but a different structure. To quantify the structural difference between two parallel sentences, we compute the tree edit distances. 49

4.4 Event argument role labeling confusion matrix (on test set) based on our proposed approach **GATE** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts. . 54

4.5 Event argument role labeling confusion matrix (on test set) based on the **Self-Attention (Transformer Encoder)** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts. 55

4.6 Relation extraction labeling confusion matrix (on test set) based on our proposed approach **GATE** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts. . 56

4.7 Relation extraction confusion matrix (on test set) based on the **Self-Attention (Transformer Encoder)** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts. . 57

5.1 Two parallel sentences in English and Hindi from XNLI (Conneau et al., 2018) dataset. The words highlighted with the same color have the same meaning. Although the sentences have a different word order, their syntactic dependency structure is similar. 59

5.2 A parallel QA example in English (en) and Spanish (es) from MLQA Lewis et al. (2020b) with predictions from mBERT and our proposed syntax-augmented mBERT. In "Q:x-C:y", x and y indicates question and context languages, respectively. Based on our analysis of the highlighted tokens' attention weights, we conjecture that mBERT answers 630 as the token is followed by "miembros", while 315 is followed by "senadores" in Spanish. 60

5.3 A simplified illustration of the multi-head self-attention in the graph attention network wherein each head attention is allowed between words within δ distance from each other in the dependency graph. For example, as shown, in one of the attention heads, the word "likes" is only allowed to attend its adjacent (δ=1) words "dog" and "play". 63

5.4 Performance improvements for XNLI, Wikiann, MLQA, and mATIS++ across languages.The languages in x-axis are grouped by language families: IE.Germanic (nl, de), IE.Romance (pt, fr, es), IE.Slavic (ru, bg), IE.Greek (el), IE.Indic (hi, ur), Afro-asiatic (ar, vi), Altaic (tr), Sino-tibetan (zh), Korean (ko), and Japanese (ja). 72

6.1 An overview of our experimental model consists of three basic components: (1) Encoder, (2) (Parsing) Decoder, and (3) (Language) Classifier. We also show how parsing and adversarial losses (L_p and L_d) are back propagated for parameter updates. 79

6.2 Example motivating the need to understand the association of program and natural languages for code summarization, generation, and translation. . . . 90

6.3 An example of generated code by PLBART that is syntactically and semantically valid, but does not match the reference. 101

6.4 Example C# code generated by PLBART that does not exactly match the reference code. 104

7.1 Summary of contributions made in the dissertation. Different chapters demonstrated the challenges in cross-lingual representation learning due to word order differences across languages (Chapter 3), how universal dependencies can be utilized to enhance representation learning (Chapter 4) and pre-trained multilingual encoders (Chapter 5) for cross-lingual transfer, and how such representations can be learned or improved by using unlabeled data (Chapter 6). 108

ACKNOWLEDGMENTS

I am fortunate that I got the opportunity to work with a great advisor like Kai-Wei Chang. He is one of the kindest and most considerate person I know. Without his continuous support, guidance, and insights, most of the work in this thesis would not have come into existence. His advising style of giving freedom to work on a research problem and being patient with my failures has shaped my thinking and approach towards research. I am expressing my deepest gratitude to Kai-Wei for all that he has done for me. I am also indebted to my thesis committee — Junghoo, Yizhou, and Guy.

I am extremely grateful to my mentors Nanyun Peng and Hongning Wang for guiding me through different research works. The knowledge I acquired while working with them is invaluable and will be an asset for the rest of my research life. I must also thank Zhisong Zhang, Xuezhe Ma for their collaboration on my first research project in cross-lingual NLP, which later became the foundation of my dissertation research. I very much appreciate Yuan Tian and Baishakhi Ray for their support in my research works.

During my stay at UVA and UCLA, I developed connections with extraordinary fellow students. I got the opportunity to collaborate with Md Masudur Rahman, Puxuan Yu, Xueying Bai, Zhechao Huang, Chao Jiang, Jianfeng Chi, Dat Duong, Md Rizwan Parvez, Kuan-Hao Huang, and Saikat Chakraborty. They are phenomenal in their work and were immensely helpful and supportive while pursuing new ideas.

Many thanks to Greg Favinger, Colin Morse, Nikos Karampatziakis, Xiao Bai, Soomin Lee, Haoran Li, and Yashar Mehdad, who were my mentors during my internships at Walmart Labs, Microsoft Research, Yahoo Research, and Facebook AI. They have provided me a comfortable environment to start with, helped me define the problem, guided me towards a solution while keeping me aware of practical considerations. Now, when I introspect, I realize how much I have learned from all of them over the years.

I want to thank the members of the UCLA NLP group, particularly for their support in reviewing my research works and providing feedback when needed. Thanks should also go to the support staff who ensure things run smoothly in our group at UVA and UCLA.

Lastly, I would like to thank my mother and everyone in my family whose efforts and sacrifices brought me here. A special thanks to Nishat for becoming my lifelong companion during the journey. Thanks for your unconditional love and support when things were not going well and for uplifting my spirits when facing the despair that naturally accompanies the research process. This is for you.

VITA

2013	B.S. Computer Science & Engineering Bangladesh University of Engineering & Technology, Dhaka, Bangladesh
2013	Software Development Engineer REVE Systems, Dhaka, Bangladesh
2013–2015	Lecturer of Computer Science & Engineering Ahsanullah University of Science & Technology, Dhaka, Bangladesh
2015–2016	Teaching Assistant, Computer Science Department University of Virginia, Charlottesville, Virginia, USA
2016	Research Intern Walmart Labs, Reston, Virginia, USA
2016–2017	Research Assistant, UVA Natural Language Processing Group University of Virginia, Charlottesville, Virginia, USA
2017	M.S. Computer Science University of Virginia, Charlottesville, Virginia, USA
2018, 2019	Teaching Assistant, Computer Science Department University of California, Los Angeles, California, USA
2018	Research Intern Microsoft AI and Research, Redmond, Washington, USA
2019	Research Intern Yahoo Research, Sunnyvale, California, USA
2020	Research Intern Facebook AI, Menlo Park, California, USA
2017–2021	Research Assistant, UCLA Natural Language Processing Group University of California, Los Angeles, California, USA

PUBLICATIONS

Ahmad, W. U., Li, H., Chang, K. W., Mehdad, Y. (2021). "Syntax-augmented Multilingual BERT for Cross-lingual Transfer." In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics.*

Ahmad, W. U., Chi, J., Le, T., Norton, T., Tian, Y., Chang, K. W. (2021). "Intent Classification and Slot Filling for Privacy Policies." In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics.*

Ahmad, W. U., Bai, X., Lee, S., & Chang, K. W. (2021). "Select, Extract and Generate: Neural Keyphrase Generation with Layer-wise Coverage Attention." In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics.*

Ahmad, W. U., Chakraborty, S., Ray, B., Chang, K. W. (2021). "Unified Pre-training for Program Understanding and Generation." In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

Ahmad, W. U., Peng, N., Chang, K. W. (2021). "GATE: Graph Attention Transformer Encoder for Cross-lingual Relation and Event Extraction." In *Proceedings of the 35th AAAI Conference on Artificial Intelligence.*

Chakraborty, S., Tafseer, M. T., Ahmad, W. U. (2021). "Simple or Complex? Learning to Predict Readability of Bengali Texts." In *Proceedings of the 35th AAAI Conference on Artificial Intelligence.*

Ahmad, W. U., Chi, J., Tian, Y., Chang, K. W. (2020). "PolicyQA: A Reading Comprehension Dataset for Privacy Policies." In *Findings of the Association for Computational Linguistics: EMNLP.*

Ahmad, W. U., Chakraborty, S., Ray, B., Chang, K. W. (2020). "A Transformer-based Approach for Source Code Summarization." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.*

Ahmad, W. U., Zhang, Z., Ma, X., Chang, K. W., Peng, N. (2019). "Cross-lingual Dependency Parsing with Unlabeled Auxiliary Languages." In *Proceedings of the 23rd Conference on Computational Natural Language Learning.*

Ahmad, W. U., Chang, K. W., Wang, H. (2019). "Context Attentive Document Ranking and Query Suggestion." In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Ahmad, W. U., Zhang, Z., Ma, X., Hovy, E., Chang, K. W., Peng, N. (2019). "On Difficulties of Cross-Lingual Transfer with Order Differences: A Case Study on Dependency Parsing." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

Duong, D., Ahmad, W. U., Eskin, E., Chang, K. W., Li, J. J. (2019). "Word and sentence embedding tools to measure semantic similarity of Gene Ontology terms by their definitions." *Journal of Computational Biology.*

Ahmad, W. U., Chang, K. W., Wang, H. (2018). "Intent-aware query obfuscation for privacy protection in personalized web search." In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Yu, P., Ahmad, W. U., Wang, H. (2018). "Hide-n-Seek: An Intent-aware Privacy Protection Plugin for Personalized Web Search." In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Ahmad, W. U., Chang, K. W., Wang, H. (2018). "Multi-task learning for document ranking and query suggestion." In *Proceedings of the 6th International Conference on Learning Representations.*

Ahmad, W. U., Chang, K. W. (2018). "A Corpus to Learn Refer-to-as Relations for Nominals." In *Proceedings of the 11th International Conference on Language Resources and Evaluation.*

Ahmad, W. U., Rahman, M. M., Wang, H. (2016). "Topic model based privacy protection in personalized Web search." In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval.*

CHAPTER 1

Introduction

1.1 Overview

In today's world, the number of speakers of some languages is in billions, while it is only a few thousand for many languages. Due to this difference in the number of speakers, the languages offer resources, such as a collection of training data on different scales. Advancements of deep neural network models have facilitated a wide range of natural language processing (NLP) applications in recent years. However, training these deep neural models require large amounts of annotated data, and its advantage over traditional statistical methods typically diminishes when such data is not available. Many successful stories in NLP credited its' success to the availability of large-scale training data. As a result, we have witnessed an increasing attempts to annotate large-scale datasets to facilitate NLP applications such as question answering, text summarization, conversational AI etc. Majority of these datasets are annotated by trained human workers and collected from various sources such as Wikipedia, news articles, online forums, general web, etc. Unfortunately, these annotations only exists in a handful of high-resource languages such as English. Annotating data for a wide range of languages is expensive and requires expert annotators. As a result, we have access to no or very limited amount of data to train models for languages such as Hindi, Arabic and we call them as low-resource languages.

Why should we care about low-resource languages? Although low-resource languages lack resources, a significant fraction of the world's population uses them in their day-to-day lives. For example, although Swahili is considered a low-resource language,

about 16 million people speak Swahili as a native language, and 82 million uses it as a second language[1]. Moreover, people from the same country often speak different languages; e.g., 1.2 billion people of India speak 460 languages. For example, a user asking a question to a digital assistant in Tamil, and the answer may be available in a document written in English. Therefore, to allow people to access information about nation, culture, events and communicate the consumed information with others, there is no alternative to enabling NLP technology to operate multilingually. Understanding multiple languages enables an NLP system to extract and process information available in many languages, facilitating information dissemination around the globe. Faster dissemination of information is sometimes critical, such as a Facebook user getting informed about a Hurricane taking place in a nearby area from a post written in his/her non-native language.

Cross-lingual Representation Learning Traditional supervised machine learning approaches form the backbone of current NLP technology. However, they are inherently ill-equipped to deal with the lack of labeled data, which poses a significant challenge in scaling to low-resource languages. To battle the unavailability of sufficient labeled data for low-resource NLP, researchers are delving into cross-lingual representation learning techniques. Cross-lingual representation learning can be viewed as an instance of transfer learning.

In *transfer learning*, the knowledge gained while solving one problem is applied to a different but related problem. In the context of deep learning, we can define transfer learning as reusing a model (or its components in part) that is trained on the source tasks as the starting point of a model for the target tasks (mostly with fewer examples). The model that is trained on the source tasks known as a *pre-trained* model and the process of further utilizing it for the target tasks is known as *fine-tuning*.

In NLP, a common way of transferring knowledge is through representations learned for words or sentences. Transferring lexical knowledge across languages is crucial as it enables us to compare the meaning of words across languages. This leads to cross-lingual

[1]https://www.babbel.com/en/magazine/how-many-people-speak-swahili

word representation models that aim to learn a joint embedding space. Such cross-lingual representations facilitate natural language understanding in multilingual contexts and benefits low-resource NLP.

One of the fundamental goals of cross-lingual representation learning is to learn language-agnostic representations to be transferred across languages. Encoding language-specific features may hinder cross-lingual transfer if the source and target languages differ in linguistic typology and semantics. For example, in English, Verb precedes Object, while in Hindi, Verb follows Object. Presumably, models capturing English word order will not transfer effectively to Hindi. In contrary, for particular NLP applications, such as dependency parsing, the knowledge of word order typology is important. Therefore, depending on the target languages and the downstream NLP tasks, adapting cross-lingual representations is a key for successful knowledge transfer.

1.2 Thesis Statement

Cross-lingual representation learning has emerged as an effective way to avail NLP systems in low-resource languages, such as Hindi, Bengali. However, languages differ in morphology, syntax, and semantics, which makes cross-lingual representation learning difficult. This thesis argues that encoding *universal* structural (grammatical, lexical) properties of languages into cross-lingual representations makes them language-agnostic. Adapting such language-agnostic representations in multilingual NLP systems improves the transferability of such systems to languages that lack annotated resources.

1.3 Outline of This Thesis

The rest of this document is organized as follows.

Chapter 2 presents a brief history of different approaches for learning word representations and their extensions to multilingual word representations learning. Then, we describe the use of modern deep neural networks in learning natural language representations and

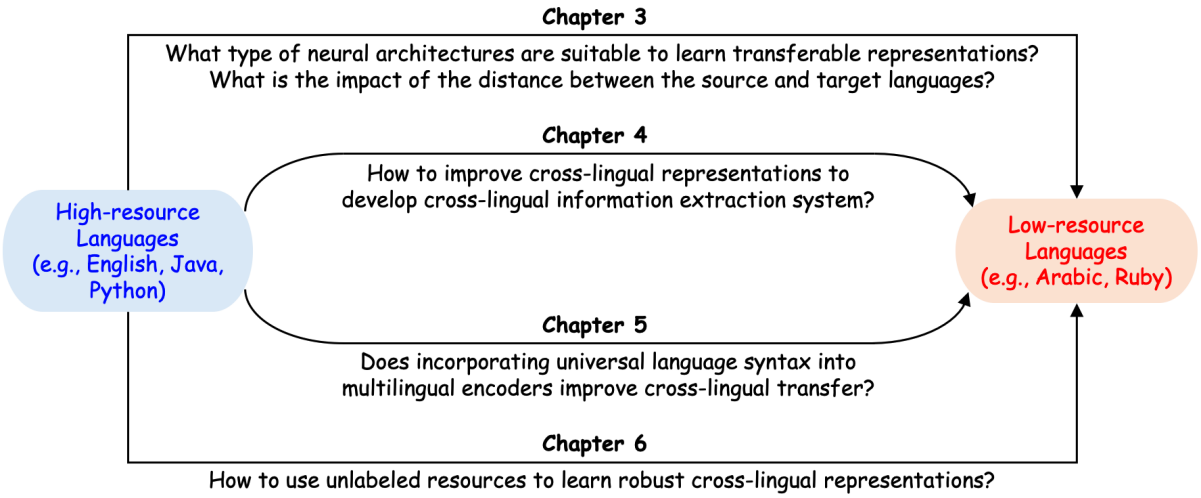


Figure 1.1: Overview of the chapters in this thesis. The single-headed arrows indicate tasks involving transfer from high-resource languages to low-resource ones.

what type of language resources are used to train them to capture cross-lingual semantics.

Chapter 3 introduces our work (Ahmad et al., 2019a) on studying the suitability of using the two preeminent neural architectures, recurrent neural networks (RNNs) (Hochreiter and Schmidhuber, 1997) and Transformers (Vaswani et al., 2017) for cross-lingual transfer learning. We then describe the effects of positional encodings in Transformers and derive a positional encoding scheme that improves Transformers' cross-lingual transferability.

Chapter 4 shows how using the universal dependency structure in learning contextual representations improves two cross-lingual information extraction (IE) tasks, (1) relation extraction and (2) event argument role labeling. Based on our work (Ahmad et al., 2021c), the chapter demonstrates that syntactic distances between entities and their arguments can characterize their relations, facilitating cross-lingual IE tasks.

Chapter 5 introduces our work (Ahmad et al., 2021b) in augmenting multilingual encoder, mBERT (Devlin et al., 2019) with universal language structure. In particular, we show that encouraging mBERT to encode the dependency structure of the input sequences while fine-tuning on downstream tasks improves cross-lingual transfer. Notably, *generalized* cross-lingual transfer improves significantly due to the supervision from linguistic structure knowledge.

Chapter 6 describes how we can exploit unlabeled monolingual resources to learn and improve the robustness of cross-lingual representations. Our work (Ahmad et al., 2019b) uses an adversarial training framework to improve mBERT on cross-lingual dependency parsing. In a recent work Ahmad et al. (2021a), we utilize monolingual resources of natural language English and programming languages, Java and Python to jointly learn multilingual representations that facilitates low-resource applications.

Finally, Chapter 7 summarizes the contributions of this thesis and provides an overview of the future directions.

CHAPTER 2

Background: Word Representation Learning

2.1 Introduction

An NLP model can be viewed as a function that takes the text data representation (or features) and makes predictions. Thus, NLP models' success largely depends on how the text data is converted into feature representations. Such feature representations are mathematical representations of the linguistic structures and are crucial for the NLP models' generalizability. Feature representations are typically learned for smaller linguistic units such as words, and representations for larger linguistic structures such as sentences, paragraphs, or documents are obtainable from word representations. Learning shared feature representations across languages is the base of cross-lingual NLP.

In this chapter, we discuss techniques for learning *monolingual* and *cross-lingual* word representations that serve as the basis for cross-lingual representation learning approaches introduced in the rest of this thesis. The history of word representations started from *one-hot representation* that represents a word as an independent categorical feature. Due to the limitations of one-shot representation, the notion of distributed word representations emerged. Distributed word representations (also known as word embeddings) represent a low-dimensional real-valued vector space that captures syntactic and semantic relationships between words. In this chapter, we limit our discussion to distributed word representation learning approaches.

Distributed Word Representation The motivation of distributed word representation is to capture the relatedness among words. Naturally, humans infer the meaning

of a word from the context in which the word appears. For example, the meaning of the word "delicious" in the sentence "The noodle dish was so delicious that I ordered it again." can be guessed based on the neighboring words (defined as context). Based on the context, humans may also guess words, such as "tasty" or "mouthwatering" since they are similar to the word under consideration. These observations form the basis of distributional hypothesis (Harris, 1954): words occurring in similar contexts share similar meaning. Word co-occurrence statistics can be computed using unlabeled text data and therefore are widely utilized to learn distributed word representations.

In literature, there are many successful approaches proposed to learn distributed word representations (Bengio et al., 2003; Collobert and Weston, 2008; Turian et al., 2010; Collobert et al., 2011; Mikolov et al., 2013a; Pennington et al., 2014; Bojanowski et al., 2017a). In the following section, we discuss a few popular methods to learn monolingual word representations that are widely used in modern NLP models.

2.2 Monolingual Word Representations

Monolingual word representations are learned from large unlabeled text corpora based on their usage in a *language*. These word representations represent words of a language as real-valued vectors, points in a n-dimensional vector space, and their geometric proximity defines the semantic similarity among words. For example, related words *king* and *queen* are closer than *king* and *mother*. Since these representations embed a word in a geometric space, they are also called word embeddings. Modern word embedding learning methods are based on neural language modeling.

Neural Language Modeling Language modeling task is defined as predicting the next word given a sequence of preceding words. In neural language modeling, a neural network takes word representations of a sequence of preceding words and outputs a probability distribution over the vocabulary for the next word prediction. An embedding matrix (where each row represents a word in a n-dimensional vector space) is used to convert

the sequence of words into a sequence of vectors. The embedding matrix and the neural network parameters are optimized using gradient descent and back-propagation.

Among the earlier approaches, Bengio et al. (2003) used a feed-forward layer to generate contextual representation of a *fixed* number of preceding words to predict the probability of the next word. Collobert and Weston (2008); Collobert et al. (2011) improved context word representation learning using convolutional neural network (LeCun et al., 1998). Later, a recurrent neural network (Elman, 1990) has shown to capture arbitrary long past context improving language model (Mikolov et al., 2010).

In traditional language modeling, preceding words (i.e., context to the left) are used to predict the next word. This is known as *left-to-right* language modeling. (Mikolov et al., 2013a) proposed to utilize both left and right context around a word (preceding and following words) for language modeling. The authors proposed the *continuous bag-of-words* (CBOW) model where a classifier is trained to predict a central (or *pivot*) word based on its left and right context and the word representations are learnt as a by-product. This paradigm of language modeling is also known as *bidirectional* language modeling. As an alternative, the authors also proposed the *skip-gram* model that follows the opposite strategy, learnt to predict the left and right context (neighboring words) given the pivot word. The skip-gram modeling became a popular choice to learn embeddings since it works well even with a small amount of training data. The word embeddings learnt via skip-gram modeling is popularly known as *Word2vec*.

While Word2vec leverages co-occurrence statistics of words within local context (neighboring words), Pennington et al. (2014) proposed to learn Global Vector Representations (GloVe) by leveraging word co-occurrence statistics across the entire corpus. GloVe applies a matrix factorization technique on a pre-computed word-context matrix. The word-context matrix records how frequently a "word" (the rows) is seen in some "context" (the columns) in a large corpus. By applying a matrix factorization technique, GloVe learns to find a low-dimensional word embeddings matrix that can explain most of the variance in the word-context matrix. In literature, both Word2vec and GloVe are found to be effective. However, there are two limitations of Word2vec and GloVe embeddings.

First, the embeddings of rare words are comparatively poorer than frequent words (a rare word has fewer neighbors), and second, those two techniques cannot learn an embedding for words that do not appear in the training corpus.

Bojanowski et al. (2017a) extended the skip-gram model and proposed fastText that solves the above two limitations by treating each word as a bag of character n-grams. According to the proposed method, fastText learns vector representation for each character n-gram, and the words are represented as the sum of their constitute character n-grams' representations. In literature, fastText embeddings are found to be more effective than Word2vec and GloVe embeddings.

The word embeddings learning approaches discussed so far provide a single vector representation for each word. Therefore, the word representation for polysemous words, such as *bank* requires capturing all relevant meaning representations (Arora et al., 2018). Several works (Reisinger and Mooney, 2010; Huang et al., 2012) proposed to learn a fixed number (more than one) of representations per word. However, all these approaches overlooked the fact that the meaning of a word depends on in what context they appear. Intuitively, contextual information (neighboring words) indicates the specific meaning of a polysemous word appearing in a context. For example, the word *bank* appearing in the sentences "The bank is not offering a good interest rate" and "He ran forward to the river bank" means a financial institute and the bank of a river, respectively. Most NLP applications deal with text inputs that are either sentences, paragraphs or documents. Therefore, learning contextual word representations using monolingual corpora and utilizing them in modern deep neural NLP models have become the de facto standard in recent years.

2.2.1 Contextualized Word Representations

As noted earlier that NLP models can be viewed as functions that take the text data representation (or features) and makes predictions. Modern neural NLP models are typically composed of a representation learning component, also known as *encoder*, and

a task-specific neural network component. While the encoder converts the input word sequence into a sequence of fixed-size vectors, the task-specific component takes the encoder's output vector representations and predicts the task-specific output. The fundamental idea of learning contextual word representations is to train such a representation learning encoder. Unlike word embedding learning approaches that learn single vector representations for words, the encoder generates vector representations for words depending on what context they appear in (e.g., sentences or paragraphs).

Contextualized word representation learning became very popular due to its effectiveness in facilitating NLP with fewer amounts of labeled data. In general, the representation learning encoder in deep neural NLP models is realized by a high complexity neural network architecture and requires a large amount of data to train. In comparison, the task-specific neural network component is simpler (e.g., a linear classifier for the text classification task), requiring less data for training. When there are abundant training examples available for an NLP task, the encoder, and the task-specific component can be jointly trained from scratch in an end-to-end fashion. However, when data is insufficient, this approach is unfeasible. Instead, we can *pre-train* the encoder on other tasks (a.k.a, source tasks) and *transfer* the learned encoder to the target task. As a result, a low-complexity task-specific component on top of the pre-trained encoder can be trained to perform the target task with a few labeled examples. This paradigm in the NLP literature is known as *transfer learning*.

In this line of work, McCann et al. (2017) first proposed to learn contextualize word vectors by using an LSTM (Hochreiter and Schmidhuber, 1997) encoder that was a part of a sequence-to-sequence model trained for machine translation task. The authors showed leveraging the trained encoder in a wide variety of text classification and question answering tasks. As a result, NLP researchers delved into learning contextual representations of words by pre-training deep neural encoders on a humongous amount of unlabeled text data using language modeling objectives and achieved notable success. The pre-trained encoders are used as feature extractors that produce contextual word vectors and are utilized in NLP models or directly fine-tuned in a downstream NLP task. ELMo (Peters

et al., 2018b), GPT (Radford et al., 2018), BERT (Devlin et al., 2018) are some of those noteworthy pre-trained language models and many of their variants such as SciBERT (Beltagy et al., 2019), ClinicalBERT (Alsentzer et al., 2019) has facilitated NLP for low-resource domains and tasks.

2.3 Cross-lingual Word Representations

Monolingual word embeddings are trained using sizeable unlabeled text corpora in each language *independently*. The vector spaces learned by the monolingual word embeddings do not capture semantic relationships between words across languages since they are trained solely using monolingual distributional information. Therefore, pre-trained word representations as features in NLP models are confined to operate in only one language. As a result, NLP models trained using task-specific supervision in one language cannot be utilized in related languages. If trained to perform a task, e.g., question answering in one language, human beings aware of multiple languages would ideally be able to perform the task in other languages they know. Having such capabilities in NLP models minimizes the need for task-specific supervision in every language, facilitating NLP in a broad spectrum of human languages, including low-resource languages.

2.3.1 Resources for Learning Cross-lingual Representations

The fundamental idea of cross-lingual word embedding learning is to project word vector representations from two or more languages into a single vector space. As a result, words with similar meanings are represented as points in the shared vector space that are geometrically closer to each other irrespective of their languages. Projection of word vector representations of multiple languages into a shared space is generally learned leveraging cross-lingual supervision from bilingual dictionaries (Klementiev et al., 2012; Mikolov et al., 2013b) or parallel corpora (Zou et al., 2013; Gouws et al., 2015). Later, a few proposed techniques alleviated the requirement of such cross-lingual supervision and only required non-parallel document-aligned data (Vulić and Moens, 2015a).

2.3.2 Techniques for Cross-lingual Representations Learning

Cluster-based Approaches The basic idea of cluster-based cross-lingual representation learning is to form clusters containing words in two or more languages that share similar linguistic properties. Täckström et al. (2012a) proposed a two-stage approach for learning such representations. In the first stage, words in one language (e.g., source language) are clustered monolingually, and in the second stage, the monolingual word clusters are projected to the target language. Each word in the source language clusters is assigned according to how often the word is aligned to the target cluster words based on word alignments from parallel corpora. To tackle words that do not appear in the alignment dictionary, Täckström et al. (2012a) proposed to jointly optimize the monolingual clustering objective in each language, followed by the cluster projection step.

Vector-based Approaches The word embeddings-based approaches that learn a shared representation space fall under this category. These approaches use different forms of cross-lingual alignment supervision to align the monolingual vector spaces. Majority of the prior works utilize cross-lingual supervision from sentence and word-level alignments (Klementiev et al., 2012; Zou et al., 2013; Kočiský et al., 2014; Luong et al., 2015a) or bilingual dictionaries (Mikolov et al., 2013b; Faruqui and Dyer, 2014; Lu et al., 2015; Smith et al., 2017; Artetxe et al., 2017). Word level alignments are primarily derived from parallel sentence corpora using statistical aligner, e.g., IBM Model 1 aligner (Brown et al., 1993), the fast-align (Dyer et al., 2013). Collecting parallel corpora for a wide spectrum of languages is expensive. In contrast, bilingual word dictionaries with a few thousand words are much easier to obtain and motivate a large pool of prior works. The underlying notion is to learn a shared vector space such that equivalent word pairs in the bilingual dictionary get similar representations.

2.3.3 Multilingual Word Representations

Cross-lingual word representation learning approaches discussed above primarily learn *bilingual* embeddings. In contrast, multilingual word embeddings are trained to jointly encode words in multiple languages (more than two) in the same vector space such that semantically similar words across the languages remain geometrically closer (Ammar et al., 2016b; Smith et al., 2017; Duong et al., 2017). Ruder et al. (2019) surveyed the existing research works on cross-lingual word embedding induction. Please refer to the survey for more detailed coverage of the works.

Contextualized Word Representations The recent development of pre-trained languages models that work as contextual word representation encoders (Devlin et al., 2018; Liu et al., 2019c; Yang et al., 2019b; Lewis et al., 2020a) has also opened up the opportunity for learning contextual representations by jointly pre-training on humongous amount of unlabeled text corpora in many languages (Devlin et al., 2018; Lample and Conneau, 2019; Conneau et al., 2019; Liu et al., 2020). These multilingual encoders learn a shared multilingual contextual embedding space; they can represent word pairs in parallel sentences with similar contextual representations. While these encoders have significantly improved cross-lingual transfer learning, they still suffer from various issues, e.g., ignore capturing universal language syntax information resulting in poor performance as discussed in Chapter 5 of this thesis.

CHAPTER 3

Cross-Lingual Transfer with Order Differences

3.1 Introduction

Cross-lingual transfer, which transfers models across languages, has tremendous practical value. It reduces the requirement of annotated data for a target language and is especially useful when the target language is lack of resources. Recently, this technique has been applied to many NLP tasks such as text categorization (Zhou et al., 2016a), tagging (Kim et al., 2017), dependency parsing (Guo et al., 2015, 2016) and machine translation (Zoph et al., 2016). Despite the preliminary success, transferring across languages is challenging as it requires understanding and handling differences between languages at levels of morphology, syntax, and semantics. It is especially difficult to learn invariant features that can robustly transfer to distant languages.

Prior work on cross-lingual transfer mainly focused on sharing word-level information by leveraging multi-lingual word embeddings (Xiao and Guo, 2014; Guo et al., 2016; Sil et al., 2018). However, words are not independent in sentences; their combinations form larger linguistic units, known as *context*. Encoding context information is vital for many NLP tasks, and a variety of approaches (e.g., convolutional neural networks and recurrent neural networks) have been proposed to encode context as a high-level feature for downstream tasks. In this paper, we study how to transfer generic contextual information across languages.

For cross-language transfer, one of the key challenges is the variation in word order among different languages. For example, the Verb-Object pattern in English can hardly be found in Japanese. This challenge should be taken into consideration in model design.

RNN is a prevalent family of models for many NLP tasks and has demonstrated compelling performances (Mikolov et al., 2010; Sutskever et al., 2014; Peters et al., 2018a). However, its sequential nature makes it heavily reliant on word order information, which exposes to the risk of encoding language-specific order information that cannot generalize across languages. We characterize this as the "*order-sensitive*" property. Another family of models known as "Transformer" uses self-attention mechanisms to capture context and was shown to be effective in various NLP tasks (Vaswani et al., 2017; Liu et al., 2018b; Kitaev and Klein, 2018). With modification in position representations, the self-attention mechanism can be more robust than RNNs to the change of word order. We refer to this as the "*order-free*" property.

In this work, we posit that *order-free* models have better transferability than *order-sensitive* models because they less suffer from overfitting language-specific word order features. To test our hypothesis, we first quantify language distance in terms of word order typology, and then systematically study the transferability of order-sensitive and order-free neural architectures on cross-lingual dependency parsing.

We use dependency parsing as a test bed primarily because of the availability of unified annotations across a broad spectrum of languages (Nivre et al., 2018). Besides, word order typology is found to influence dependency parsing (Naseem et al., 2012; Täckström et al., 2013; Zhang and Barzilay, 2015; Ammar et al., 2016a; Aufrant et al., 2016). Moreover, parsing is a low-level NLP task (Hashimoto et al., 2017) that can benefit many downstream applications (McClosky et al., 2011; Gamallo et al., 2012; Jie et al., 2017).

We conduct evaluations on 31 languages across a broad spectrum of language families, as shown in Table 6.1. Our empirical results show that *order-free* encoding and decoding models generally perform better than the *order-sensitive* ones for cross-lingual transfer, especially when the source and target languages are distant.

Language Families	Languages
Afro-Asiatic	Arabic (ar), Hebrew (he)
Austronesian	Indonesian (id)
IE.Baltic	Latvian (lv)
IE.Germanic	Danish (da), Dutch (nl), English (en), German (de), Norwegian (no), Swedish (sv)
IE.Indic	Hindi (hi)
IE.Latin	Latin (la)
IE.Romance	Catalan (ca), French (fr), Italian (it), Portuguese (pt), Romanian (ro), Spanish (es)
IE.Slavic	Bulgarian (bg), Croatian (hr), Czech (cs), Polish (pl), Russian (ru), Slovak (sk), Slovenian (sl), Ukrainian (uk)
Japanese	Japanese (ja)
Korean	Korean (ko)
Sino-Tibetan	Chinese (zh)
Uralic	Estonian (et), Finnish (fi)

Table 3.1: The selected languages grouped by language families. "IE" is the abbreviation of Indo-European.

3.2 Quantifying Language Distance

We first verify that we can measure "language distance" base on word order since it is a significant distinctive feature to differentiate languages (Dryer, 2007). The World Atlas of Language Structures (WALS) (Dryer and Haspelmath, 2013a) provides a great reference for word order typology and can be used to construct feature vectors for languages (Littell et al., 2017). But since we already have the universal dependency annotations, we take an empirical way and directly extract word order features using directed dependency relations (Liu, 2010).

We conduct our study using the Universal Dependencies (UD) Treebanks (v2.2) (Nivre et al., 2018). We select 31 languages for evaluation and analysis, with the selection criterion being that the total token number in the treebanks of that language is over 100K. We group these languages by their language families in Table 6.1. Detailed statistical information of the selected languages and treebanks can be found in Appendix A[1].

[1]Please refer to the supplementary materials for all the appendices of this paper.

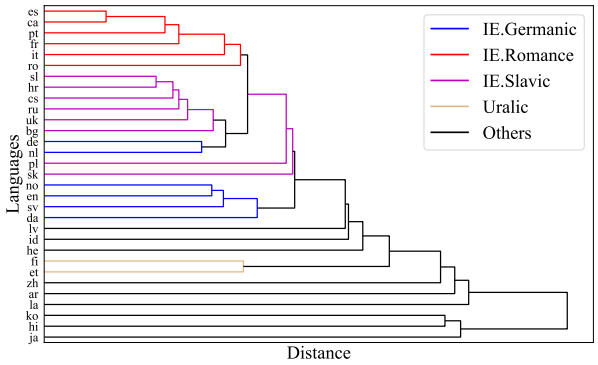


Figure 3.1: Hierarchical clustering (with the Nearest Point Algorithm) dendrogram of the languages by their word-ordering vectors.

We look at finer-grained dependency types than the 37 universal dependency labels[2] in UD v2 by augmenting the dependency labels with the universal part-of-speech (POS) tags of the head and modifier[3] nodes. Specifically, we use triples "(ModifierPOS, HeadPOS, DependencyLabel)" as the augmented dependency types. With this, we can investigate language differences in a fine-grained way by defining directions on these triples (i.e. modifier before head or modifier after head).

We conduct feature selection by filtering out rare types as they can be unstable. We defer the results in 52 selected types and more details to Appendix C. For each dependency type, we collect the statistics of directionality (Liu, 2010; Wang and Eisner, 2017). Since there can be only two directions for an edge, for each dependency type, we use the relative frequency of the left-direction (modifier before head) as the directional feature. By concatenating the directional features of all selected triples, we obtain a word-ordering feature vector for each language. We calculate the *word-ordering distance* using these vectors. In this work, we simply use Manhattan distance, which works well as shown in our analysis (Section 3.4.3).

We perform hierarchical clustering based on the word-ordering vectors for the selected languages, following (Östling, 2015). As shown in Figure 3.1, the grouping of the ground

[2]http://universaldependencies.org/u/dep/index.html

[3]In this paper, we use the term of "modifier", which can also be described as "dependent" or "child" node.

truth language families is almost recovered. The two outliers, German (de) and Dutch (nl), are indeed different from English. For instance, German and Dutch adopt a larger portion of Object-Verb order in embedded clauses. The above analysis shows that word order is an important feature to characterize differences between languages. Therefore, it should be taken into consideration in the model design.

3.3 Models

Our primary goal is to conduct cross-lingual transfer of syntactic dependencies without providing any annotation in the target languages. The overall architecture of models that are studied in this research is described as follows. The first layer is an input embedding layer, for which we simply concatenate word and POS embeddings. The POS embeddings are trained from scratch, while the word embeddings are fixed and initialized with the multilingual embeddings by (Smith et al., 2017). These inputs are fed to the encoder to get contextual representations, which is further used by the decoder for predicting parse trees.

For the cross-lingual transfer, we hypothesize that the models capturing less language-specific information of the source language will have better transferability. We focus on the word order information, and explore different encoders and decoders that are considered as *order-sensitive* and *order-free*, respectively.

3.3.1 Contextual Encoders

Considering the sequential nature of languages, RNN is a natural choice for the encoder. However, modeling sentences word by word in the sequence inevitably encodes word order information, which may be specific to the source language. To alleviate this problem, we adopt the self-attention based encoder (Vaswani et al., 2017) for cross-lingual parsing. It can be less sensitive to word order but not necessarily less potent at capturing contextual information, which makes it suitable for our study.

RNNs Encoder Following prior work (Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017), we employ k-layer bidirectional LSTMs (Hochreiter and Schmidhuber, 1997) on top of the input vectors to obtain contextual representations. Since it explicitly depends on word order, we will refer it as an *order-sensitive* encoder.

Self-Attention Encoder The original self-attention encoder (Transformer) takes absolute positional embeddings as inputs, which capture much order information. To mitigate this, we utilize relative position representations (Shaw et al., 2018a), with further simple modification to make it order-agnostic: the original relative position representations discriminate left and right contexts by adding signs to distances, while we discard the directional information.

We directly base our descriptions on those in (Shaw et al., 2018a). For the relative positional self-attention encoder, each layer calculates multiple attention heads. In each head, the input sequence of vectors $\mathbf{x} = (x_1, \ldots, x_n)$ are transformed into the output sequence of vectors $\mathbf{z} = (z_1, \ldots, z_n)$, based on the self-attention mechanism:

$$
\begin{aligned}
z_i &= \sum_{j=1}^{n} \alpha_{ij}(x_j W^V + a_{ij}^V) \\
\alpha_{ij} &= \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}} \\
e_{ij} &= \frac{x_i W^Q (x_j W^K + a_{ij}^K)^T}{\sqrt{d_z}}
\end{aligned}
$$

Here, a_{ij}^V and a_{ij}^K are relative positional representations for the two position i and j. Similarly, we clip the distance with a maximum threshold of k (which is empirically set to 10), but we do not discriminate positive and negative values. Instead, since we do not want the model to be aware of directional information, we use the absolute values of the position differences:

$$
a_{ij}^K = w_{clip(|j-i|,k)}^K \quad a_{ij}^V = w_{clip(|j-i|,k)}^V \quad clip(x,k) = \min(|x|,k)
$$

Therefore, the learnable relative postion representations have $k+1$ types rather than

$2k+1$: we have $w^K = (w_0^K, \ldots, w_k^K)$, and $w^V = (w_0^V, \ldots, w_k^V)$.

With this, the model knows only what words are surrounding but cannot tell the directions. Since self-attention encoder is less sensitive to word order, we refer to it as an *order-free* encoder.

3.3.2 Structured Decoders

With the contextual representations from the encoder, the decoder predicts the output tree structures. We also investigate two types of decoders with different sensitivity to ordering information.

Stack-Pointer Decoder Recently, (Ma et al., 2018) proposed a top-down transition-based decoder and obtained state-of-the-art results. Thus, we select it as our transition-based decoder. To be noted, in this Stack-Pointer decoder, RNN is utilized to record the decoding trajectory and also can be sensitive to word order. Therefore, we will refer to it as an *order-sensitive* decoder.

Graph-based Decoder Graph-based decoders assume simple factorization and can search globally for the best structure. Recently, with a deep biaffine attentional scorer, (Dozat and Manning, 2017) obtained state-of-the-art results with simple first-order factorization (Eisner, 1996; McDonald et al., 2005). This method resembles the self-attention encoder and can be regarded as a self-attention output layer. Since it does not depend on ordering information, we refer to it as an *order-free* decoder.

3.4 Experiments and Analysis

In this section, we compare four architectures for cross-lingual transfer dependency parsing with a different combination of order-free and order-sensitive encoder and decoder. We conduct several detailed analyses showing the pros and cons of both types of models.

3.4.1 Setup

Settings In our main experiments[4] (those except Section 3.4.3.5), we take English as the source language and 30 other languages as target languages. We only use the source language for both training and hyper-parameter tuning. During testing, we directly apply the trained model to target languages with the inputs from target languages passed through pretrained multilingual embeddings that are projected into a common space as the source language. The projection is done by the offline transformation method (Smith et al., 2017) with pre-trained $300d$ monolingual embeddings from FastText (Bojanowski et al., 2017b). We freeze word embeddings since fine-tuning on them may disturb the multi-lingual alignments. We also adopt gold UPOS tags for the inputs.

For other hyper-parameters, we adopted similar ones as in the Biaffine Graph Parser (Dozat and Manning, 2017) and the Stack-Pointer Parser (Ma et al., 2018). Detailed hyper-parameter settings can be found in Appendix B. Throughout our experiments, we adopted the language-independent UD labels and a sentence length threshold of 140. The evaluation metrics are Unlabeled attachment score (UAS) and labeled attachment score (LAS) with punctuations excluded[5]. We trained our cross-lingual models five times with different initialization and reported average scores.

Systems As described before, we have an *order-free* (Self-Attention) and an *order-sensitive* (BiLSTM-RNN) encoder, as well as an *order-free* (Biaffine Attention Graph-based) and an *order-sensitive* (Stack-Pointer) decoder. The combination gives us four different models, named in the format of "Encoder" plus "Decoder". For clarity, we also mark each model with their encoder-decoder order sensitivity characteristics. For example, "SelfAtt-Graph (OF-OF)" refers to the model with self-attention order-free encoder and graph-based order-free decoder. We benchmark our models with a baseline shift-reduce

[4]Our implementation is publicly available at: https://github.com/uclanlp/CrossLingualDepParser

[5]In our evaluations, we exclude tokens whose POS tags are "PUNCT" or "SYM". This setting is different from the one adopted in the CoNLL shared task (Zeman et al., 2018). However, the patterns are similar as shown in Appendix D where we report the punctuation-included test evaluations.

Lang	Dist. to English	SelfAtt-Graph (OF-OF)	RNN-Graph (OS-OF)	SelfAtt-Stack (OF-OS)	RNN-Stack (OS-OS)	Baseline (Guo et al., 2015)	Supervised (RNN-Graph)
en	0.00	90.35/88.40	90.44/88.31	90.18/88.06	**91.82**†/**89.89**†	87.25/85.04	90.44/88.31
no	0.06	80.80/72.81	80.67/72.83	80.25/72.07	**81.75**†/**73.30**†	74.76/65.16	94.52/92.88
sv	0.07	80.98/73.17	81.23/73.49	80.56/72.77	**82.57**†/**74.25**†	71.84/63.52	89.79/86.60
fr	0.09	77.87/72.78	**78.35**†/**73.46**†	76.79/71.77	75.46/70.49	73.02/64.67	91.90/89.14
pt	0.09	**76.61**†/67.75	76.46/**67.98**	75.39/66.67	74.64/66.11	70.36/60.11	93.14/90.82
da	0.10	76.64/67.87	77.36/68.81	76.39/67.48	**78.22**†/**68.83**	71.34/61.45	87.16/84.23
es	0.12	74.49/66.44	**74.92**†/**66.91**†	73.15/65.14	73.11/64.81	68.75/59.59	93.17/90.80
it	0.12	80.80/75.82	**81.10**/**76.23**†	79.13/74.16	80.35/75.32	75.06/67.37	94.21/92.38
hr	0.13	**61.91**†/**52.86**†	60.09/50.67	60.58/51.07	60.80/51.12	52.92/42.19	89.66/83.81
ca	0.13	73.83/65.13	**74.24**†/**65.57**†	72.39/63.72	72.03/63.02	68.23/58.15	93.98/91.64
pl	0.13	**74.56**†/**62.23**†	71.89/58.59	73.46/60.49	72.09/59.75	66.74/53.40	94.96/90.68
uk	0.13	**60.05**/**52.28**†	58.49/51.14	57.43/49.66	59.67/51.85	54.10/45.26	85.98/82.21
sl	0.13	**68.21**†/**56.54**†	66.27/54.57	66.55/54.58	67.76/55.68	60.86/48.06	86.79/82.76
nl	0.14	68.55/60.26	67.88/60.11	67.88/59.46	**69.55**†/**61.55**†	63.31/53.79	90.59/87.52
bg	0.14	**79.40**†/**68.21**†	78.05/66.68	78.16/66.95	78.83/67.57	73.08/61.23	93.74/89.61
ru	0.14	60.63/51.63	59.99/50.81	59.36/50.25	**60.87**/**51.96**	55.03/45.09	94.11/92.56
de	0.14	**71.34**†/**61.62**†	69.49/59.31	69.94/60.09	69.58/59.64	65.14/54.13	88.58/83.68
he	0.14	**55.29**/**48.00**†	54.55/46.93	53.23/45.69	54.89/40.95	46.03/26.57	89.34/84.49
cs	0.14	**63.10**†/**53.80**†	61.88/52.80	61.26/51.86	62.26/52.32	56.15/44.77	94.03/91.87
ro	0.15	**65.05**†/**54.10**†	63.23/52.11	62.54/51.46	60.98/49.79	56.01/44.04	90.07/84.50
sk	0.17	**66.65**/**58.15**†	65.41/56.98	65.34/56.68	66.56/57.48	57.75/47.73	90.19/86.38
id	0.17	**49.20**†/**43.52**†	47.05/42.09	47.32/41.70	46.77/41.28	40.84/33.67	87.19/82.60
lv	0.18	70.78/49.30	**71.43**†/**49.59**	69.04/47.80	70.56/48.53	62.33/41.42	83.67/78.13
fi	0.20	66.27/48.69	**66.36**/**48.74**	64.82/47.50	66.25/48.28	58.51/38.65	88.04/85.04
et	0.20	**65.72**†/**44.87**†	65.25/44.40	64.12/43.26	64.30/43.50	56.13/34.86	86.76/83.28
zh*	0.23	**42.48**†/**25.10**†	41.53/24.32	40.56/23.32	40.92/23.45	40.03/20.97	73.62/67.67
ar	0.26	**38.12**†/**28.04**†	32.97/25.48	32.56/23.70	32.85/24.99	32.69/22.68	86.17/81.83
la	0.28	**47.96**†/**35.21**†	45.96/33.91	45.49/33.19	43.85/31.25	39.08/26.17	81.05/76.33
ko	0.33	**34.48**†/**16.40**†	33.66/15.40	32.75/15.04	33.11/14.25	31.39/12.70	85.05/80.76
hi	0.40	**35.50**†/**26.52**†	29.32/21.41	31.38/23.09	25.91/18.07	25.74/16.77	95.63/92.93
ja*	0.49	**28.18**†/**20.91**†	18.41/11.99	20.72/13.19	15.16/9.32	15.39/08.41	89.06/78.74
Average	0.17	**64.06**†/**53.82**†	62.71/52.63	62.22/52.00	62.37/51.89	57.09/45.41	89.44/85.62

Table 3.2: Results (UAS%/LAS%, excluding punctuation) on the test sets. Languages are sorted by the word-ordering distance to English, as shown in the second column. '*' refers to results of delexicalized models, '†' means that the best transfer model is statistically significantly better (by paired bootstrap test, $p < 0.05$) than all other transfer models. Models are marked with their encoder and decoder order sensitivity, OF denotes order-free and OS denotes order-sensitive.

transition-based parser, which gave previous state-of-the-art results for single-source zero-resource cross-lingual parsing (Guo et al., 2015). Since they used older datasets, we re-trained the model on our datasets with their implementation[6]. We also list the supervised learning results using the "RNNGraph" model on each language as a reference of the upper-line for cross-lingual parsing.

[6]https://github.com/jiangfeng1124/acl15-clnndep. We also evaluated our models on the older dataset and compared with their results, as shown in Appendix F.

3.4.2 Results

The results on the test sets are shown in Table 6.2. The languages are ordered by their order typology distance to English. In preliminary experiments, we found our lexicalized models performed poorly on Chinese (zh) and Japanese (ja). We found the main reason was that their embeddings were not well aligned to English. Therefore, we use delexicalized models, where only POS tags are used as inputs. The delexicalized results[7] for Chinese and Japanese are listed in the rows marked with "*".

Overall, the "SelfAtt-Graph" model performs the best in over half of the languages and beats the runner-up "RNN-Graph" by around 1.3 in UAS and 1.2 in LAS on average. When compared with "RNN-Stack" and "SelfAtt-Stack", the average difference is larger than 1.5 points. This shows that models capture less word order information generally perform better at cross-lingual parsing. Compared with the baseline, our superior results show the importance of the contextual encoder. Compared with the supervised models, the cross-lingual results are still lower by a large gap, indicating space for improvements.

After taking a closer look, we find an interesting pattern in the results: while the model performances on the source language (English) are similar, RNN-based models perform better on languages that are closer to English (upper rows in the table), whereas for languages that are "distant" from English, the "SelfAtt-Graph" performs much better. Such patterns correspond well with our hypothesis, that is, the design of models considering word order information is crucial in cross-lingual transfer. We conduct more thorough analysis in the next subsection.

3.4.3 Analysis

We further analyze how different modeling choices influence cross-lingual transfer. Since we have not touched the training sets for languages other than English, in this subsection,

[7]We found delexicalized models to be better only at zh and ja, for about 5 and 10 points respectively. For other languages, they performed worse for about 2 to 5 points. We also tried models without POS, and found them worse for about 10 points on average. We leave further investigation of input representations to future work.

we evaluate and analyze the performance of target languages using training splits in UD. Performance of English is evaluated on the test set. We verify that the trends observed in test set are similar to those on the training sets. As mentioned in the previous section, the bilingual embeddings for Chinese and Japanese do not align well with English. Therefore, we report the results with delexicalizing. In the following, we discuss our observations, and detailed results are listed in Appendix E.

3.4.3.1 Encoder Architecture

We assume models that are less sensitive to word order perform better when transfer to distant languages. To empirically verify this point, we conduct controlled comparisons on various encoders with the same graph-based decoder. Table 3.3 shows the average performances in all languages.

Model	UAS%	LAS%
SelfAtt-Relative (Ours)	64.57	54.14
SelfAtt-Relative+Dir	63.93	53.62
RNN	63.25	52.94
SelfAtt-Absolute	61.76	51.71
SelfAtt-NoPosi	28.18	21.45

Table 3.3: Comparisons of different encoders (averaged results over all languages on the original training sets).

To compare models with various degrees of sensitivity to word order, we include several variations of self-attention models. The "SelfAtt-NoPosi" is the self-attention model without any positional information. Although it is most insensitive to word order, it performs poorly possibly because of the lack of access to the locality of contexts. The self-attention model with absolute positional embeddings ("SelfAtt-Absolute") also does not perform well. In the case of parsing, relative positional representations may be more useful as indicated by the improvements brought by the directional relative position representations ("SelfAtt-Relative+Dir") (Shaw et al., 2018a). Interestingly, the RNN encoder ranks between "SelfAtt-Relative+Dir" and "SelfAtt-Absolute"; all these three encoders explicitly capture word order information in some way. Finally, by discarding

the information of directions, our relative position representation ("SelfAtt-Relative") performs the best (significantly better at $p < 0.05$).

One crucial observation we have is that the patterns of breakdown performances for "SelfAtt-Relative+Dir" are similar to those of RNN: on closer languages, the direction-aware model performs better, while on distant languages the non-directional one generally obtains better results. Since the only difference between our proposed "SelfAtt-Relative" model and the "SelfAtt-Relative+Dir" model is the directional encoding, we believe the better performances should credit to its effectiveness in capturing useful context information without depending too much on the language-specific order information.

These results suggest that a model's sensitivity to word order indeed affects its cross-lingual transfer performances. In later sections, we stick to our "SelfAtt-Relative" variation of the self-attentive encoder and focus on the comparisons among the four main models.

3.4.3.2 Performance v.s. Language Distance

We posit that order-free models can do better than order-sensitive ones on cross-lingual transfer parsing when the target languages have different word orders to the source language. Now we can analyze this with the word-ordering distance.

For each target language, we collect two types of distances when comparing it to English: one is the *word-ordering distance* as described in Section 3.2, the other is the *performance distance*, which is the gap of evaluation scores[8] between the target language and English. The performance distance can represent the general transferability from English to this language. We calculate the correlation of these two distances on all the concerned languages, and the results turn to be quite high: the Pearson and Spearman correlations are *around 0.90 and 0.87* respectively, using the evaluations of any of our four cross-lingual transfer models. This suggests that word order can be an important factor of cross-lingual transferability.

[8]In the rest of this paper, we simply average UAS and LAS for evaluation scores unless otherwise noted.

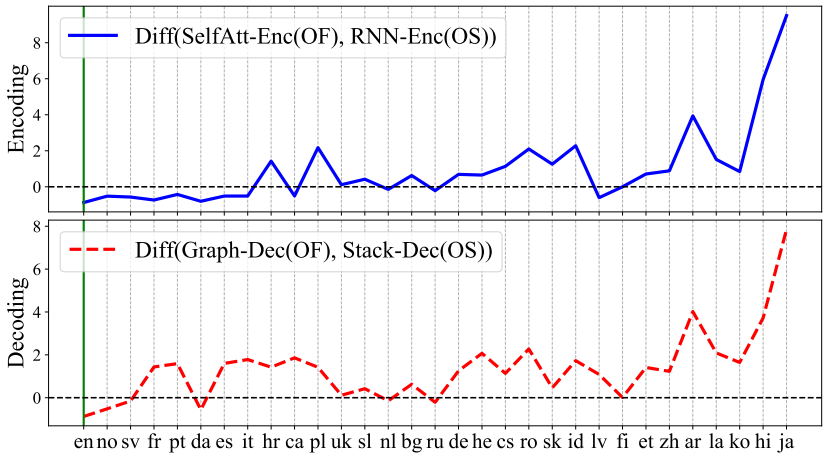
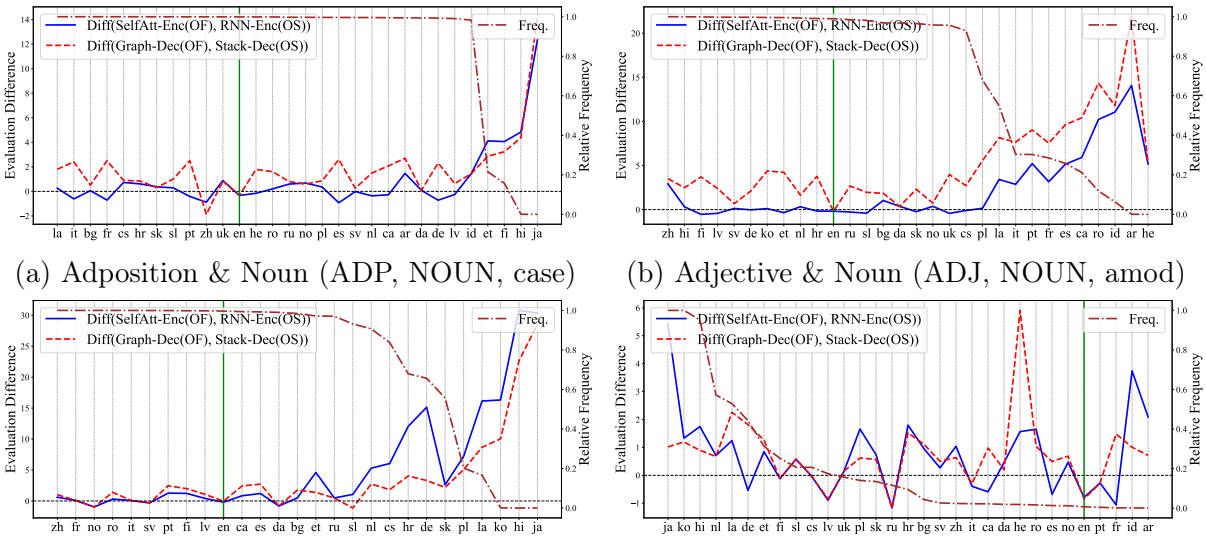


Figure 3.2: Evaluation score differences between Order-Free (OF) and Order Sensitive (OS) modules. We show results of both encoder (blue solid curve) and decoder (dashed red curve). Languages are sorted by their word-ordering distances to English from left to right. The position of English is marked with a green bar.

Furthermore, we individually analyze the encoders and decoders of the dependency parsers. Since we have two architectures for each of the modules, when examining one, we take the highest scores obtained by any of the other modules. For example, when comparing RNN and Self-Attention encoders, we take the best evaluation scores of "RNN-Graph" and "RNN-Stack" for RNN and the best of "SelfAtt-Graph" and "SelfAtt-Stack" for Self-Attention. Figure 3.2 shows the score differences of encoding and decoding architectures against the languages' distances to English. For both the encoding and decoding module, we observe a similar overall pattern: the order-free models, in general, perform better than order-sensitive ones in the languages that are distant from the source language English. On the other hand, for some languages that are closer to English, order-sensitive models perform better, possibly benefiting from being able to capture similar word ordering information. The performance gap between order-free and order-sensitive models are positively correlated with language distance.

3.4.3.3 Performance Breakdown by Types

Moreover, we compare the results on specific dependency types using concrete examples. For each type, we sort the languages by their relative frequencies of left-direction (modifier



(a) Adposition & Noun (ADP, NOUN, case)

(b) Adjective & Noun (ADJ, NOUN, amod)

(c) Auxiliary & Verb (AUX, VERB, aux)

(d) Object & Verb (NOUN, VERB, obj)

Figure 3.3: Analysis on specific dependency types. To save space, we merge the curves of encoders and decoders into one figure. The blue and red curves and left y-axis represent the differences in evaluation scores, the brown curve and right y-axis represents the relative frequency of left-direction (modifier before head) on this type. The languages (x-axis) are sorted by this relative frequency from high to low.

before head) and plot the performance differences for encoders and decoders. We highlight the source language English in green. Figure 3.3 shows four typical example types: Adposition and Noun, Adjective and Noun, Auxiliary and Verb, and Object and Verb. In Figure 3.3a, we examine the "case" dependency type between adpositions and nouns. The pattern is similar to the overall pattern. For languages that mainly use prepositions as in English, different models perform similarly, while for languages that use postpositions, order-free models get better results. The patterns of adjective modifier (Figure 3.3b) and auxiliary (Figure 3.3c) are also similar.

On dependencies between verbs and object nouns, although in general order-free models perform better, the pattern diverges from what we expect. There can be several possible explanations for this. Firstly, the tokens which are noun objects of verbs only take about 3.1% on average over all tokens. Considering just this specific dependency type, the correlation between frequency distances and performance differences is 0.64, which is far less than 0.9 when considering all types. Therefore, although Verb-Object ordering is a typical example, we cannot take it as the whole story of word order. Secondly,

Verb-Object dependencies can often be difficult to decide. They sometimes are long-ranged and have complex interactions with other words. Therefore, merely reducing modeling order information can have complicated effects. Moreover, although our relative-position self-attention encoder does not explicitly encode word positions, it may still capture some positional information with relative distances. For example, the words in the middle of a sentence will have different distance patterns from those at the beginning or the end. With this knowledge, the model can still prefer the pattern where a verb is in the middle as in English's Subject-Verb-Object ordering and may find sentences in Subject-Object-Verb languages strange. It will be interesting to explore more ways to weaken or remove this bias.

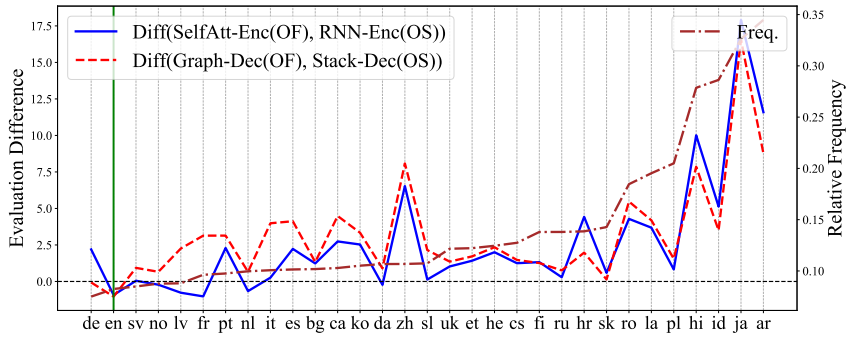


Figure 3.4: Evaluation differences of models on d=1 dependencies. Annotations are the same as in Figure 3.3, languages are sorted by percentages (represented by the brown curve and right y-axis) of d=1 dependencies.

3.4.3.4 Analysis on Dependency Distances

We now look into dependency lengths and directions. Here, we combine dependency length and direction into dependency distance d, by using negative signs for dependencies with left-direction (modifier before head) and positive for right-direction (head before modifier). We find a seemingly strange pattern at dependency distances $|d|$=1: for all transfer models, evaluation scores on d=-1 can reach about 80, but on d=1, the scores are only around 40. This may be explained by the relative frequencies of dependency distances as shown in Table 3.4, where there is a discrepancy between English and the average of other languages at d=1. About 80% of the dependencies with $|d|$=1 in English

is the left direction (modifier before head), while overall other languages have more right directions at $|d|$=1. This suggests an interesting future direction of training on more source languages with different dependency distance distributions.

d	English	Average
<-2	14.36	12.93
-2	15.45	11.83
-1	31.55	30.42
1	7.51	14.22
2	9.84	10.49
>2	21.29	20.11

Table 3.4: Relative frequencies (%) of dependency distances. English differs from the Average at d=1.

We further compare the four models on the d=1 dependencies and as shown in Figure 3.4, the familiar pattern appears again. The order-free models perform better at the languages which have more d=1 dependencies. Such finding indicates that our model design of reducing the ability to capture word order information can help on short-ranged dependencies of different directions to the source language. However, the improvements are still limited. One of the most challenging parts of unsupervised cross-lingual parsing is modeling cross-lingually shareable and language-unspecific information. In other words, we want flexible yet powerful models. Our exploration of the order-free self-attentive models is the first step.

3.4.3.5 Transfer between All Language Pairs

Finally, we investigate the transfer performance of all source-target language pairs.[9] We first generate a performance matrix A, where each entry (i, j) records the transfer performance from a source language i to a target language j. We then report the following two aggregate performance measures on A in Figure 3.5: 1) *As-source* reports the average over columns of A for each row of the source language and 2) *As-target* reports the average

[9]Because the size of training corpus for each language is different in UD, to compare among languages, we train a parser on the first 4,000 sentences for each language and evaluate its transfer performance on all other languages.

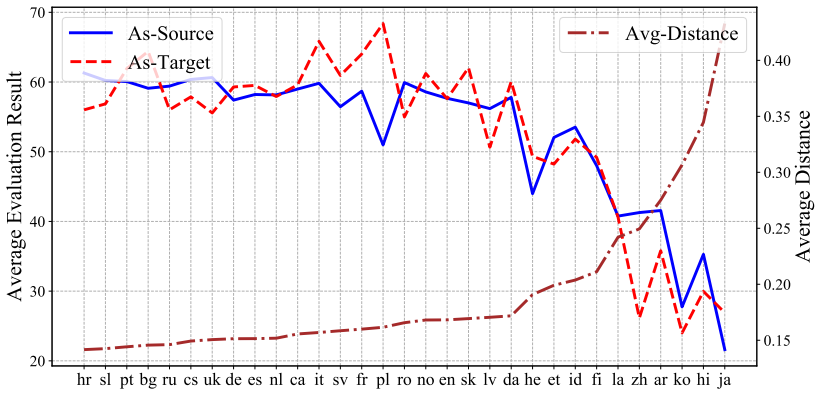


Figure 3.5: Transfer performance of all source-target language pairs. The blue and red curves show the averages over columns and over rows of the source-target pair performance matrix (see text for details). The brown curve and the right y-axis legend represent the average language distance between one language and all others.

over rows of A for each column of the target language. As a reference, we also plot the average word-order distance between one language to other languages. Results show that both *As-source* (blue line) and *As-target* (red line) highly are anti-correlated (Pearson correlation coefficients are -0.90 and -0.87, respectively) with average language distance (brown line).

3.5 Related Work

Cross-language transfer learning employing deep neural networks has widely been studied in the areas of natural language processing (Ma and Xia, 2014a; Guo et al., 2015; Kim et al., 2017; Kann et al., 2017; Cotterell and Duh, 2017), speech recognition (Xu et al., 2014; Huang et al., 2013), and information retrieval (Vulić and Moens, 2015b; Sasaki et al., 2018; Litschko et al., 2018). Learning the language structure (e.g., morphology, syntax) and transferring knowledge from the source language to the target language is the main underneath challenge, and has been thoroughly investigated for a wide variety of NLP applications, including sequence tagging (Yang et al., 2016; Buys and Botha, 2016), name entity recognition (Xie et al., 2018), dependency parsing (Tiedemann, 2015; Agić et al., 2014), entity coreference resolution and linking (Kundu et al., 2018; Sil et al., 2018),

sentiment classification (Zhou et al., 2015, 2016b), and question answering (Joty et al., 2017).

Existing work on unsupervised cross-lingual dependency parsing, in general, trains a dependency parser on the source language and then directly run on the target languages. Training of the monolingual parsers are often delexicalized, i.e., removing all lexical features from the source treebank (Zeman and Resnik, 2008; McDonald et al., 2013), and the underlying feature model is selected from a shared part-of-speech (POS) representation utilizing the Universal POS Tagset (Petrov et al., 2012). Another pool of prior work improves the delexicalized approaches by adapting the model to fit the target languages better. Cross-lingual approaches that facilitate the usage of lexical features includes choosing the source language data points suitable for the target language (Søgaard, 2011; Täckström et al., 2013), transferring from multiple sources (McDonald et al., 2011; Guo et al., 2016; Täckström et al., 2013), using cross-lingual word clusters (Täckström et al., 2012b) and lexicon mapping (Xiao and Guo, 2014; Guo et al., 2015). In this paper, we consider single-source transfer–train a parser on a single source language, and evaluate it on the target languages to test the transferability of neural architectures.

Multilingual transfer (Ammar et al., 2016a; Naseem et al., 2012; Zhang and Barzilay, 2015) is another broad category of techniques applied to parsing where knowledge from many languages having a common linguistic typology is utilized. Recent works (Aufrant et al., 2016; Wang and Eisner, 2018a,b) demonstrated the significance of explicitly extracting and modeling linguistic properties of the target languages to improve cross-lingual dependency parsing. Our work is different in that we focus on the neural architectures and explore their influences on cross-lingual transfer.

3.6 Summary

In this work, we conduct a comprehensive study on how the design of neural architectures affects cross-lingual transfer learning. We examine two notable families of neural architectures (sequential RNN v.s. self-attention) using dependency parsing as the evaluation

task. We show that *order-free* models perform better than *order-sensitive* ones when there is a significant difference in the word order typology between the target and source language. In the future, we plan to explore multi-source transfer and incorporating prior linguistic knowledge into the models for better cross-lingual transfer.

CHAPTER 4

Cross-lingual Representation Learning for Information Extraction

4.1 Introduction

Relation and event extraction are two challenging information extraction (IE) tasks; wherein a model learns to identify semantic relationships between entities and events in narratives. They provide useful information for many natural language processing (NLP) applications such as knowledge graph completion (Lin et al., 2015) and question answering (Chen et al., 2019b). Figure 5.1 gives an example of relation and event extraction tasks. Recent advances in cross-lingual transfer learning approaches for relation and event extraction learns a universal encoder that produces language-agnostic contextualized representations so the model learned on one language can easily transfer to others. Recent works (Huang et al., 2018; Subburathinam et al., 2019a) suggested embedding *universal dependency structure* into contextual representations improves cross-lingual transfer for information extraction.

There are a couple of advantages of leveraging dependency structures. First, the syntactic distance between two words[1] in a sentence is typically smaller than the sequential distance. For example, in the sentence *A fire in a Bangladeshi garment factory has left at least 37 people dead and 100 hospitalized*, the sequential and syntactic distance between "fire" and "hospitalized" is 15 and 4, respectively. Therefore, encoding syntax structure helps capture long-range dependencies (Liu et al., 2018c). Second, languages have different

[1]The shortest path in the dependency graph structure.

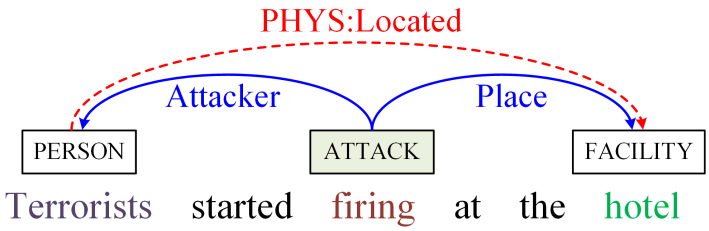


Figure 4.1: A relation (red dashed) between two entities and an event of type *Attack* (triggered by "firing") including two arguments and their role labels (blue) are highlighted.

word order, e.g., adjectives precede or follow nouns as ("red apple") in English or ("pomme rouge") in French. Thus, processing sentences sequentially suffers from the word order difference issue (Ahmad et al., 2019a), while modeling dependency structures can mitigate the problem in cross-lingual transfer (Liu et al., 2019a).

A common way to leverage dependency structures for cross-lingual NLP tasks is using universal dependency parses.[2] A large pool of recent works in IE (Liu et al., 2018c; Zhang et al., 2018b; Subburathinam et al., 2019a; Fu et al., 2019; Sun et al., 2019a; Liu et al., 2019a) employed Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017) to learn sentence representations based on their universal dependency parses, where a k-layers GCN aggregates information of words that are k hop away. Such a way of embedding structure may hinder cross-lingual transfer when the source and target languages have different path length distributions among words (see Table 4.1). Presumably, a two-layer GCN would work well on English but may not transfer well to Arabic.

Moreover, GCNs have shown to perform poorly in modeling long-distance dependencies or disconnected words in the dependency tree (Zhang et al., 2019a; Tang et al., 2020). In contrast, the self-attention mechanism (Vaswani et al., 2017) is capable of capturing long-range dependencies. Consequently, a few recent studies proposed dependency-aware self-attention and found effective for machine translation (Deguchi et al., 2019; Bugliarello and Okazaki, 2020). The key idea is to allow attention between connected words in the dependency tree and gradually aggregate information across layers. However, IE tasks are relatively low-resource (the number of annotated documents available for training is

[2]https://universaldependencies.org/

small), and thus stacking more layers is not feasible. Besides, our preliminary analysis indicates that syntactic distance between entities could characterize certain relation and event types.[3] Hence, we propose to allow attention between all words but use the *pairwise syntactic distances* to weigh the attention.

We introduce a **G**raph **A**ttention **T**ransformer **E**ncoder (GATE) that utilizes self-attention (Vaswani et al., 2017) to learn structured contextual representations. On one hand, GATE enjoys the capability of capturing long-range dependencies, which is crucial for languages with longer sentences, e.g., Arabic.[4] On the other hand, GATE is agnostic to language word order as it uses syntactic distance to model pairwise relationship between words. This characteristic makes GATE suitable to transfer across typologically diverse languages, e.g., English to Arabic. One crucial property of GATE is that it allows information propagation among different heads in the multi-head attention structure based on syntactic distances, which allows to learn the correlation between different mention types and target labels.

We conduct experiments on cross-lingual transfer among English, Chinese, and Arabic languages using the ACE 2005 benchmark (Walker et al., 2006). The experimental results demonstrate that GATE outperforms three recently proposed relation and event extraction methods by a significant margin.[5] We perform a thorough ablation study and analysis, which shows that GATE is less sensitive to source language's characteristics (e.g., word order, sentence structure) and thus excels in the cross-lingual transfer.

---

[3]In ACE 2005 dataset, the relation type *PHYS:Located* exists among *{PER, ORG, LOC, FAC, GPE}* entities. The average syntactic distance in English and Arabic sentences among *PER* and any of the *{LOC, FAC, GPE}* entities are approx. 2.8 and 4.2, while the distance between *PER* and *ORG* is 3.3 and 1.5.

[4]After tokenization, on average, ACE 2005 English and Arabic sentences have approximately 30 and 210 words, respectively.

[5]Code available at https://github.com/wasiahmad/GATE

4.2 Background

In this paper, we focus on *sentence-level* relation extraction (Subburathinam et al., 2019a; Ni and Florian, 2019) and event extraction (Subburathinam et al., 2019a; Liu et al., 2019a) tasks. Below, we first introduce the basic concepts, the notations, as well as define the problem and the scope of the work.

Relation Extraction is the task of identifying the relation type of an ordered pair of entity mentions. Formally, given a pair of entity mentions from a sentence s - $(e_s, e_o; s)$ where e_s and e_o denoted as the subject and object entities respectively, the relation extraction (RE) task is defined as predicting the relation $r \in R \cup \{\texttt{None}\}$ between the entity mentions, where R is a pre-defined set of relation types. In the example provided in Figure 5.1, there is a `PHYS:Located` relation between the entity mentions "Terrorists" and "hotel".

Event Extraction can be decomposed into two sub-tasks, *Event Detection* and *Event Argument Role Labeling*. Event detection refers to the task of identifying *event triggers* (the words or phrases that express event occurrences) and their types. In the example shown in Figure 5.1, the word "firing" triggers the `Attack` event.

Event argument role labeling (EARL) is defined as predicting whether words or phrases (arguments) participate in events and their roles. Formally, given an event trigger e_t and a mention e_a (an entity, time expression, or value) from a sentence s, the argument role labeling refers to predicting the mention's role $r \in R \cup \{\texttt{None}\}$, where R is a pre-defined set of role labels. In Figure 5.1, the "Terrorists" and "hotel" entities are the arguments of the `Attack` event and they have the `Attacker` and `Place` role labels, respectively.

In this work, we focus on the EARL task; we assume event mentions (triggers) of the input sentence are provided.

Zero-Short Cross-Lingual Transfer refers to the setting, where there is no labeled examples available for the *target* language. We train neural relation extraction and event argument role labeling models on one (single-source) or multiple (multi-source) *source*

languages and then deploy the models in target languages. The overall cross-lingual transfer approach consists of four steps:

1. Convert the input sentence into a language-universal tree structure using an off-the-shelf universal dependency parser, e.g., UDPipe[6] (Straka and Straková, 2017).
2. Embed the words in the sentence into a shared semantic space across languages. We use off-the-shelf multilingual contextual encoders (Devlin et al., 2019; Conneau et al., 2019) to form the word representations. To enrich the word representations, we concatenate them with *universal* part-of-speech (POS) tag, dependency relation, and entity type embeddings (Subburathinam et al., 2019a). We collectively refer them as *language-universal* features.
3. Based on the word representations, we encode the input sentence using the proposed GATE architecture that leverages the syntactic depth and distance information. Note that this step is the main focus of this work.
4. A pair of classifier predicts the target relation and argument role labels based on the encoded representations.

4.3 Approach

Our proposed approach GATE revises the multi-head attention architecture in Transformer Encoder (Vaswani et al., 2017) to model syntactic information while encoding a sequence of input vectors (represent the words in a sentence) into contextualized representations. We first review the standard multi-head attention mechanism (§5.2.1). Then, we introduce our proposed method GATE (§4.3.2). Finally, we describe how we perform relation extraction (§4.3.3) and event argument role labeling (§4.3.4) tasks.

[6]http://ufal.mff.cuni.cz/udpipe

4.3.1 Transformer Encoder

Unlike recent works (Zhang et al., 2018b; Subburathinam et al., 2019a) that use GCNs (Kipf and Welling, 2017) to encode the input sequences into contextualized representations, we propose to employ Transformer encoder as it excels in capturing long-range dependencies. First, the sequence of input word vectors, $x = [x_1, \ldots, x_{|x|}]$ where $x_i \in R^d$ are packed into a matrix $H^0 = [x_1, \ldots, x_{|x|}]$. Then an L-layer Transformer Encoder $H^l = Transformer_l(H^{l-1})$, $l \in [1, L]$ takes H^0 as input and generates different levels of latent representations $H^l = [h_1^l, \ldots, h_{|x|}^l]$, recursively. Typically the latent representations generated by the last layer (L-th layer) are used as the contextual representations of the input words. To aggregate the output vectors of the previous layer, multiple (n_h) self-attention heads are employed in each Transformer layer. For the l-th Transformer layer, the output of the previous layer $H^{l-1} \in \mathbb{R}^{|x| \times d_{model}}$ is first linearly projected to queries Q, keys K, and values V using parameter matrices $W_l^Q, W_l^K \in \mathbb{R}^{d_{model} \times d_k}$ and $W_l^V \in \mathbb{R}^{d_{model} \times d_v}$, respectively.

$$Q_l = H^{l-1}W_l^Q, K_l = H^{l-1}W_l^K, V_l = H^{l-1}W_l^V.$$

The output of a self-attention head A_l is computed as:

$$A_l = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V_l, \tag{4.1}$$

where the matrix $M \in \mathbb{R}^{|x| \times |x|}$ determines whether a pair of tokens can attend each other.

$$M_{ij} = \begin{cases} 0, & \text{allow to attend} \\ -\infty, & \text{prevent from attending} \end{cases} \tag{4.2}$$

The matrix M is deduced as a *mask*. By default, the matrix M is a *zero-matrix*. In the next section, we discuss how we manipulate the mask matrix M to incorporate syntactic depth and distance information in sentence representations.

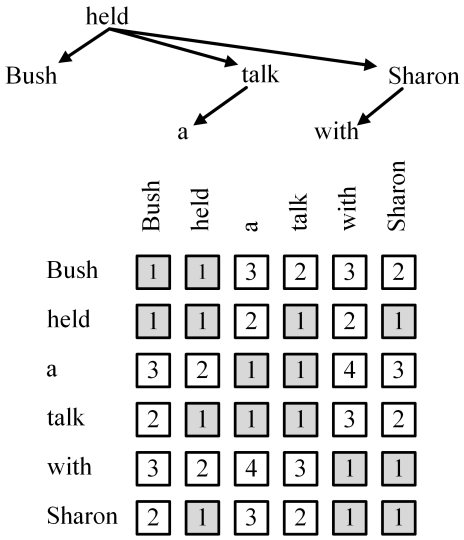


Figure 4.2: Distance matrix showing the shortest path distances between all pairs of words. The dependency arc direction is ignored while computing pairwise distances. The diagonal value is set to 1, indicating a self-loop. If we set the values in white cells (with value > 1) to 0, the distance matrix becomes an adjacency matrix.

4.3.2 Graph Attention Transformer Encoder

The self-attention as described in §5.2.1 learns how much attention to put on words in a text sequence when encoding a word at a given position. In this work, we revise the self-attention mechanism such that it takes into account the syntactic structure and distances when a token attends to all the other tokens. The key idea is to manipulate the mask matrix to impose the graph structure and retrofit the attention weights based on pairwise syntactic distances. We use the universal dependency parse of a sentence and compute the syntactic (shortest path) distances between every pair of words. We illustrate an example in Figure 4.2.

We denote distance matrix $D \in \mathbb{R}^{|x| \times |x|}$ where D_{ij} represents the syntactic distance between words at position i and j in the input sequence. If we want to allow tokens to attend their adjacent tokens (that are 1 hop away) at each layer, then we can set the

mask matrix as follows.

$$M_{ij} = \begin{cases} 0, & D_{ij} = 1 \\ -\infty, & \text{otherwise} \end{cases}$$

We generalize this notion to model a distance based attention; allowing tokens to attend tokens that are within distance δ (hyper-parameter).

$$M_{ij} = \begin{cases} 0, & D_{ij} \leq \delta \\ -\infty, & \text{otherwise} \end{cases} \tag{4.3}$$

During our preliminary analysis, we observed that syntactic distances between entity mentions or event mentions often correlate with the target label. For example, if an `ORG` entity mention appears closer to a `PER` entity than a `LOC` entity, then the {`PER, ORG`} entity pair is more likely to have the `PHYS:Located` relation. We hypothesize that modeling syntactic distance between words can help to identify complex semantic structure such as events and entity relations. Hence we revise the attention head A_l (defined in Eq. (5.1)) computation as follows.

$$A_l = F\left(\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)\right) V_l. \tag{4.4}$$

Here, softmax produces an attention matrix $P \in R^{|x|\times|x|}$ where P_i denotes the attentions that i-th token pays to the all the tokens in the sentence, and F is a function that modifies those attention weights. We can treat F as a parameterized function that can be learned based on distances. However, we adopt a simple formulation of F such that GATE pays more attention to tokens that are closer and less attention to tokens that are faraway in the parse tree. We define the (i, j)-th element of the attention matrix produced by F as:

$$F(P)_{ij} = \frac{P_{ij}}{Z_i D_{ij}}, \tag{4.5}$$

where $Z_i = \sum_j \frac{P_{ij}}{D_{ij}}$ is the normalization factor and D_{ij} is the distance between i-th and j-th token. We found this formulation of F effective for the IE tasks.

4.3.3 Relation Extractor

Relation Extractor predicts the relationship label (or `None`) for each mention pair in a sentence. For an input sentence s, GATE produces contextualized word representations $h^l_1, \ldots, h^l_{|x|}$ where $h^l_i \in \mathbb{R}^{d_{model}}$. As different sentences and entity mentions may have different lengths, we perform max-pooling over their contextual representations to obtain fixed-length vectors. Suppose for a pair of entity mentions $e_s = [h^l_{bs}, \ldots, h^l_{es}]$ and $e_o = [h^l_{bo}, \ldots, h^l_{eo}]$, we obtain single vector representations $\hat{e}_s$ and $\hat{e}_o$ by performing max-pooling. Following Zhang et al. (2018b); Subburathinam et al. (2019a), we also obtain a vector representation for the sentence, $\hat{s}$ by applying max-pooling over $[h^l_1, \ldots, h^l_{|x|}]$ and concatenate the three vectors. Then the concatenation of the three vectors $[\hat{e}_s; \hat{e}_o; \hat{s}]$ are fed to a linear classifier followed by a Softmax layer to predict the relation type between entity mentions e_s and e_o as follows.

$$\mathcal{O}_r = \text{softmax}(\boldsymbol{W}_r^T [\hat{e_s}; \hat{e_o}; \hat{s}] + \boldsymbol{b}_r),$$

where $\boldsymbol{W}_r \in R^{3d_{model} \times r}$ and $\boldsymbol{b}_r \in R^r$ are parameters, and r is the total number of relation types. The probability of t-th relation type is denoted as $P(r_t|s, e_s, e_o)$, which corresponds to the t-th element of $\mathcal{O}_r$. To train the relation extractor, we adopt the cross-entropy loss.

$$\mathcal{L}_r = -\sum_{s=1}^{N} \sum_{o=1}^{N} \log(P(y^r_{so}|s, e_s, e_o)),$$

where N is the number of entity mentions in the input sentence s and y^r_{so} denotes the ground truth relation type between entity mentions e_s and e_o.

	Sequential			Syntactic		
	En	Zh	Ar	En	Zh	Ar
Relation Mention	4.8	3.9	25.8	2.2	2.6	5.1
Event Mention & Argument	9.8	21.7	58.1	3.1	4.6	12.3

Table 4.1: Average sequential and syntactic (shortest path) distance between relation mentions and event mentions and their candidate arguments in ACE05 dataset. Distances are computed by ignoring the order of mentions.

4.3.4 Event Argument Role Labeler

Event argument role labeler predicts the argument mentions (or `None` for non-argument mentions) of an event mention and assigns a role label to each argument from a pre-defined set of labels. To label an argument candidate $e_a = [h^l_{ba}, \ldots, h^l_{ea}]$ for an event trigger $e_t = [h^l_{bt}, \ldots, h^l_{et}]$ in sentence $s = [h^l_1, \ldots, h^l_{|x|}]$, we apply max-pooling to form vectors $\hat{e}_a$, $\hat{e}_t$, and $\hat{s}$ respectively, which is same as that for relation extraction. Then we concatenate the vectors ($[\hat{e_t}; \hat{e_a}; \hat{s}]$) and pass it through a linear classifier and Softmax layer to predict the role label as follows.

$$\mathcal{O}_a = \text{softmax}(\boldsymbol{W}_a^T [\hat{e_t}; \hat{e_a}; \hat{s}] + \boldsymbol{b}_a),$$

where $\boldsymbol{W}_a \in R^{3d_{model} \times r}$ and $\boldsymbol{b}_a \in R^r$ are parameters, and r is the total number of argument role label types. We optimize the role labeler by minimizing the cross-entropy loss.

4.4 Experiment Setup

	English	Chinese	Arabic
Relations Mentions	8,738	9,317	4,731
Event Mentions	5,349	3,333	2,270
Event Arguments	9,793	8,032	4,975

Table 4.2: Statistics of the ACE 2005 dataset.

Model	Event Argument Role Labeling						Relation Extraction					
	En ⇓ Zh	En ⇓ Ar	Zh ⇓ En	Zh ⇓ Ar	Ar ⇓ En	Ar ⇓ Zh	En ⇓ Zh	En ⇓ Ar	Zh ⇓ En	Zh ⇓ Ar	Ar ⇓ En	Ar ⇓ Zh
CL_Trans_GCN	41.8	55.6	41.2	52.9	39.6	40.8	56.7	65.3	65.9	59.7	59.6	46.3
CL_GCN	51.9	50.4	53.7	51.5	50.3	51.9	49.4	58.3	65.0	55.0	56.7	42.4
CL_RNN	60.4	53.9	55.7	52.5	50.7	50.9	53.7	63.9	70.9	57.6	67.1	55.7
Transformer	61.5	55.0	58.0	57.7	54.3	57.0	57.1	63.4	69.6	60.6	67.0	52.6
Transformer_RPR	62.3	60.8	57.3	66.3	**57.5**	**59.8**	**58.0**	59.9	70.0	55.6	66.5	**56.5**
GATE (this work)	**63.2**	**68.5**	**59.3**	**69.2**	53.9	57.8	55.1	**66.8**	**71.5**	**61.2**	**69.0**	54.3

Table 4.3: Single-source transfer results (F-score % on the test set) using perfect event triggers and entity mentions. The language on top and bottom of ⇓ denotes the source and target languages, respectively.

4.4.1 Dataset

We conduct experiments based on the Automatic Content Extraction (ACE) 2005 corpus (Walker et al., 2006) that includes manual annotation of relation and event mentions (with their arguments) in three languages: English (En), Chinese (Zh), and Arabic (Ar). We present the data statistics in Table 4.2. ACE defines an ontology that includes 7 entity types, 18 relation subtypes, and 33 event subtypes. We add a class label `None` to denote that two entity mentions or a pair of an event mention and an argument candidate under consideration do not have a relationship belong to the target ontology. We use the same dataset split as Subburathinam et al. (2019a) and follow their preprocessing steps. We refer the readers to Subburathinam et al. (2019a) for the dataset preprocessing details.

4.4.2 Evaluation Criteria

Following the previous works (Ji and Grishman, 2008; Li et al., 2013; Li and Ji, 2014; Subburathinam et al., 2019a), we set the evaluation criteria as, (1) a relation mention is correct if its predicted type and the head offsets of the two associated entity mentions are correct, and (2) an event argument role label is correct if the event type, offsets, and argument role label match any of the reference argument mentions.

Model	{En, Zh} ⇓ Ar	{En, Ar} ⇓ Zh	{Zh, Ar} ⇓ En
Event Argument Role Labeling			
CL_Trans_GCN	57.0	44.5	44.8
CL_GCN	58.9	56.2	57.9
CL_RNN	53.5	62.5	60.8
Transformer	59.5	62.0	60.7
Transformer_RPR	71.1	**68.4**	**62.2**
GATE (this work)	**73.9**	65.3	61.3
Relation Extraction			
CL_Trans_GCN	66.8	54.4	69.5
CL_GCN	64.0	46.6	65.8
CL_RNN	66.5	60.5	73.0
Transformer	**68.3**	59.3	73.7
Transformer_RPR	65.0	**62.3**	73.8
GATE (this work)	67.0	57.9	**74.1**

Table 4.4: Multi-source transfer results (F-score % on the test set) using perfect event triggers and entity mentions. The language on top and bottom of ⇓ denotes the source and target languages, respectively.

4.4.3 Baseline Models

To compare GATE on relation and event argument role labeling tasks, we chose three recently proposed approaches as baselines. The source code of the baselines are not publicly available at the time this research is conducted. Therefore, we reimplemented them.

• CL_Trans_GCN (Liu et al., 2019a) is a context-dependent lexical mapping approach where each word in a source language sentence is mapped to its best-suited translation in the target language. We use multilingual word embeddings (Joulin et al., 2018) as the continuous representations of tokens along with the language-universal features embeddings including part-of-speech (POS) tag embedding, dependency relation label embedding, and entity type embedding.[7] Since this model focuses on the target language, we train this baseline for each combination of source and target languages.

[7]Due to the design principle of Liu et al. (2019a), we cannot use multilingual contextual encoders in CL_Trans_GCN.

• CL_GCN (Subburathinam et al., 2019a) uses GCN (Kipf and Welling, 2017) to learn structured common space representation. To embed the tokens in an input sentence, we use multilingual contextual representations (Devlin et al., 2019; Conneau et al., 2019) and the language-universal feature embeddings. We train this baseline on the source languages and directly evaluate on the target languages.

• CL_RNN (Ni and Florian, 2019) uses a bidirectional Long Short-Term Memory (LSTM) type recurrnet neural networks (Hochreiter and Schmidhuber, 1997) to learn contextual representation. We feed language-universal features for words in a sentence, constructed in the same way as Subburathinam et al. (2019a). We train and evaluate this baseline in the same way as CL_GCN.

In addition to the above three baseline methods, we compare GATE with the following two encoding methods.

• Transformer (Vaswani et al., 2017) uses multi-head self-attention mechanism and is the base structure of our proposed model, GATE. Note that GATE has the same number of parameters as Transformer since GATE does not introduce any new parameter while modeling the pairwise syntactic distance into the self-attention mechanism. Therefore, we credit the GATE's improvements over the Transformer to its distance-based attention modeling strategy.

• Transformer_RPR (Shaw et al., 2018b) uses relative position representations to encode the structure of the input sequences. This method uses the pairwise *sequential* distances while GATE uses pairwise *syntactic* distances to model attentions between tokens.

4.4.4 Implementation Details

To embed words into vector representations, we use multilingual BERT (M-BERT) (Devlin et al., 2019). Note that we do not fine-tune M-BERT, but only use it as a feature extractor. We use the universal part-of-speech (POS) tags, dependency relation labels, and seven entity types defined by ACE: person, organization, geo-political entity, location, facility, weapon, and vehicle. We embed these language-universal features into fixed-length vectors

and concatenate them with M-BERT vectors to form the input word representations. We set the model size (d_{model}), number of encoder layers (L), and attention heads (n_h) in multi-head to 512, 1, and 8 respectively. We tune the distance threshold δ (as shown in Eq. (4.3)) in $[1, 2, 4, 8, \infty]$ for each attention head on each source language (more details are provided in the supplementary).

We implement all the baselines and our approach based on the implementation of Zhang et al. (2018b) and OpenNMT (Klein et al., 2017). We used `transformers`[8] to extract M-BERT and XLM-R features. We provide a detailed description of the dataset, hyper-parameters, and training of the baselines and our approach in the supplementary.

4.5 Results and Analysis

We compare GATE with five baseline approaches on event argument role labeling (EARL) and relation extraction (RE) tasks, and the results are presented in Table 4.3 and 4.4.

4.5.1 Single-source transfer

In the single-source transfer setting, all the models are individually trained on *one* source language, e.g., English and directly evaluated on the other two languages (target), e.g., Chinese and Arabic. Table 4.3 shows that GATE outperforms all the baselines in four out of six transfer directions on both tasks. CL_RNN surprisingly outperforms CL_GCN in most settings, although CL_RNN uses a BiLSTM that is not suitable to transfer across syntactically different languages (Ahmad et al., 2019a). We hypothesize the reason being GCNs cannot capture long-range dependencies, which is crucial for the two tasks. In comparison, by modeling distance-based pairwise relationships among words, GATE excels in cross-lingual transfer.

A comparison between Transformer and GATE demonstrates the effectiveness of syntactic distance-based self-attention over the standard mechanism. From Table 4.3, we see

[8]https://github.com/huggingface/transformers

Model	EARL		RE	
	Chinese	Arabic	Chinese	Arabic
Wang et al. (2019b)				
Absolute	61.2	53.5	57.8	65.2
Relative	55.3	47.1	**58.1**	66.4
GATE	**63.2**	**68.5**	55.1	**66.8**

Table 4.5: GATE vs. Wang et al. (2019b) results (F-score %) on event argument role labeling (EARL) and relation extraction (RE); using English as source and Chinese, Arabic as the target languages, respectively. To limit the maximum relative position, the clipping distance is set to 10 and 5 for EARL and RE tasks, respectively.

GATE outperforms Transformer with an average improvement of 4.7% and 1.3% in EARL and RE tasks, respectively. Due to implicitly modeling graph structure, Transformer_RPR performs effectively. However, GATE achieves an average improvement of 1.3% and 1.9% in EARL and RE tasks over Transformer_RPR. Overall, the significant performance improvements achieved by GATE corroborate our hypothesis that syntactic distance-based attention helps in the cross-lingual transfer.

4.5.2 Multi-source transfer

In the multi-source cross-lingual transfer, the models are trained on a pair of languages: {English, Chinese}, {English, Arabic}, and {Chinese, Arabic}. Hence, the models observe more examples during training, and as a result, the cross-lingual transfer performance improves compared to the single-source transfer setting. In Table 4.4, we see GATE outperforms the previous three IE approaches in multi-source transfer settings, except on RE for the source:{English, Arabic} and target: Chinese language setting. On the other hand, GATE performs competitively to Transformer and Transformer_RPR baselines. Due to observing more training examples, Transformer and Transformer_RPR perform more effectively in this setting. The overall result indicates that GATE more efficiently learns transferable representations for the IE tasks.

4.5.3 Encoding dependency structure

GATE encodes the dependency structure of sentences by guiding the attention mechanism in self-attention networks (SANs). However, an alternative way to encode the sentence structure is through positional encoding for SANs. Conceptually, the key difference is the modeling of syntactic distances to capture fine-grained relations among tokens. Hence, we compare these two notions of encoding the dependency structure to emphasize the promise of modeling syntactic distances.

To this end, we compare the GATE with Wang et al. (2019b) that proposed structural position encoding using the dependency structure of sentences. Results are presented in Table 4.5. We see that Wang et al. (2019b) performs well on RE but poorly on EARL, especially on the Arabic language. While GATE directly uses syntactic distances between tokens to guide the self-attention mechanism, Wang et al. (2019b) learns parameters to encode structural positions that can become sensitive to the source language. For example, the average shortest path distance between event mentions and their candidate arguments in English and Arabic is 3.1 and 12.3, respectively (see Table 4.1). As a result, a model trained in English may learn only to attend closer tokens, thus fails to generalize on Arabic.

Moreover, we anticipate that different order of subject and verb in English and Arabic[9] causes Wang et al. (2019b) to transfer poorly on the EARL task (as event triggers are mostly verbs). To verify our anticipation, we modify the relative structural position encoding (Wang et al., 2019b) by dropping the directional information (Ahmad et al., 2019a), and observed a performance increase from 47.1 to 52.2 for English to Arabic language transfer. In comparison, GATE is order-agnostic as it models syntactic distance; hence, it has a better transferability across typologically diverse languages.

[9]According to WALS (Dryer and Haspelmath, 2013b), the order of subject (S), object (O), and verb (V) for English, Chinese and Arabic is SVO, SVO, and VSO.

Model	EARL		RE	
	English	Chinese*	English	Chinese*
CL_GCN	51.5	56.3	46.9	50.7
CL_RNN	55.6	59.3	56.8	**62.0**
GATE	**63.8**	**64.2**	**58.8**	57.0

Table 4.6: Event argument role labeling (EARL) and relation extraction (RE) results (F-score %); using Chinese as the source and English as the target language. * indicates the English examples are translated into Chinese using `Google Cloud Translate`.

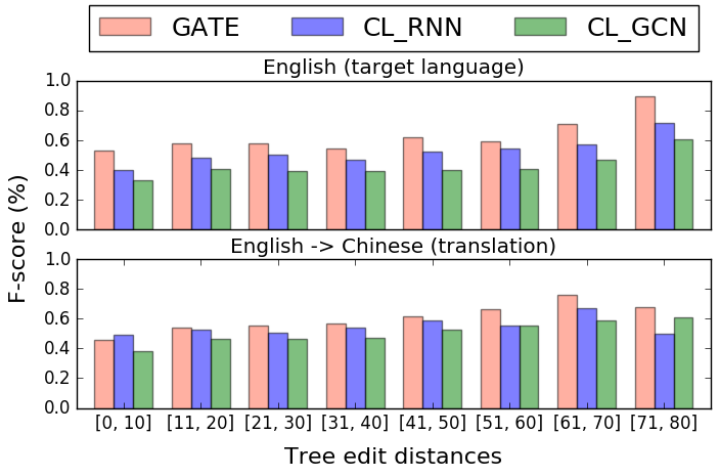


Figure 4.3: Models trained on the Chinese language perform on event argument role labeling in English and their parallel Chinese sentences. The parallel sentences have the same meaning but a different structure. To quantify the structural difference between two parallel sentences, we compute the tree edit distances.

4.5.4 Sensitivity towards source language

Intuitively, an RE or EARL model would transfer well on target languages if the model is less sensitive towards the source language characteristics (e.g., word order, grammar structure). To measure sensitivity towards the source language, we evaluate the model performance on the target language and their parallel (translated) source language sentences. We hypothesize that if a model performs significantly well on the translated source language sentences, then the model is more sensitive towards the source language and may not be ideal for cross-lingual transfer. To test the models on this hypothesis, we translate all the ACE05 English test set examples into Chinese using `Google Cloud`

Word features	EARL		RE	
	Chinese	Arabic	Chinese	Arabic
Multi-WE	35.9	43.7	41.0	54.9
M-BERT	**57.1**	54.8	**55.1**	66.8
XLM-R	51.8	**61.7**	51.4	**68.1**

Table 4.7: Contribution of multilingual word embeddings (Multi-WE) Joulin et al. (2018), M-BERT Devlin et al. (2019), and XLM-R Conneau et al. (2019) as a source of word features; using English as source and Chinese, Arabic as the target languages, respectively.

`Translate`.[10] We train GATE and two baselines on the Chinese and evaluate them on both English (test set) examples and their Chinese translations. To quantify the difference between the dependency structure of an English and its Chinese translation sentences, we compute *edit distance* between two tree structures using the APTED[11] algorithm (Pawlik and Augsten, 2015, 2016).

The results are presented in Table 4.6. We see that `CL_GCN` and `CL_RNN` have much higher accuracy on the translated (Chinese) sentences than the target language (English) sentences. On the other hand, GATE makes a roughly similar number of correct predictions when the target and translated sentences are given as input. Figure 4.3 illustrates how the models perform when the structural distance between target sentences and their translation increases. The results suggest that GATE performs substantially better than the baselines when the target language sentences are structurally different from the source language. The overall findings signal that GATE is less sensitive to source language features, and we credit this to the modeling of distance-based syntactic relationships between words. We acknowledge that there might be other factors associated with a model's language sensitivity. However, we leave the detailed analysis for measuring a model's sensitivity towards languages as future work.

[10]Details are provided in the supplementary.

[11]https://pypi.org/project/apted/

Input features	EARL		RE	
	Chinese	Arabic	Chinese	Arabic
M-BERT	52.5	47.4	44.0	49.7
+ POS tag	49.3	47.5	44.1	47.0
+ Dep. label	49.7	51.0	48.6	47.0
+ Entity type	**57.8**	**60.2**	**56.3**	**63.0**

Table 4.8: Ablation on the use of language-universal features (part-of-speech (POS) tag, dependency relation label, and entity type) in GATE (F-score (%); using English as source and Chinese, Arabic as the target languages, respectively.

4.5.5 Ablation study

We perform a detailed ablation study on language-universal features and sources of word features to examine their individual impact on cross-lingual transfer. The results are presented in Table 4.7 and 4.8. We observed that M-BERT and XLM-R produced word features performed better in Chinese and Arabic, respectively, while they are comparable in English. On average M-BERT performs better, and thus we chose it as the word feature extractor in all our experiments. Table 4.8 shows that part-of-speech and dependency relation embedding has a limited contribution. This is perhaps due to the tokenization errors, as pointed out by Subburathinam et al. (2019a). However, the use of language-universal features is useful, particularly when we have minimal training data. We provide more analysis and results in the supplementary.

4.5.6 Error Analysis

We compare our proposed approach GATE and the self-attention mechanism (Vaswani et al., 2017) on the event argument role labeling (EARL) and relation extraction (RE) tasks. We consider the models trained on English language and evaluate them on Chinese language. We do not use the event trigger type as features while training models for the EARL task. We present the confusion matrices of these two models in Figure 4.4, 4.5, 4.6, and 4.7. In general, GATE makes more correct predictions. We noticed that in transferring from English to Chinese on the EARL task, GATE improves notably on Destination, Entity, Person, Place relation types. The syntactic distance between event triggers and

Model	True Positive	True Negative	False Positive	False Negative
Self-Attention	386	563	179	300
GATE	585	493	249	157

Table 4.9: Comparing GATE and Self-Attention on the EARL task using English and Chinese as the source and target languages, respectively. The rates are aggregated from confusion matrices shown in Figure 4.4 and 4.5.

their argument mentions that share those types corroborates with our hypothesis that distance-based dependency relations help in cross-lingual transfer.

However, we observed that GATE makes more *false positive* and less *false negative* predictions than the self-attention mechanism. We summarize the prediction rates on EARL in Table 4.9. There are several factors that may be associated with these wrong predictions. To shed light on those factors, we manually inspect 50 examples and our findings suggests that wrong predictions are due to three primary reasons. First, there are errors in the ground truth annotations in the ACE dataset. Second, the knowledge required for prediction is not available in the input sentence. Third, there are entity mentions, event triggers, and contextual phrases in the test data that rarely appear in the training data.

4.6 Related Work

Relation and event extraction has drawn significant attention from the natural language processing (NLP) community. Most of the approaches developed in past several years are based on supervised machine learning, using either symbolic features (Ahn, 2006; Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011; Li et al., 2013; Li and Ji, 2014) or distributional features (Liao and Grishman, 2011; Nguyen et al., 2016; Miwa and Bansal, 2016; Liu et al., 2018a; Zhang et al., 2018a; Lu and Nguyen, 2018; Chen et al., 2015; Nguyen and Grishman, 2015; Zeng et al., 2014; Peng et al., 2017; Nguyen and Grishman, 2018; Zhang et al., 2018b; Subburathinam et al., 2019a; Liu et al., 2019a; Huang et al., 2020) from a large number of annotations. Joint learning or inference (Bekoulis et al., 2018; Li et al., 2014; Zhang et al., 2019b; Liu et al., 2018c; Nguyen et al.,

2016; Yang and Mitchell, 2016; Han et al., 2019, 2020) are also among the noteworthy techniques.

Most previous works on cross-lingual transfer for relation and event extraction are based on annotation projection (Kim et al., 2010a; Kim and Lee, 2012), bilingual dictionaries (Hsi et al., 2016; Ni and Florian, 2019), parallel data (Chen and Ji, 2009; Kim et al., 2010b; Qian et al., 2014) or machine translation (Zhu et al., 2014; Faruqui and Kumar, 2015; Zou et al., 2018a). Learning common patterns across languages is also explored (Lin et al., 2017; Wang et al., 2018; Liu et al., 2018a). In contrast to these approaches, Subburathinam et al. (2019a); Liu et al. (2019a) proposed to use graph convolutional networks (GCNs) (Kipf and Welling, 2017) to learn multi-lingual structured representations. However, GCNs struggle to model long-range dependencies or disconnected words in the dependency tree. To overcome the limitation, we use the syntactic distances to weigh the attentions while learning contextualized representations via the multi-head attention mechanism (Vaswani et al., 2017).

Moreover, our proposed syntax driven distance-based attention modeling helps to mitigate the word order difference issue (Ahmad et al., 2019a) that hinders cross-lingual transfer. Prior works studied dependency structure modeling (Liu et al., 2019a), source reordering (Rasooli and Collins, 2019a), adversarial training (Ahmad et al., 2019b), constrained inference (Meng et al., 2019) to tackle word order differences across typologically different languages.

4.7 Summary

In this chapter, we showed that modeling fine-grained syntactic structural information based on the dependency parse of a sentence improves cross-lingual transfer. We developed a **G**raph **A**ttention **T**ransformer **E**ncoder (GATE) to generate structured contextual representations. Extensive experiments on three languages demonstrates the effectiveness of GATE in cross-lingual relation and event extraction. In the future, we want to explore other sources of language-universal information to improve representation learning.

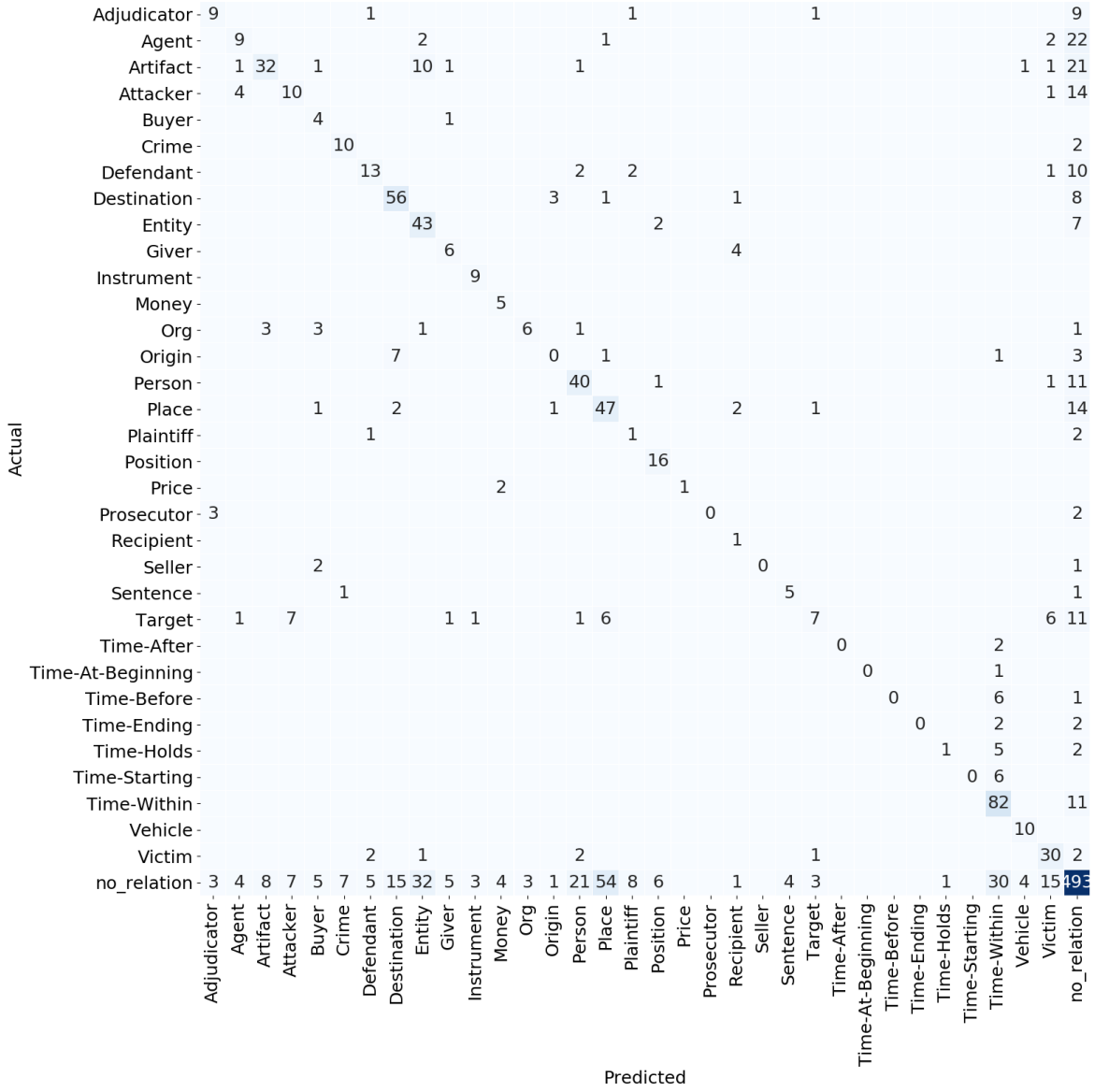


Figure 4.4: Event argument role labeling confusion matrix (on test set) based on our proposed approach **GATE** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts.

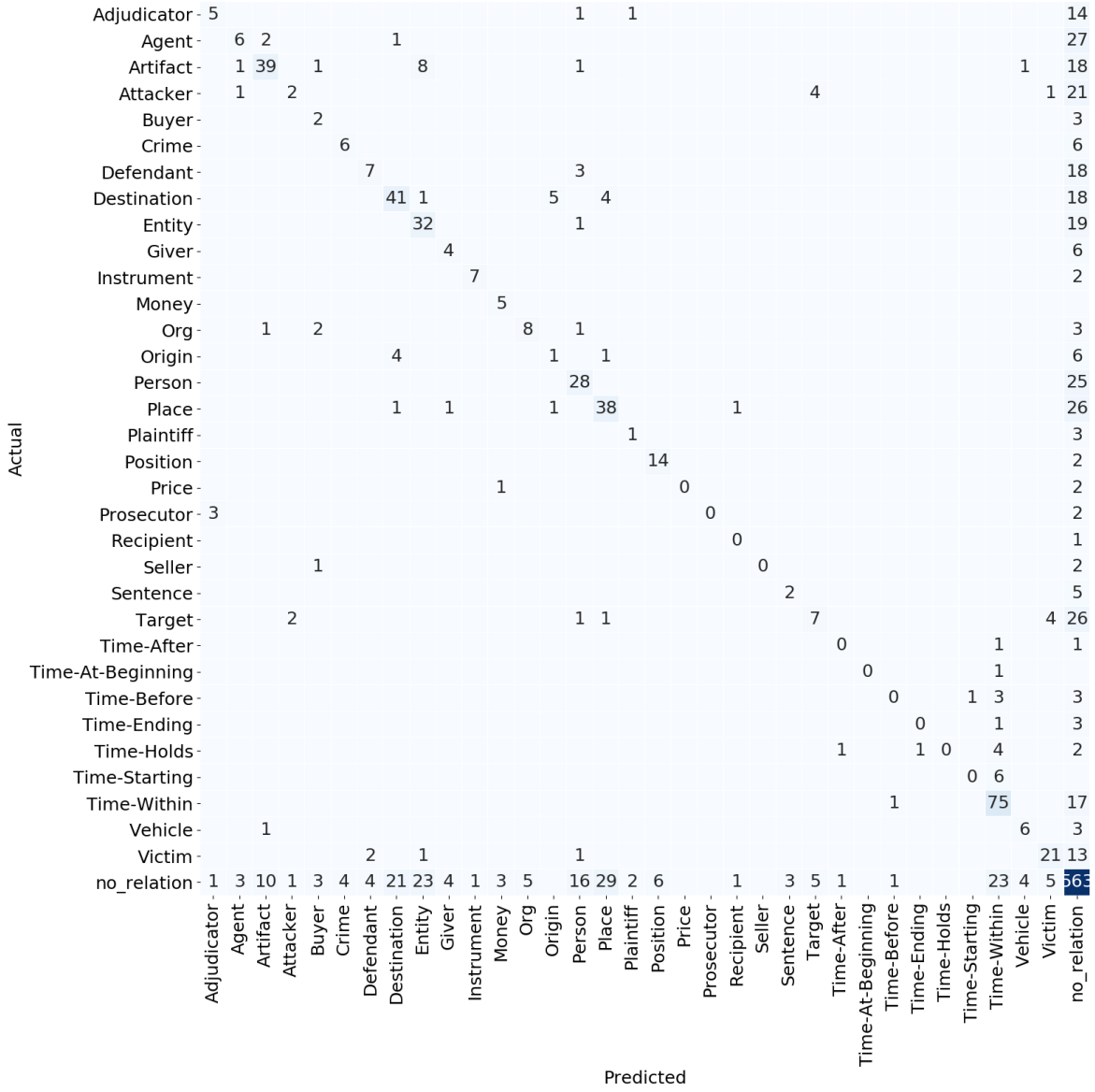


Figure 4.5: Event argument role labeling confusion matrix (on test set) based on the **Self-Attention (Transformer Encoder)** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts.

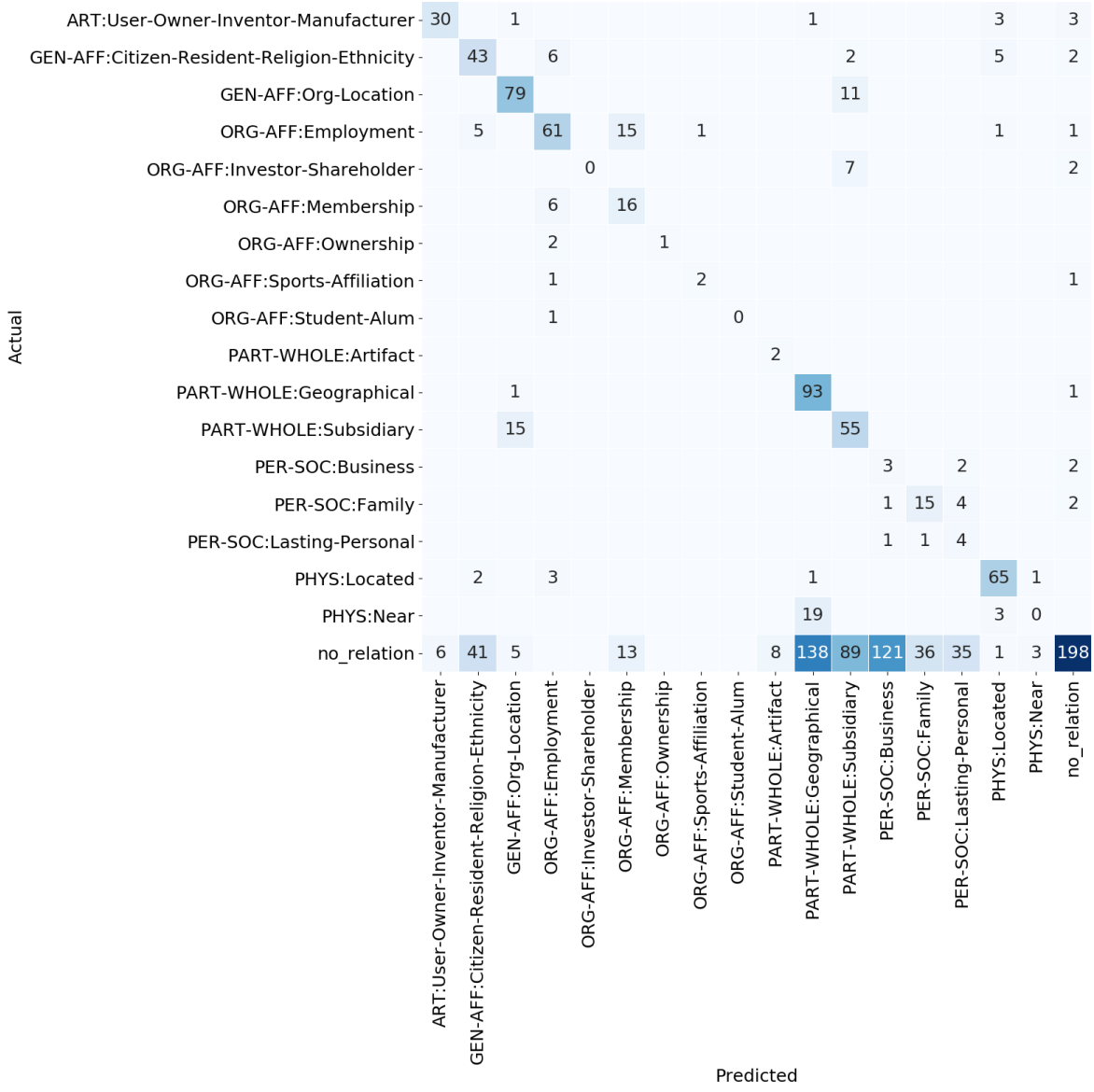


Figure 4.6: Relation extraction labeling confusion matrix (on test set) based on our proposed approach **GATE** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts.

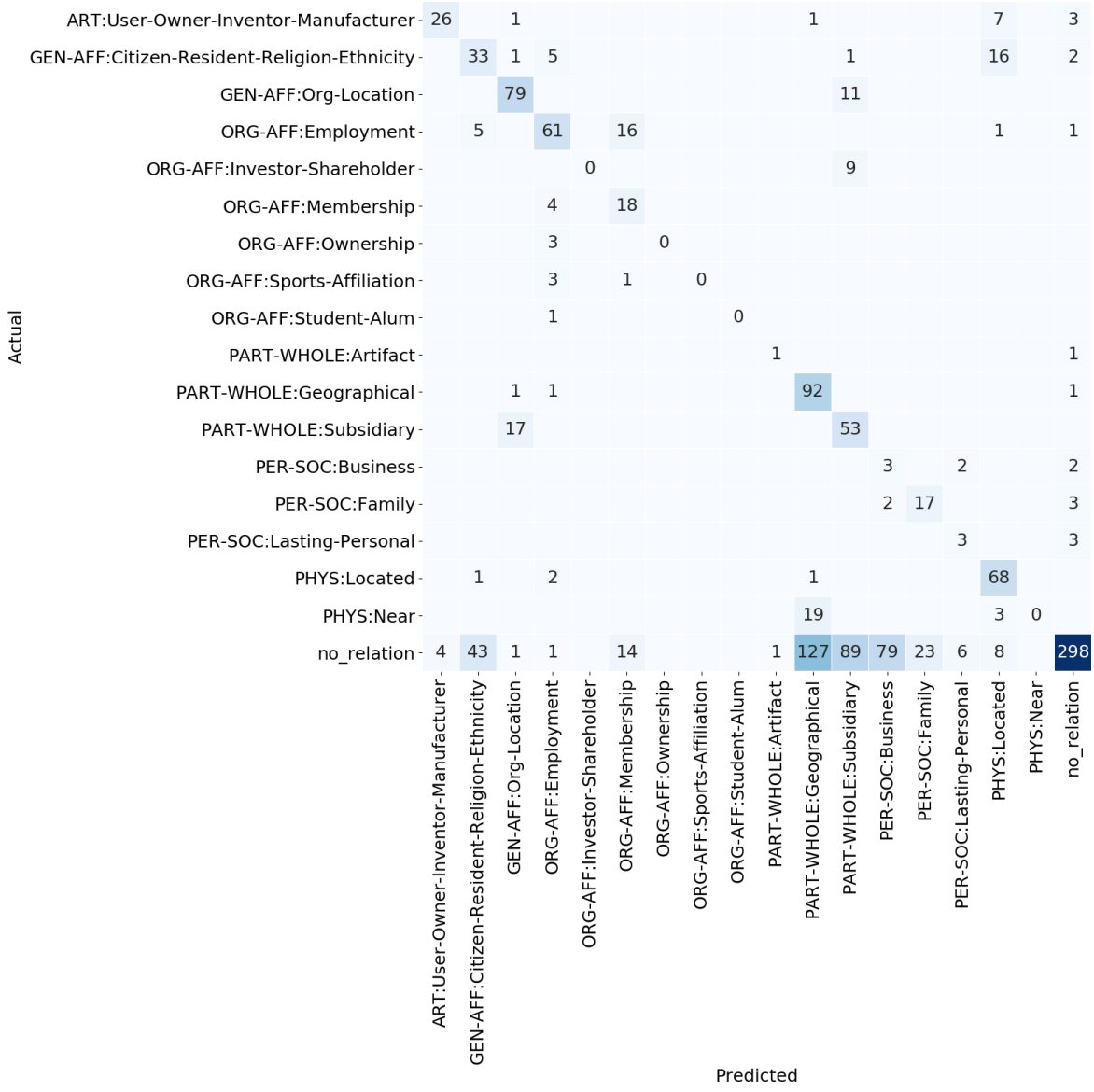


Figure 4.7: Relation extraction confusion matrix (on test set) based on the **Self-Attention (Transformer Encoder)** using English and Chinese as the source and target languages, respectively. The diagonal values indicate the number of correct predictions, while the other values denote the incorrect prediction counts.

CHAPTER 5

Syntax-augmented Pre-trained Encoders for Cross-lingual Transfer

5.1 Introduction

Cross-lingual transfer reduces the requirement of labeled data to perform natural language processing (NLP) in a target language, and thus has the ability to avail NLP applications in low-resource languages. However, transferring across languages is challenging because of linguistic differences at levels of morphology, syntax, and semantics. For example, word order difference is one of the crucial factors that impact cross-lingual transfer (Ahmad et al., 2019a). The two sentences in English and Hindi, as shown in Figure 5.1 have the same meaning but a different word order (while English has an SVO (`Subject-Verb-Object`) order, Hindi follows SOV). However, the sentences have a similar dependency structure, and the constituent words have similar part-of-speech tags. Presumably, language syntax can help to bridge the typological differences across languages.

In recent years, we have seen a colossal effort to pre-train Transformer encoder (Vaswani et al., 2017) on large-scale unlabeled text data in one or many languages. Multilingual encoders, such as mBERT (Devlin et al., 2019) or XLM-R (Conneau et al., 2020) map text sequences into a shared multilingual space by jointly pre-training in many languages. This allows us to transfer the multilingual encoders across languages and have found effective for many NLP applications, including text classification (Bowman et al., 2015; Conneau et al., 2018), question answering (Rajpurkar et al., 2016; Lewis et al., 2020b), named entity recognition (Pires et al., 2019; Wu and Dredze, 2019a), and more. Since the introduction of mBERT, several works (Wu and Dredze, 2019a; Pires et al., 2019; K et al.,

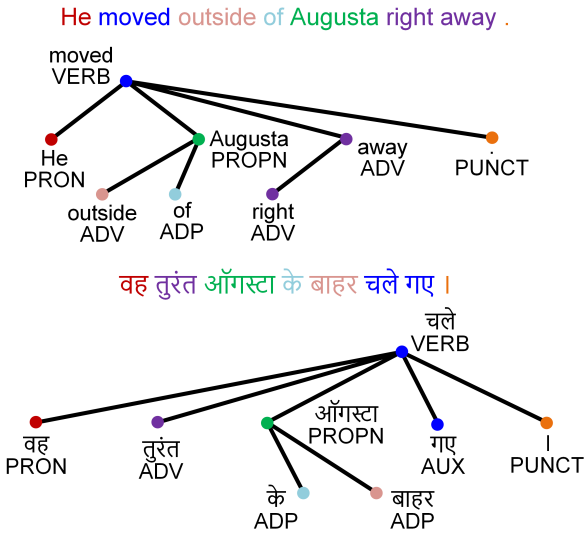


Figure 5.1: Two parallel sentences in English and Hindi from XNLI (Conneau et al., 2018) dataset. The words highlighted with the same color have the same meaning. Although the sentences have a different word order, their syntactic dependency structure is similar.

2020) attempted to reason their success in cross-lingual transfer. In particular, Wu and Dredze (2019a) showed that mBERT captures language syntax that makes it effective for cross-lingual transfer. A few recent works (Hewitt and Manning, 2019; Jawahar et al., 2019; Chi et al., 2020) suggest that BERT learns compositional features; mimicking a tree-like structure that agrees with the Universal Dependencies taxonomy.

However, fine-tuning for the downstream task in a source language may not require mBERT to retain structural features or learn to encode syntax. We argue that encouraging mBERT to learn the correlation between syntax structure and target labels can benefit cross-lingual transfer. To support our argument, we show an example of question answering (QA) in Figure 5.2. In the example, mBERT predicts incorrect answers given the Spanish language context that can be corrected by exploiting syntactic clues. Utilizing syntax structure can also benefit *generalized* cross-lingual transfer (Lewis et al., 2020b) where the input text sequences belong to different languages. For example, answering an English question based on a Spanish passage or predicting text similarity given the two sentences as shown in Figure 5.1. In such a setting, syntactic clues may help to align sentences.

In this work, we propose to augment mBERT with universal language syntax while fine-

Q	English	How many members of the Senate are elected?
	Spanish	Cuántos miembros del Senado son elegidos?
C	English	The Chamber of Deputies has 630 elected members, while the Senate has 315 elected members. . . .
	Spanish	La cámara de los diputados está formada por 630 miembros, mientras que hay 315 senadores más los senadores vitalicios. . . .
A	mBERT	[Q:English-C:English] 315 (✓); [Q:Spanish-C:Spanish] 630 (✗) [Q:Spanish-C:English] 315 (✓); [Q:English-C:Spanish] 630 (✗)
	mBERT + Syn.	[Q:English-C:English] 315 (✓); [Q:Spanish-C:Spanish] 315 (✓) [Q:Spanish-C:English] 315 (✓); [Q:English-C:Spanish] 315 (✓)

Figure 5.2: A parallel QA example in English (en) and Spanish (es) from MLQA Lewis et al. (2020b) with predictions from mBERT and our proposed syntax-augmented mBERT. In "Q:x-C:y", x and y indicates question and context languages, respectively. Based on our analysis of the highlighted tokens' attention weights, we conjecture that mBERT answers 630 as the token is followed by "miembros", while 315 is followed by "senadores" in Spanish.

tuning on downstream tasks. We use a graph attention network (GAT) (Veličković et al., 2018) to learn structured representations of the input sequences that are incorporated into the self-attention mechanism. We adopt an auxiliary objective to train GAT such that it embeds the dependency structure of the input sequence accurately. We perform an evaluation on *zero-shot* cross-lingual transfer for text classification, question answering, named entity recognition, and task-oriented semantic parsing. Experiment results show that augmenting mBERT with syntax improves cross-lingual transfer, such as in PAWS-X and MLQA, by 1.4 and 1.6 points on average across all the target languages. Syntax-augmented mBERT achieves remarkable gain in the generalized cross-lingual transfer; in PAWS-X and MLQA, performance is boosted by 3.9 and 3.1 points on average across all language pairs. Furthermore, we discuss challenges and limitations in modeling universal language syntax. We release the code to help future works.[1]

[1]https://github.com/wasiahmad/Syntax-MBERT

5.2 Syntax-augmented Multilingual BERT

Multilingual BERT (mBERT) (Devlin et al., 2019) enables cross-lingual learning as it embeds text sequences into a shared multilingual space. mBERT is fine-tuned on downstream tasks, e.g., text classification using *monolingual* data and then directly employed to perform on the target languages. This refers to *zero-shot* cross-lingual transfer. Our main idea is to augment mBERT with language syntax for zero-shot cross-lingual transfer. We employ graph attention network (GAT) (Veličković et al., 2018) to learn syntax representations and fuse them into the self-attention mechanism of mBERT.

In this section, we first briefly review the transformer encoder that bases mBERT (§ 5.2.1), and then describe the graph attention network (GAT) that learns syntax representations from dependency structure of text sequences (§ 5.2.2). Finally, we describe how language syntax is explicitly incorporated into the transformer encoder (§ 5.2.3).

5.2.1 Transformer Encoder

Transformer encoder (Vaswani et al., 2017) is composed of an embedding layer and stacked encoder layers. Each encoder layer consists of two sublayers, a multi-head attention layer followed by a fully connected feed-forward layer. We detail the process of encoding an input token sequence $(w_1, \ldots, w_n)$ into a sequence of vector representations $H = [h_1, \ldots, h_n]$ as follows.

Embedding Layer is parameterized by two embedding matrices — the token embedding matrix $W_e \in R^{U \times d_{model}}$ and the position embedding matrix $W_p \in R^{U \times d_{model}}$ (where U is the vocabulary size and d_{model} is the encoder output dimension). An input text sequence enters into the model as two sequences: the token sequence $(w_1, \ldots, w_n)$ and the corresponding absolute position sequence $(p_1, \ldots, p_n)$. The output of the embedding layer is a sequence of vectors $\{x_i\}_{i=1}^{n}$ where $x_i = w_i W_e + p_i W_p$. The vectors are packed into matrix $H^0 = [x_1, \ldots, x_n] \in R^{n \times d_{model}}$ and fed to an L-layer encoder.

Multi-head Attention allows to jointly attend to information from different representation subspaces, known as *attention heads*. Multi-head attention layer composed of h attention heads with the same parameterization structure. At each attention head, the output from the previous layer H^{l-1} is first linearly projected into queries, keys, and values as follows.

$$Q = H^{l-1}W_l^Q, K = H^{l-1}W_l^K, V = H^{l-1}W_l^V,$$

where the parameters $W_l^Q, W_l^K \in R^{d_{model} \times d_k}$ and $W_l^V \in R^{d_{model} \times d_v}$ are unique per attention head. Then scaled dot-product attention is performed to compute the output vectors $\{o_i\}_{i=1}^n \in R^{n \times d_v}$.

$$\begin{aligned} &\text{Attention}(Q, K, V, M, d_k) \\ &= \text{softmax}\left(\frac{QK^T + M}{\sqrt{d_k}}\right)V, \end{aligned} \tag{5.1}$$

where $M \in \mathbb{R}^{n \times n}$ is the masking matrix that determines whether a pair of input positions can attend each other. In classic multi-head attention, M is a *zero* matrix (all positions can attend each other).

The output vectors from all the attention heads are concatenated and projected into d_{model} dimension using the parameter matrix $W_o \in R^{hd_v \times d_{model}}$. Finally the vectors are passed through a feed-forward network to output $H^l \in R^{n \times d_{model}}$.

5.2.2 Graph Attention Network

We embed the syntax structure of the input token sequences using their universal dependency parse. A dependency parse is a directed graph where the nodes represent words, and the edges represent dependencies (the dependency relation between the head and dependent words). We use a graph attention network (GAT) (Veličković et al., 2018) to embed the dependency tree structure of the input sequence. We illustrate GAT in Figure 5.3.

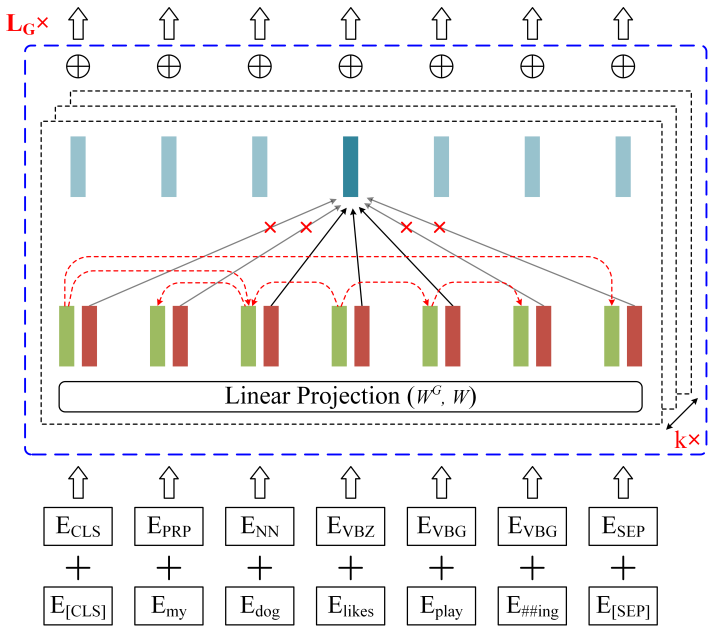


Figure 5.3: A simplified illustration of the multi-head self-attention in the graph attention network wherein each head attention is allowed between words within δ distance from each other in the dependency graph. For example, as shown, in one of the attention heads, the word "likes" is only allowed to attend its adjacent (δ=1) words "dog" and "play".

Given the input sequence, the words (w_i) and their part-of-speech tags (pos_i) are embedded into vectors using two parameter matrices: the token embedding matrix W_e and the part-of-tag embedding W_{pos}. The input sequence is then encoded into an input matrix $\mathcal{G}^0 = [g_1, \ldots, g_n]$, where $g_i = w_i W_e + pos_i W_{pos} \in R^{d_{model}}$. Note that token embedding matrix W_e is shared between GAT and the Transformer encoder. Then $\mathcal{G}^0$ is fed into an $L_{\mathcal{G}}$-layer GAT where each layer generates word representations by attending their adjacent words. GAT uses the multi-head attention mechanism and perform a *dependency-aware* self-attention as

$$\mathcal{O} = \text{Attention}(\mathcal{T}, \mathcal{T}, V, M, d_g) \tag{5.2}$$

namely setting the query and key matrices to be the same $\mathcal{T} \in R^{n \times d_g}$ respectively and

the mask M by

$$M_{ij} = \begin{cases} 0, & D_{ij} \leq \delta \\ -\infty, & \text{otherwise} \end{cases} \tag{5.3}$$

where D is the distance matrix and D_{ij} indicates the shortest path distance between word i and j in the dependency graph structure.

Typically in GAT, δ is set to 1; allowing attention between adjacent words only. However, in our study, we find setting δ to [2, 4] helpful for the downstream tasks. Finally, the vector representations from all the attention heads (as in Eq. (5.2)) are concatenated to form the output representations $\mathcal{G}^l \in R^{n \times kd_g}$, where k is the number of attention heads employed. The goal of the GAT encoder is to encode the dependency structure into vector representations. Therefore, we design GAT to be light-weight; consisting of much less parameters in comparison to Transformer encoder. Note that, GAT does not employ positional representations and only consists of multi-head attention; there is no feed-forward sublayer and residual connections.

Dependency Tree over Wordpieces and Special Symbols mBERT tokenizes the input sequence into subword units, also known as wordpieces. Therefore, we modify the dependency structure of linguistic tokens to accommodate wordpieces. We introduce additional dependencies between the first subword (head) and the rest of the subwords (dependents) of a linguistic token. More specifically, we introduce new edges from the head subword to the dependent subwords. The inputs to mBERT use special symbols: [CLS] and [SEP]. We add an edge from the [CLS] token to the root of the dependency tree and the [SEP] tokens.

5.2.3 Syntax-augmented Transformer Encoder

We want the Transformer encoder to consider syntax structure while performing the self-attention between input sequence elements. We use the syntax representations produced

by GAT (outputs from the last layer, denoting as $\mathcal{G}$) to *bias* the self-attention.

$$O = \text{Attention}(Q + \mathcal{G}G_l^Q, K + \mathcal{G}G_l^K, V, M, d_k),$$

where $G_l^Q, G_l^K \in R^{d_{kd_g} \times d_k}$ are new parameters that learn representations to bias the self-attention. We consider the addition terms $(\mathcal{G}G_l^Q, \mathcal{G}G_l^K)$ as syntax-bias that provide syntactic clues to guide the self-attention. The high-level intuition behind the syntax bias is to attend tokens with a specific part-of-speech tag sequence or dependencies.[2]

Syntax-heads mBERT employs h (=12) attention heads and the syntax representations can be infused into one or more of these heads, and we refer them as *syntax-heads*. In our experiments, we observed that instilling structural information into many attention heads degenerates the performance. For the downstream tasks, we consider one or two syntax-heads that gives the best performance.[3]

Syntax-layers refers to the encoder layers that are infused by syntax representations from GAT. mBERT has a 12-layer encoder and our study finds considering all of the layers as *syntax-layers* beneficial for cross-lingual transfer.

5.2.4 Fine-tuning

We jointly fine-tune mBERT and GAT on downstream tasks in the source language (English in this work) following the standard procedure. However, the task-specific training may not guide GAT to encode the tree structure. Therefore, we adopt an auxiliary objective that supervises GAT to learn representations which can be used to decode the tree structure. More specifically, we use GAT's output representations

[2]In example shown in Figure 5.2, token dependencies: [en: root → has → has → members → 315], and [es: root → formada → hay → senadores → 315] or corresponding part-of-speech tag sequence [VERB → VERB → NOUN → NUM]) may help mBERT to predict the correct answer.

[3]This aligns with the findings of Hewitt and Manning (2019) as they showed 64 or 128 dimension of the contextual representations are sufficient to capture the syntax structure.

Dataset	Task	|Train|	|Dev|	|Test|	|Lang|	Metric
XNLI	Classification	392K	2.5K	5K	13	Accuracy
PAWS-X	Classification	49K	2K	2K	7	Accuracy
MLQA	QA	87K	34K	4.5K-11K	7	F1 / Exact Match
XQuAD	QA	87K	34K	1190	10	F1 / Exact Match
Wikiann	NER	20K	10K	1K-10K	15	F1
CoNLL	NER	15K	2K-3K	1.5K-5K	4	F1
mTOP	Semantic Parsing	15.7K	2.2K	2.8K-4.4K	5	Exact Match
mATIS++	Semantic Parsing	4.5K	490	893	9	Exact Match

Table 5.1: Statistics of the evaluation datasets. |Train|, |Dev| and |Test| are the numbers of examples in the training, dev and test sets, respectively. For train set, the number is for the source language, English, while for dev and test set, the number is for each target language. |Lang| is the number of target languages we consider for each task.

$\mathcal{G} = [g_1, \ldots, g_n]$ to predict the tree distance between all pairs of words (g_i, g_j) and the tree depth $||g_i||$ of each word w_i in the input sequence. Following Hewitt and Manning (2019), we apply a linear transformation $\theta_1 \in R^{m \times kd_g}$ to compute squared distances as follows.

$$d_{\theta_1}(g_i, g_j)^2 = (\theta_1(g_i - g_j))^T(\theta_1(g_i - g_j)).$$

The parameter matrix θ_1 is learnt by minimizing:

$$\min_{\theta_1} \sum_s \frac{1}{n^2} \sum_{i,j} |dist(w_i, w_j)^2 - d_\theta(g_i, g_j)^2|,$$

where s denotes all the text sequences in the training corpus. Similarly, we train another parameter matrix θ_2 to compute squared vector norms, $d_{\theta_2}(g_i) = (\theta_2 g_i)^T(\theta_2 g_i)$ that characterize the tree depth of the words. We train GAT's parameters and θ_1, θ_2 by minimizing the loss: $\mathcal{L} = \mathcal{L}_{task} + \alpha(\mathcal{L}_{dist} + \mathcal{L}_{depth})$, where α is weight for the tree structure prediction loss.

Pre-training GAT Unlike mBERT's parameters, GAT's parameters are trained from scratch during task-specific fine-tuning. For low-resource tasks, GAT may not learn to encode the syntax structure accurately. Therefore, we utilize the universal dependency parses (Nivre et al., 2019) to pre-train GAT on the source and target languages. Note

that, the pre-training objective for GAT is to predict the tree distances and depths as described above.

5.3 Experiment Setup

To study syntax-augmented mBERT's performance in a broader context, we perform an evaluation on four NLP applications: text classification, named entity recognition, question answering, and task-oriented semantic parsing. Our evaluation focuses on assessing the usefulness of utilizing universal syntax in the *zero-shot* cross-lingual transfer.

5.3.1 Evaluation Tasks

Text Classification We conduct experiments on two widely used cross-lingual text classification tasks: (i) natural language inference and (ii) paraphrase detection. We use the XNLI (Conneau et al., 2018) and PAWS-X (Yang et al., 2019a) datasets for the tasks, respectively. In both tasks, a pair of sentences is given as input to mBERT. We combine the dependency tree structure of the two sentences by adding two edges from the [CLS] token to the roots of the dependency trees.

Named Entity Recognition is a structure prediction task that requires to identify the named entities mentioned in the input sentence. We use the Wikiann dataset (Pan et al., 2017) and a subset of two tasks from CoNLL-2002 (Tjong Kim Sang, 2002) and CoNLL-2003 NER (Tjong Kim Sang and De Meulder, 2003). We collect the CoNLL datasets from XGLUE (Liang et al., 2020). In both datasets, there are 4 types of named entities: Person, Location, Organization, and Miscellaneous.[4]

Question Answering We evaluate on two cross-lingual question answering benchmarks, MLQA (Lewis et al., 2020b), and XQuAD (Artetxe et al., 2020). We use the SQuAD dataset (Rajpurkar et al., 2016) for training and validation. In the QA task, the inputs

[4]Miscellaneous entity type covers named entities that do not belong to the other three types

are a question and a context passage that consists of many sentences. We formulate QA as a multi-sentence reading comprehension task; jointly train the models to predict the answer sentence and extract the answer span from it. We concatenate the question and each sentence from the context passage and use the [CLS] token representation to score the candidate sentences. We adopt the confidence method from Clark and Gardner (2018) and pick the highest-scored sentence to extract the answer span during inference. We provide more details of the QA models in Appendix.

Task-oriented Semantic Parsing The fourth evaluation task is cross-lingual task-oriented semantic parsing. In this task, the input is a user utterance and the goal is to predict the intent of the utterance and fill the corresponding slots. We conduct experiments on two recently proposed benchmarks: (i) mTOP (Li et al., 2021) and (ii) mATIS++ (Xu et al., 2020). We jointly train the BERT models as suggested in Chen et al. (2019a).

We summarize the evaluation task benchmark datasets and evaluation metrics in Table 5.1.

5.3.2 Implementation Details

We collect the universal part-of-speech tags and the dependency parse of sentences by pre-processing the datasets using UDPipe.[5] We fine-tune mBERT on the pre-processed datasets and consider it as the baseline for our proposed syntax-augmented mBERT. We extend the XTREME framework (Hu et al., 2020) that is developed based on `transformers` API (Wolf et al., 2020). We use the same hyper-parameter setting for mBERT models, as suggested in XTREME. For the graph attention network (GAT), we set $L_G = 4, k = 4$, and $d_g = 64$ (resulting in $\sim$0.5 million parameters). We tune δ[6] (shown in Eq. (5.3))

[5]https://ufal.mff.cuni.cz/udpipe/2

[6]We observed that the value of δ depends on the downstream task and the source language. For example, a larger δ value is beneficial for tasks taking a pair of text sequences as inputs, while a smaller δ value results in better performances for tasks taking single text input. Experiments on PAWS-X using each target language as the source language indicate that δ should be set to a larger value for source

Model	en	ar	bg	de	el	es	fr	hi	ru	tr	ur	vi	zh	ko	ja	nl	pt	AVG
Classification - XNLI Conneau et al. (2018)																		
[1]	80.8	64.3	68.0	70.0	65.3	73.5	73.4	58.9	67.8	60.9	57.2	69.3	67.8	-	-	-	-	67.5
mBERT	81.8	63.8	68.0	70.7	65.4	73.8	72.4	59.3	68.4	60.7	56.7	68.6	67.8	-	-	-	-	67.5
+ Syn.	81.6	**65.4**	**69.3**	70.7	**66.5**	**74.1**	**73.2**	**60.5**	68.8	**62.4**	**58.7**	**69.9**	**69.3**	-	-	-	-	68.5
Classification - PAWS-X Yang et al. (2019a)																		
[1]	94.0	-	-	85.7	-	87.4	87.0	-	-	-	-	-	77.0	69.6	73.0	-	-	82.0
mBERT	93.9	-	-	85.7	-	88.4	87.6	-	-	-	-	-	78.0	73.6	73.1	-	-	82.9
+ Syn.	94.0	-	-	85.9	-	**89.1**	**88.2**	-	-	-	-	-	**80.7**	**76.3**	**75.8**	-	-	84.3
NER - Wikiann Pan et al. (2017)																		
[1]	85.2	41.1	77.0	78.0	72.5	77.4	79.6	65.0	64.0	71.8	36.9	71.8	-	59.6	-	81.8	80.8	69.5
mBERT	83.6	38.8	77.0	76.0	70.4	74.7	78.9	63.4	63.5	70.9	37.7	73.5	-	59.3	-	81.9	78.7	68.5
+ Syn.	84.4	**40.0**	77.0	**77.0**	**71.5**	**76.1**	79.3	64.2	63.8	71.4	37.3	72.7	-	59.3	-	81.9	79.0	69.0
NER - CoNLL Tjong Kim Sang (2002); Tjong Kim Sang and De Meulder (2003)																		
[2]	90.6	-	-	69.2	-	75.4	-	-	-	-	-	-	-	-	-	77.9	-	78.2
mBERT	90.7	-	-	68.3	-	74.5	-	-	-	-	-	-	-	-	-	77.6	-	77.8
+ Syn.	90.6	-	-	**69.1**	-	73.6	-	-	-	-	-	-	-	-	-	**78.5**	-	78.0
QA - MLQA Lewis et al. (2020b)																		
[3]	77.7	45.7	-	57.9	-	64.3	-	43.8	-	-	-	57.1	57.5	-	-	-	-	57.7
mBERT	80.5	47.2	-	59.0	-	63.9	-	47.5	-	-	-	56.5	56.6	-	-	-	-	58.7
+ Syn.	80.4	**48.9**	-	**60.8**	-	**65.9**	-	46.7	-	-	-	**59.3**	**60.1**	-	-	-	-	60.3
QA - XQuAD Artetxe et al. (2020)																		
[1]	83.5	61.5	-	70.6	62.6	75.5	-	59.2	71.3	55.4	-	69.5	58.0	-	-	-	-	66.7
mBERT	84.2	54.8	-	68.9	60.2	71.1	-	55.7	68.6	48.9	-	64.0	57.2	-	-	-	-	63.4
+ Syn.	84.0	55.5	-	**71.4**	**61.3**	**72.8**	-	54.6	68.4	**49.8**	-	**67.6**	56.1	-	-	-	-	64.2
Semantic Parsing - mTOP Li et al. (2021)																		
mBERT	81.0	-	-	28.1	-	40.2	38.8	9.8	-	-	-	-	-	-	-	-	-	39.6
+ Syn.	81.3	-	-	**30.0**	-	**43.0**	**41.2**	**11.5**	-	-	-	-	-	-	-	-	-	41.4
Semantic Parsing - mATIS++ Xu et al. (2020)																		
mBERT	86.0	-	-	38.1	-	43.7	36.9	16.2	-	1.3	-	-	7.8	-	28.2	-	38.2	32.9
+ Syn.	86.2	-	-	**40.1**	-	**44.5**	**38.9**	**18.7**	-	1.5	-	-	8.0	-	27.3	-	37.3	33.6

Table 5.2: Cross-lingual transfer results for all the evaluation tasks (on test set) across 17 languages. We report F1 score for the question answering (QA) datasets (for other datasets, see Table 5.1). We train and evaluate mBERT on the same pre-processed datasets and considers its performance as the *baseline* (denoted by "mBERT" rows in the table) for syntax-augmented mBERT (denoted by "+ Syn." rows in the table). Bold-faced values indicate that the syntax-augmented mBERT is statistically significantly better (by paired bootstrap test, $p < 0.05$) than the *baseline*. We include results from published works ([1]: Hu et al. (2020), [2]: Liang et al. (2020), and [3]: Lewis et al. (2020b)) as a reference. Except for the QA datasets, all our results are averaged over three different seeds.

and α (weight of the tree structure prediction loss) in the range $[1, 2, 4, 8]$ and $[0.5 - 1.0]$, respectively. We detail the hyper-parameters in the Appendix.

language with longer text sequences (e.g., Arabic) and vice versa.

s_1/s_2	en	de	es	fr	ja	ko	zh
en	-	0.7	1.6	1.4	4.7	2.5	5.4
de	0.5	-	2.0	2.1	5.1	3.5	5.9
es	1.0	2.1	-	1.7	4.6	3.0	6.6
fr	0.9	1.7	1.9	-	5.0	2.7	5.4
ja	5.2	5.3	5.6	5.1	-	5.9	5.1
ko	3.1	2.8	4.3	3.9	6.4	-	5.1
zh	5.8	5.5	6.3	6.0	6.1	4.5	-

(a) PAWS-X

q/c	en	es	de	ar	hi	vi	zh
en	-	-0.2	0.3	0.4	0.9	0.6	1.1
es	4.1	-	3.5	5.4	5.3	7.3	7.6
de	3.5	2.8	-	4.0	2.9	4.0	5.0
de	1.8	2.4	1.1	-	-0.1	6.2	4.4
hi	1.0	1.8	0.5	0.2	-	-0.6	1.0
vi	5.6	4.5	5.5	6.9	4.2	-	5.5
zh	3.8	3.3	4.4	2.4	0.9	5.4	-

(b) MLQA

Table 5.3: The performance difference between syntax-augmented mBERT and mBERT in the *generalized* cross-lingual transfer setting. The rows and columns indicate (a) language of the first and second sentences in the candidate pairs and (b) context and question languages. The gray cells have a value greater than or equal to the average performance difference, which is 3.9 and 3.1 for (a) and (b).

5.4 Experiment Results

We aim to address the following questions.

1. Does augmenting mBERT with syntax improve (generalized) cross-lingual transfer?
2. Does incorporating syntax benefit specific languages or language families?
3. Which NLP tasks or types of tasks get more benefits from utilizing syntax?

5.4.1 Cross-lingual Transfer

Experiment results to compare mBERT and syntax-augmented mBERT are presented in Table 5.2. Overall, the incorporation of language syntax in mBERT improves cross-lingual transfer for the downstream tasks, in many languages by a significant margin ($p < 0.05$, t-test). The average performances across all languages on XNLI, PAWS-X, MLQA, and mTOP benchmarks improve significantly (by at least 1 point). On the other benchmarks: Wikiann, CoNLL, XQuAD, and mATIS++, the average performance improvements are 0.5, 0.2, 0.8, and 0.7 points, respectively. Note that the performance gains in the source language (English) for all the datasets except Wikiann is ≤ 0.3. This indicates that cross-lingual transfer gains are not due to improving the downstream tasks, but instead, language syntax helps to transfer across languages.

5.4.2 Generalized Cross-lingual Transfer

In the generalized cross-lingual transfer setting (Lewis et al., 2020b), the input text sequences for the downstream tasks (e.g., text classification, QA) may come from different languages. As shown in Figure 5.2, given the context passage in English, a multilingual QA model should answer the question written in Spanish. Due to the parallel nature of the existing benchmark datasets: XNLI, PAWS-X, MLQA, and XQuAD, we evaluate mBERT and its' syntax-augmented variant on the generalized cross-lingual transfer setting. The results for PAWS-X and MLQA are presented in Table 5.3 (results for the other datasets are provided in Appendix).

In both text classification and QA benchmarks, we observe significant improvements for most language pairs. In the PAWS-X text classification task, language pairs with different typologies (e.g., en-ja, en-zh) have the most gains. When Chinese (zh) or Japanese (ja) is in the language pairs, the performance is boosted by at least 4.5%. The dataset characteristics explain this; the task requires modeling structure, context, and word order information. On the other hand, in the XNLI task, the performance gain pattern is scattered, and this is perhaps syntax plays a less significant role in the XNLI task. The largest improvements result when the languages of the premise and hypothesis sentences belong to {Bulgarian, Chinese} and {French, Arabic}.

In both QA datasets, syntax-augmented mBERT boosts performance when the question and context languages are typologically different except the Hindi language. Surprisingly, we observe a large performance gain when questions in Spanish and German are answered based on the English context. Based on our manual analysis on MLQA, we suspect that although questions in Spanish and German are translated from English questions (by human), the context passages are from Wikipedia that often are not exact translation of the corresponding English passage. Take the context passages in Figure 5.2 as an example. We anticipate that syntactic clues help a QA model in identifying the correct answer span when there are more than one semantically equivalent and plausible answer choices.

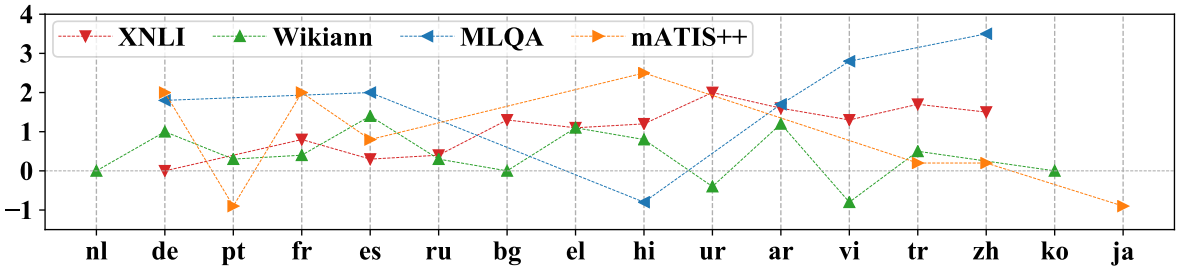


Figure 5.4: Performance improvements for XNLI, Wikiann, MLQA, and mATIS++ across languages.The languages in x-axis are grouped by language families: IE.Germanic (nl, de), IE.Romance (pt, fr, es), IE.Slavic (ru, bg), IE.Greek (el), IE.Indic (hi, ur), Afro-asiatic (ar, vi), Altaic (tr), Sino-tibetan (zh), Korean (ko), and Japanese (ja).

5.4.3 Analysis & Discussion

We discuss and analyze our findings on the following points based on the empirical results.

Impact on Languages We study if fine-tuning syntax-augmented mBERT on English (source language) impacts specific target languages or families of languages. We show the performance gains on the target languages grouped by their families in four downstream tasks in Figure 5.4. There is no observable trend in the overall performance improvements across tasks. However, the XNLI curve weakly indicates that when target languages are typologically different from the source language, there is an increase in the transfer performance (comparing left half to the right half of the curve).

Impact of Pre-training GAT Before fine-tuning syntax-augmented mBERT, we pre-train GAT on the 17 target languages (discussed in § 5.2.4). In our experiments, we observe such pre-training boosts semantic parsing performance, while there is a little gain on the classification and QA tasks. We also observe that pre-training GAT diminishes the gain of fine-tuning with the auxiliary objective (predicting the tree structure). We hypothesize that pre-training or fine-tuning GAT using auxiliary objective helps when there is limited training data. For example, semantic parsing benchmarks have a small number of training examples, while XNLI has many. As a result, the improvement due to pre-training or fine-tuning GAT in the semantic parsing tasks is significant, and in the

XNLI task, it is marginal.

Discussion To foster research in this direction, we discuss additional experiment findings.

• A natural question is, instead of using GAT, why we do not modify attention heads in mBERT to embed the dependency structure (as shown in Eq. 5.3). We observed a consistent performance drop across all the tasks if we intervene in self-attention (blocking pair-wise attention). We anticipate fusing GAT encoded syntax representations helps as it adds bias to the self-attention. For future works, we suggest exploring ways of adding structure bias, e.g., scaling attention weights based on dependency structure (Bugliarello and Okazaki, 2020).

• Among the evaluation datasets, Wikiann consists of sentence fragments, and the semantic parsing benchmarks consist of user utterances that are typically short in length. Sorting and analyzing the performance improvements based on sequence lengths suggests that the utilization of dependency structure has limited scope for shorter text sequences. However, part-of-speech tags help to identify span boundaries improving the slot filling tasks.

5.4.4 Limitations and Challenges

In this work, we assume we have access to an off-the-shelf universal parser, e.g., UDPipe (Straka and Straková, 2017) or Stanza (Qi et al., 2020) to collect part-of-speech tags and the dependency structure of the input sequences. Relying on such a parser has a limitation that it may not support all the languages available in benchmark datasets, e.g., we do not consider Thai and Swahili languages in the benchmark datasets.

There are a couple of challenges in utilizing the universal parsers. First, universal parsers tokenize the input sequence into words and provide part-of-speech tags and dependencies for them. The tokenized words may not be a part of the input.[7] As a result,

[7]For example, in the German sentence "Wir gehen zum kino" (we are going to the cinema), the token "zum" is decomposed into words "zu" and "dem".

tasks requiring extracting text spans (e.g., QA) need additional mapping from input tokens to words. Second, the parser's output word sequence is tokenized into wordpieces that often results in inconsistent wordpieces resulting in degenerated performance in the downstream tasks.[8]

5.5 Related Work

Syntax-aware Multi-head Attention A large body of prior works investigated the advantages of incorporating language syntax to enhance the self-attention mechanism (Vaswani et al., 2017). Existing techniques can be broadly divided into two types. The first type of approach relies on an external parser (or human annotation) to get a sentence's dependency structure during inference. This type of approaches embed the dependency structure into contextual representations which benefits the target NLP task, e.g., information extraction (Ahmad et al., 2021c; Sachan et al., 2021), semantic role labeling (Zhang et al., 2019c), question answering (Zhang et al., 2020), and machine translation (Wu et al., 2017a; Chen et al., 2017; Wang et al., 2019a,b; Bugliarello and Okazaki, 2020). Our proposed method falls under this category; however, unlike prior works, our study investigates if fusing the universal dependency structure into the self-attention of existing multilingual encoders help cross-lingual transfer. Graph attention networks (GATs) that use multi-head attention has also been adopted for NLP tasks, such as text classification (Huang and Carley, 2019) also fall into this category. The second category of approaches does not require the syntax structure of the input text during inference. These approaches are trained to predict the dependency parse via supervised learning (Strubell et al., 2018; Deguchi et al., 2019). For example, Strubell et al. (2018) introduced linguistically-informed self-attention (LISA); trains self-attention via multi-task learning combining the target task with dependency parsing.

[8]This happen for languages, such as Arabic as parsers normalize the input that lead to inconsistent characters between input text and the output tokenized text.

Encoding Syntax for Language Transfer Universal language syntax, e.g., part-of-speech (POS) tags, dependency parse structure, and relations are shown to be helpful for cross-lingual transfer (Kozhevnikov and Titov, 2013; Pražák and Konopík, 2017; Wu et al., 2017a; Subburathinam et al., 2019b; Liu et al., 2019b; Zhang et al., 2019c; Xie et al., 2020; Ahmad et al., 2021c). Many of these prior works utilized graph neural networks (GNN) to encode the dependency graph structure of the input sequences. In this work, we utilize graph attention networks (GAT) (Veličković et al., 2018), a variant of GNN that employs the multi-head attention mechanism.

5.6 Summary

In this chapter, we presented an approach to incorporate universal language syntax into multilingual BERT (mBERT) by infusing structured representations into its multi-head attention mechanism. We employ a modified graph attention network to encode the syntax structure of the input sequences. The results endorse the effectiveness of our proposed approach in the cross-lingual transfer. We discuss limitations and challenges to drive future works.

CHAPTER 6

Representation Learning using Unlabeled Data

6.1 Introduction

Representation learning using unlabeled text data has been the fundamental theme to make the modern NLP models transferable across languages. The feature space learned by cross-lingual representation learning techniques often embeds language-dependent features that hinder cross-lingual transfer. Removal of such language-dependent features from the representation spaces can facilitate cross-lingual transfer learning. Since unlabeled text data comes at no price, we can utilize them to design language-agnostic representation learning techniques. The first half of this chapter is based on Ahmad et al. (2019b). In that work, we propose leveraging unannotated sentences from auxiliary languages to help learn language-agnostic representations. Specifically, we present an adversarial training technique for learning contextual encoders that produce invariant representations across languages to facilitate the cross-lingual transfer. We conduct experiments on cross-lingual dependency parsing where we train a dependency parser on a source language and transfer it to a wide range of target languages. Experiments on 28 target languages demonstrate that adversarial training significantly improves transfer performances under several different settings.

Pre-trained language representations have been the key ingredient for transfer learning in NLP. The success of leveraging unlabeled text data to learn language representations for NLP encouraged researchers to learn representations for natural and programming languages jointly. In the second half of this chapter, we present PLBART (Ahmad et al., 2021a), a sequence-to-sequence model capable of performing a broad spectrum of program

and language understanding and generation tasks. PLBART is pre-trained on an extensive collection of Java and Python functions and associated NL text via denoising autoencoding. We show that PLBART outperforms or rivals state-of-the-art models on code summarization in English, code generation, and code translation in seven programming languages. Moreover, PLBART performs effectively in program understanding tasks, *e.g.,* program repair, clone detection, and vulnerable code detection. We also show that PLBART learns program syntax, style (*e.g.,* identifier naming convention), logical flow (*e.g., if* block inside an *else* block is equivalent to *else if* block) that are crucial to program semantics and thus excels even with limited annotations.

6.2 Language-agnostic Representation Learning

A typical NLP model consists of a representation learning component, also known as encoders that convert input text sequences into contextualized representations. In cross-lingual transfer, most recent approaches assume that the inputs from different languages are aligned into the same embedding space via multilingual word embeddings or multilingual contextualized word vectors that are fed into the encoder, such that the an NLP model trained on a source language can be transferred to target languages. However, when training a model on the source language, the encoder not only learns to embed a sentence but it also carries language-specific properties, such as word order typology. Therefore, the parser suffers when it is transferred to a language with different language properties. Motivated by this, we study how to train an encoder for generating language-agnostic representations that can be transferred across a wide variety of languages.

We propose to utilize *unlabeled corpora* of one or more auxiliary languages to train an encoder that learns language-agnostic contextual representations of sentences to facilitate cross-lingual transfer. To utilize the unlabeled auxiliary language corpora, we adopt adversarial training (Goodfellow et al., 2014) of the encoder and a classifier that predicts the language identity of an input sentence from its encoded representation produced by the encoder. The adversarial training encourages the encoder to produce language

invariant representations such that the language classifier fails to predict the correct language identity. As the encoder is jointly trained with a loss for the primary task on the source language and adversarial loss on all languages, we hypothesize that it will learn to capture task-specific features as well as generic structural patterns applicable to many languages, and thus have better transferability.

To verify the proposed approach, we conduct experiments on neural dependency parsers trained on English (source language) and directly transfer them to 28 target languages, with or without the assistance of unlabeled data from auxiliary languages. We chose dependency parsing as the primary task since it is one of the core NLP applications and the development of Universal Dependencies (Nivre et al., 2016) provides consistent annotations across languages, allowing us to investigate transfer learning in a wide range of languages. Thorough experiments and analyses are conducted to address a couple of research questions: (1) Does encoder trained with adversarial training generate language-agnostic representations? and (2) Does language-agnostic representations improve cross-language transfer?

6.2.1 Training Language-agnostic Encoders

We study cross-lingual transfer for dependency parsing. A dependency parser consists of (1) an encoder that transforms an input text sequence into latent representations and (2) a decoding algorithm that generates the corresponding parse tree. We train the encoder of a dependency parser in an *adversarial* fashion to guide it to avoid capturing language-specific information. In particular, we introduce a language identification task where a classifier predicts the language identity (id) of an input sentence from its encoded representation. Then the encoder is trained such that the classifier fails to predict the language id while the parser decoder predicts the parse tree accurately from the encoded representation. We hypothesize that such an encoder would have better cross-lingual transferability. The overall architecture of our model is illustrated in Figure 6.1. In the following, we present the details of the model and training method.

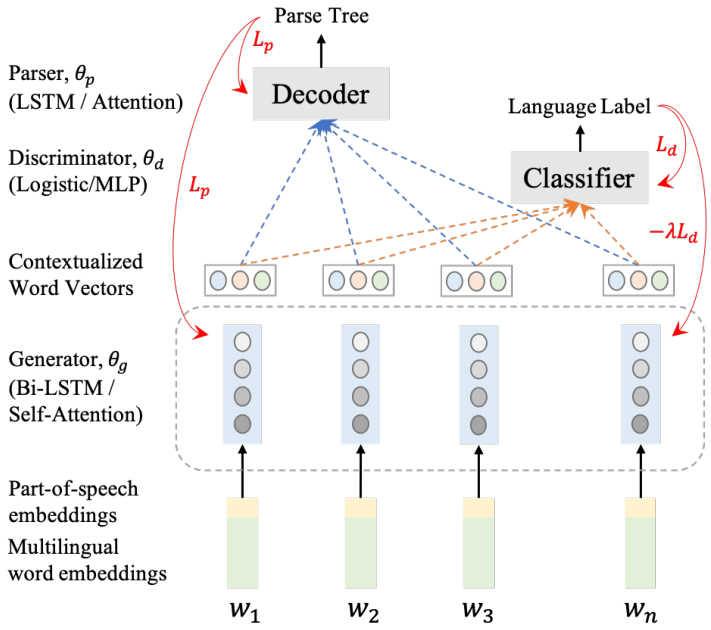


Figure 6.1: An overview of our experimental model consists of three basic components: (1) Encoder, (2) (Parsing) Decoder, and (3) (Language) Classifier. We also show how parsing and adversarial losses (L_p and L_d) are back propagated for parameter updates.

6.2.2 Proposed Method

Our model consists of three basic components, (1) a general encoder, (2) a decoder for parsing, and (3) a classifier for language identification. The encoder learns to generate contextualized representations for the input sentence (a word sequence) which are fed to the decoder and the classifier to predict the dependency structure and the language identity (id) of that sentence. The encoder and the decoder jointly form the parsing model and we consider two alternatives[1] from (Ahmad et al., 2019c): "SelfAtt-Graph" and "RNN-Stack". The "SelfAtt-Graph" parser consists of a modified self-attentional encoder (Shaw et al., 2018a) and a graph-based deep bi-affine decoder (Dozat and Manning, 2017), while the "RNN-Stack" parser is composed of a Recurrent Neural Network (RNN) based encoder and a stack-pointer decoder (Ma et al., 2018).

We stack a classifier (a linear classifier or a multi-layer Perceptron (MLP)) on top

[1](Ahmad et al., 2019c) studied order-sensitive and order-free models and their performances in cross-lingual transfer. In this work, we adopt two typical ones and study the effects of adversarial training on them.

Algorithm 1 Training procedure.

Parameters to be trained: Encoder (θ_g), Decoder (θ_p), and Classifier (θ_d)
X^a = Annotated source language data
X^b = Unlabeled auxiliary language data
I = Number of warm-up iterations
k = Number of learning steps for the discriminator (D) at each iteration
λ = Coefficient of $\mathcal{L}_d$
α_1, α_1 = learning rate; B = Batch size

Require:

for $j = 0, \cdots, I$ **do**
    Update $\theta_g := \theta_g - \alpha_1 \nabla_{\theta_g} \mathcal{L}_p$
    Update $\theta_p := \theta_p - \alpha_1 \nabla_{\theta_p} \mathcal{L}_p$
for $j = I, \cdots, num\_iter$ **do**
    for k steps **do**
        $(x_a^i)_{i=1}^{B/2} \leftarrow$ Sample a batch from X^a
        $(x_b^i)_{i=1}^{B/2} \leftarrow$ Sample a batch from X^b
        Update $\theta_d := \theta_d - \alpha_2 \nabla_{\theta_d} \mathcal{L}_d$
    Total loss $\mathcal{L} := \mathcal{L}_p - \lambda \mathcal{L}_d$
    Update $\theta_g := \theta_g - \alpha_1 \nabla_{\theta_g} \mathcal{L}$
    Update $\theta_p := \theta_p - \alpha_1 \nabla_{\theta_p} \mathcal{L}$

of the encoder to perform the language identification task. The identification task can be framed as either a word- or sentence-level classification task. For the sentence-level classification, we apply average pooling[2] on the contextual word representations generated by the encoder to form a fixed-length representation of the input sequence, which is fed to the classifier. For the word-level classification, we perform language classification for each token individually. In this work, following the terminology in adversarial learning literature, we interchangeably call the encoder as the generator, G and the classifier as the discriminator, D.

Training

Algorithm 1 describes the training procedure. We have two types of loss functions: $\mathcal{L}_p$ for the parsing task and $\mathcal{L}_d$ for the language identification task. For the former, we update

[2]We also experimented with max-pooling and weighted pooling but average pooling resulted in stable performance.

the encoder and the decoder as in the regular training of a parser. For the latter, we adopt adversarial training to update the encoder and the classifier. We present the detailed training schemes in the following.

Parsing To train the parser, we adopt both cross-entropy objectives for these two types of parsers as in (Dozat and Manning, 2017; Ma et al., 2018). The encoder and the decoder are jointly trained to optimize the probability of the dependency trees (y) given sentences (x):

$$\mathcal{L}_p = -\log p(y|x).$$

The probability of a tree can be further factorized into the products of the probabilities of each token's (m) head decision ($h(m)$) for the graph-based parser, or the probabilities of each transition step decision (t_i) for the transition-based parser:

$$\begin{aligned} \text{Graph:} \quad & \mathcal{L}_p = -\sum\nolimits_m \log p(h(m)|x, m), \\ \text{Transition:} \quad & \mathcal{L}_p = -\sum\nolimits_i \log p(t_i|x, t_{<i}). \end{aligned}$$

Language Identification Our objective is to train the contextual encoder in a dependency parsing model such that it encodes language specific features as little as possible, which may help cross-lingual transfer. To achieve our goal, we utilize adversarial training by employing unlabeled auxiliary language corpora.

Setup We adopt the basic generative adversarial network (GAN) for the adversarial training. We assume that X^a and X^b be the corpora of the source and auxiliary language sentences, respectively. The discriminator acts as a binary classifier and is adopted to distinguish the source and auxiliary languages. For the training of the discriminator, weights are updated according to the original classification loss:

$$\mathcal{L}_d = \mathbb{E}_{x\sim X^a}[\log\ D(G(x)] + \mathbb{E}_{x\sim X^b}[\log\ (1 - D(G(x))].$$

Language Families	Languages
Afro-Asiatic	Arabic (ar), Hebrew (he)
Austronesian	Indonesian (id)
IE.Baltic	Latvian (lv)
IE.Germanic	Danish (da), Dutch (nl), English (en), German (de), Norwegian (no), Swedish (sv)
IE.Indic	Hindi (hi)
IE.Latin	Latin (la)
IE.Romance	Catalan (ca), French (fr), Italian (it), Portuguese (pt), Romanian (ro), Spanish (es)
IE.Slavic	Bulgarian (bg), Croatian (hr), Czech (cs), Polish (pl), Russian (ru), Slovak (sk), Slovenian (sl), Ukrainian (uk)
Korean	Korean (ko)
Uralic	Estonian (et), Finnish (fi)

Table 6.1: The selected languages grouped by language families. "IE" is the abbreviation of Indo-European.

For the training of dependency parsing, the generator, G collaborates with the parser but acts as an adversary with respect to the discriminator. Therefore, the generator weights (θ_g) are updated by minimizing the loss function,

$$\mathcal{L} = \mathcal{L}_p - \lambda \mathcal{L}_d,$$

where λ is used to scale the discriminator loss ($\mathcal{L}_d$). In this way, the generator is guided to build language-agnostic representations in order to fool the discriminator while being helpful for the parsing task. Meanwhile, the parser can be guided to rely more on the language-agnostic features.

6.2.3 Experiments and Analysis

In this section, we discuss our experiments and analysis on cross-lingual dependency parsing transfer from a variety of perspectives and show the advantages of adversarial training.

Settings In our experiments, we study single-source parsing transfer, where a parsing model is trained on one source language and directly applied to the target languages. We conduct experiments on the Universal Dependencies (UD) Treebanks (v2.2) (Nivre et al., 2018) using 29 languages, as shown in Table 6.1. We use the publicly available implementation[3] of the "SelfAtt-Graph" and "RNN-Stack" parsers.[4] (Ahmad et al., 2019c) show that the "SelfAtt-Graph" parser captures less language-specific information and performs better than the 'RNN-Stack" parser for distant target languages. Therefore, we use the "SelfAtt-Graph" parser in most of our experiments. Besides, the multilingual variant of BERT (mBERT) (Devlin et al., 2018) has shown to perform well in cross-lingual tasks (Wu and Dredze, 2019b) and outperform the models trained on multilingual word embeddings by a large margin. Therefore, we consider conducting experiments with both multilingual word embeddings and mBERT. We use aligned multilingual word embeddings (Smith et al., 2017; Bojanowski et al., 2017b) with 300 dimensionss or contextualized word representations provided by multilingual BERT[5] (Devlin et al., 2018) with 768 dimensions as the word representations. In addition, we use the Gold universal POS tags to form the input representations.[6] We freeze the word representations during training to avoid the risk of disarranging the multilingual representation alignments.

We select six auxiliary languages[7] (French, Portuguese, Spanish, Russian, German, and Latin) for unsupervised language adaptation via adversarial training. We tune the scaling parameter λ in the range of $[0.1, 0.01, 0.001]$ on the source language validation set and report the test performance with the best value. For gradient reversal (GR) and GAN based adversarial objectives, we use Adam (Kingma and Ba, 2015) to optimize the

[3]https://github.com/uclanlp/CrossLingualDepParser

[4]We adopt the same hyper-parameters, experiment settings and evaluation metrics as those in (Ahmad et al., 2019c).

[5]https://github.com/huggingface/pytorch-transformers

[6]We concatenate the word and POS representations. In our future work, we will conduct transfer learning for both POS tagging and dependency parsing.

[7]We want to cover languages from different families and with varying distances from the source language (English).

discriminator parameters, and for WGAN, we use RMSProp (Tieleman and Hinton, 2012). The learning rate is set to 0.001 and 0.00005 for Adam and RMSProp, respectively. We train the parsing models for 400 and 500 epochs with multilingual BERT and multilingual word embeddings respectively. We tune the parameter I (as shown in Algorithm 1) in the range of $[50, 100, 150]$.

Language Test. The goal of training the contextual encoder adversarially with unlabeled data from auxiliary languages is to encourage the encoder to capture more language-agnostic representations and less language-dependent features. To test whether the contextual encoders retain language information after adversarial training, we train a multi-layer Perceptron (MLP) with softmax on top of the *fixed* contextual encoders to perform a 7-way classification task.[8] If a contextual encoder performs better in the language test, it indicates that the encoder retains language specific information.

Results and Analysis

Table 6.2 presents the main transfer results of the "SelfAtt-Graph" parser when training on only English (en, baseline), English with French (en-fr), and English with Russian (en-ru). The results demonstrate that the adversarial training with the auxiliary language identification task benefits cross-lingual transfer with a small performance drop on the source language. When multi-lingual embedding is employed, the performance significantly improves, in terms of UAS of 0.48 and 0.61 over the 29 languages when French and Russian are used as the auxiliary language, respectively. When richer multilingual representation technique like mBERT is employed, adversarial training can still improve cross-lingual transfer performances (0.21 and 0.54 UAS over the 29 languages by using French and Russian, respectively). Similar to the "SelfAtt-Graph"parser, the "RNN-Stack" parser resulted in significant improvements in cross-lingual transfer from unsupervised language adaptation. We discuss our detailed experimental analysis in the following.

[8]With the source (English) and six auxiliary languages.

Lang	Multilingual Word Embeddings			Multilingual BERT		
	(en)	(en-fr)	(en-ru)	(en)	(en-fr)	(en-ru)
en	**90.23**/**88.23**	90.01/88.08	89.93/87.93	**93.19**/**91.21**	92.81/90.97	92.77/90.86
no	80.82/72.94	80.60/72.83	**80.98**/**73.10**	**85.81**/**79.03**	85.50/78.64	85.43/78.76
sv	80.33/72.54	79.90/72.16	**80.43**/**72.68**	85.61/78.34	**85.64**/**78.58**	85.44/78.33
fr	77.71/72.35	**78.49**†/**73.30**†	78.31/73.29	85.22/80.78	84.76/80.26	**85.91**†/**81.63**†
pt	76.41/67.35	76.88†/67.74	**77.09**†/**67.81**	82.93/73.33	82.71/73.13	**83.43**†/**73.88**†
da	**76.58**/**68.11**	75.99/67.64	76.25/68.03	82.36/73.53	**82.40**/73.68	82.36/**73.86**†
es	73.76/65.46	**74.14**/65.78	74.08/**65.84**	80.81/72.66	81.11/72.80	**81.38**†/**73.29**†
it	80.89/75.61	**81.33**†/**76.14**†	80.70/75.57	87.07/82.38	86.90/82.22	**87.41**/**82.67**
hr	62.21/52.67	**63.38**†/**53.83**†	63.11†/53.62†	72.96/62.65	73.39†/62.20	**74.20**†/**63.55**†
ca	73.18/64.53	**73.46**†/64.71	73.40/**64.90**†	80.40/71.42	80.30/71.42	**80.75**/**71.78**
pl	74.65/62.72	75.65†/63.31†	**75.93**/**63.60**	81.51/69.25	82.33†/69.91†	**82.48**†/**70.54**†
uk	59.25/51.92	60.58†/**52.72**†	**60.81**†/52.66†	69.98/61.52	70.24/61.61	**71.21**†/**62.84**†
sl	67.51/56.42	68.14/56.52	**68.40**/**56.87**	75.15/63.12	74.60/62.52	**75.50**/**63.65**†
nl	68.54/59.99	68.80/60.23	**69.23**†/**60.51**†	76.76/68.35	**76.94**/68.28	76.89/**68.76**†
bg	79.09/67.61	**80.01**†/**68.42**	79.72/68.39	86.82/75.47	87.08/75.40	**87.61**†/**75.94**†
ru	60.91/52.03	61.42†/52.27†	**61.67**†/**52.41**†	71.92/62.09	72.31/62.15	**72.88**†/**62.94**†
de	**71.41**/**61.97**	70.70/61.41	71.05/61.84	78.66/69.81	78.04/69.23	**79.08**†/**70.26**†
he	55.70/48.08	**57.33**†/**49.37**†	57.15†/49.36†	64.46/**55.82**	64.97†/55.63	**65.30**†/55.76
cs	63.30/54.14	63.94†/54.63†	**64.37**†/**55.08**†	73.78/63.52	**74.57**†/63.86	74.56†/**64.17**†
ro	65.13/53.98	**65.86**/**54.76**	65.57/54.42	75.10/62.99	75.85†/**63.92**†	**76.06**†/63.78†
sk	66.79/58.23	**67.46**†/**58.77**	67.42†/58.70	76.30/67.38	77.08†/67.57	**77.86**†/**68.28**†
id	49.85/44.09	**52.05**†/**45.76**†	51.57/45.31	56.80/50.24	**57.45**†/50.27	57.30†/**50.70**†
lv	70.45/49.47	70.03/49.38	**70.67**†/**49.61**†	**75.63**/53.93	75.27/53.78	75.62/**54.29**
fi	66.11/48.73	65.84/48.61	**66.28**/**48.82**	71.59/**53.81**	71.35/53.63	**71.74**/53.79
et	65.01/44.78	65.31†/45.12†	**65.38**†/**45.32**†	71.55/50.98	**71.73**/**51.27**	71.25/51.16
ar	37.63/27.48	38.72†/**28.00**†	**38.98**†/27.89†	49.27/37.62	50.37†/39.37†	**50.95**†/**39.57**†
la	47.74/34.90	48.80†/35.64†	**49.17**†/**35.73**†	51.83/38.20	51.48/38.00	**52.20**/**38.28**
ko	**34.44**/**16.18**	33.98/15.93	34.23/16.08	38.10/20.62	38.03/20.59	**38.98**†/**21.54**†
hi	36.34/27.43	36.72/27.40	**37.37**†/**28.01**†	45.40/35.03	**47.74**†/**35.90**†	46.10†/34.74
Average	65.92/55.86	66.40†/56.22†	**66.53**†/**56.32**†	73.34/62.93	73.55/62.99	**73.88**†/**63.43**†

Table 6.2: Cross-lingual transfer performances (UAS%/LAS%, excluding punctuation) of the SelfAtt-Graph parser Ahmad et al. (2019c) on the test sets. In column 1, languages are sorted by the word-ordering distance to English. (en-fr) and (en-ru) denotes the source-auxiliary language pairs. '†' indicates that the adversarially trained model results are statistically significantly better (by permutation test, $p < 0.05$) than the model trained only on the source language (en). Results show that the utilization of unlabeled auxiliary language corpora improves cross-lingual transfer performance significantly.

Lang (Src. + Aux.)	Auxiliary Language Perf.		Average Cross-lingual Perf.		Lang. Test Perf.	
	AT	MTL	AT	MTL	AT	MTL
en + fr	**78.49/73.30**†	78.26/72.98†	**66.40/56.22**	66.18/56.04	**62.25**	59.94
en + pt	**76.53/67.45**†	75.88/66.75	**66.40/56.22**	66.27/56.08	60.17	**72.02**
en + es	73.66/65.48	**74.04/65.83**†	**66.38/56.24**	66.22/56.12	56.78	**74.52**
en + ru	**61.67/52.41**†	61.08/52.04	**66.53/56.32**	66.35/56.20	37.34	**60.56**
en + de	**71.65/62.11**†	71.17/61.88	**66.41/56.13**	66.18/56.12	61.22	**72.08**
en + la	**49.22/35.94**†	48.04/35.09†	**66.45/56.20**	66.17/56.05	50.04	**64.91**

Table 6.3: Comparison between adversarial training (AT) and multi-task learning (MTL) of the contextual encoders. Columns 2–5 demonstrate the parsing performances (UAS%/LAS%, excluding punctuation) on the auxiliary languages and average of the 29 languages. Columns 6–7 present accuracy (%) of the language label prediction test. '†' indicates that the performance is higher than the baseline performance (shown in the 2nd column of Table 6.2).

Impact of Adversarial Training To understand the impact of different adversarial training types and objectives, we apply adversarial training on both word- and sentence-level with gradient reversal (GR), GAN, and WGAN objectives. Among the adversarial training objectives, we observe that in most cases, the GAN objective results in better performances than the GR and WGAN objectives. Our finding is in contrast to (Adel et al., 2018) where GR was reported to be the better objective. To further investigate, we perform the language test on the encoders trained via these two objectives. We find that the GR-based trained encoders perform consistently better than the GAN based ones on the language identification task, showing that via GAN-based training, the encoders become more language-agnostic. In a comparison between GAN and WGAN, we notice that GAN-based training consistently performs better.

Comparing word- and sentence-level adversarial training, we observe that predicting language identity at the word-level is slightly more useful for the "SelfAtt-Graph" model, while the sentence-level adversarial training results in better performances for the "RNN-Stack" model. There is no clear dominant strategy. In addition, we study the effect of using a linear classifier or a multi-layer Perceptron (MLP) as the discriminator and find that the interaction between the encoder and the linear classifier resulted in improvements.[9]

[9]This is a known issue in GAN training as the discriminator becomes too strong, it fails to provide useful signals to the generator. In our case, MLP as the discriminator predicts the language labels with higher accuracy and thus fails.

Adversarial v.s. Multi-task Training An alternative way to leverage auxiliary language corpora is by encoding language-specific information in the representation via multi-task learning. In the multi-task learning (MTL) setup, the model observes the same amount of data (both labeled and unlabeled) as the adversarially trained (AT) model. The only difference between the MTL and AT models is that in the MTL models, the contextual encoders are encouraged to capture language-dependent features while in the AT models, they are trained to encode language-agnostic features.

The experiment results using multi-task learning in comparison with the adversarial training are presented in Table 6.3. Interestingly, although the MTL objective sounds contradiction to adversarial learning, it has a positive effect on the cross-lingual parsing, as the representations are learned with certain additional information from new (unlabeled) data. Using MTL, we sometimes observe improvements over the baseline parser, as indicated with the † sign, while the AT models consistently perform better than both the baseline and the MTL model (as shown in Columns 2–5 in Table 6.3). The comparisons on parsing performances do not reveal whether the contextual encoders learn to encode language-agnostic or dependent features.

Therefore, we perform language test with the MTL and AT (GAN based) encoders, and the results are shown in Table 6.3, Columns 6–7. The results indicate that the MTL encoders consistently perform better than the AT encoders, which verifies our hypothesis that adversarial training motivates the contextual encoders to encode language-agnostic features.

6.2.4 Related Work

Adversarial Training. The concept of adversarial training via Generative Adversarial Networks (GANs) (Goodfellow et al., 2014; Szegedy et al., 2014; Goodfellow et al., 2015) was initially introduced in computer vision for image classification and received enormous success in improving model's robustness on input images with perturbations. Later many variants of GANs (Arjovsky et al., 2017; Gulrajani et al., 2017) were proposed to improve

its' training stability. In NLP, adversarial training was first utilized for domain adaptation (Ganin et al., 2016). Since then adversarial training has started to receive an increasing interest in the NLP community and applied to many NLP applications including part-of-speech (POS) tagging (Gui et al., 2017; Yasunaga et al., 2018), dependency parsing (Sato et al., 2017), relation extraction (Wu et al., 2017b), text classification (Miyato et al., 2017; Liu et al., 2017; Chen and Cardie, 2018), dialogue generation (Li et al., 2017).

In the context of cross-lingual NLP tasks, many recent works adopted adversarial training, such as in sequence tagging (Adel et al., 2018), text classification (Xu and Yang, 2017; Chen et al., 2018), word embedding induction (Zhang et al., 2017; Lample et al., 2018), relation classification (Zou et al., 2018b), opinion mining (Wang and Pan, 2018), and question-question similarity reranking (Joty et al., 2017). However, existing approaches only consider using the *target* language as the auxiliary language. It is unclear whether the language invariant representations learned by previously proposed methods can perform well on a wide variety of *unseen* languages. To the best of our knowledge, we are the first to study the effects of language-agnostic representations on a broad spectrum of languages.

Unsupervised Cross-lingual Parsing. Unsupervised cross-lingual transfer for dependency parsing has been studied over the past few years (Agić et al., 2014; Ma and Xia, 2014b; Xiao and Guo, 2014; Tiedemann, 2015; Guo et al., 2015; Aufrant et al., 2015; Rasooli and Collins, 2015; Duong et al., 2015; Schlichtkrull and Søgaard, 2017; Ahmad et al., 2019c; Rasooli and Collins, 2019b; He et al., 2019). Here, "unsupervised transfer" refers to the setting where a parsing model trained only on the source language is directly transferred to the target languages. In this work, we relax the setting by allowing unlabeled data from one or more auxiliary (helper) languages other than the source language. This setting has been explored in a few prior works. (Cohen et al., 2011) learn a generative target language parser with unannotated target data as a linear interpolation of the source language parsers. (Täckström et al., 2013) adopt unlabeled target language data and a learning method that can incorporate diverse knowledge

sources through ambiguous labeling for transfer parsing. In comparison, we leverage unlabeled auxiliary language data to learn language-agnostic contextual representations to improve cross-lingual transfer.

Multilingual Representation Learning. The basic of the unsupervised cross-lingual parsing is that we can align the representations of different languages into the same space, at least at the word level. The recent development of bilingual or multilingual word embeddings provide us with such shared representations. We refer the readers to the surveys of (Ruder et al., 2017) and (Glavaš et al., 2019) for details. The main idea is that we can train a model on top of the source language embeddings which are aligned to the same space as the target language embeddings and thus all the model parameters can be directly shared across languages. During transfer to a target language, we simply replace the source language embeddings with the target language embeddings. This idea is further extended to learn multilingual contextualized word representations, for example, multilingual BERT (Devlin et al., 2018), have been shown very effective for many cross-lingual transfer tasks (Wu and Dredze, 2019b). In this work, we show that further improvements can be achieved by adaptating the contextual encoders via unlabeled auxiliary languages even when the encoders are trained on top of multilingual BERT.

6.3 Representation Learning for Programming Languages

Engineers and developers write software programs in a programming language (PL) like Java, Python, etc., and often use natural language (NL) to communicate with each other. Use of NL in software engineering ranges from writing documentation, commit messages, bug reports to seeking help in different forums (*e.g.,* Stack Overflow), etc. Automating different software engineering applications, such as source code summarization, generation, and translation, heavily rely on the understanding of PL and NL—we collectively refer them as PLUG (stands for, Program and Language Understanding and Generation)

(a) Program snippet in Python

```python
def sort_list(uns):
    return sorted(uns, key=lambda x:x[0])
```

(b) Program snippet in Java

```java
static Tuple[] sortArray(Tuple[] uns){
    return Arrays.sort(
      uns, new Comparator<Tuple>() {
       public int compare(
        Tuple o1, Tuple o2) {
           return o1.get(0) == o2.get(0);
        }
    });
}
```

Summary: sort a list of tuples by first element

Figure 6.2: Example motivating the need to understand the association of program and natural languages for code summarization, generation, and translation.

applications or tasks. Note that the use of NL in software development is quite different than colloquially written and spoken language. For example, NL in software development often contains domain-specific jargon, *e.g.,* when software engineers use *Code Smell*[10], it means a potential problem in code (something other than *Smell* in regular English language).

Our goal is to develop a general-purpose model that can be used in various PLUG applications. Recent advancements in deep learning and the availability of large-scale PL and developers' NL data ushered in the automation of PLUG applications. One important aspect of PLUG applications is that they demand a profound understanding of program syntax and semantics and mutual dependencies between PL and NL. For example, Figure 6.2 shows two implementations of the same algorithm (sorting) in two PL and corresponding NL summary. An automatic translation tool must understand that function *sorted* in Python acts similar to *Arrays.sort* in Java and the *lambda* operation in Python is equivalent to instantiating a *Comparator* object in Java. Similarly, a tool that summarizes either of these code must understand that *x[0]* in Python or *Tuple.get(0)* in

[10]https://en.wikipedia.org/wiki/Code_smell

Java refers to the first element in the tuple list.

Most of the available data in PL and NL are unlabeled and cannot be trivially used to acquire PLUG task-specific supervision. However, PLUG tasks have a common prerequisite — understanding PL and NL syntax and semantics. Leveraging unlabelled data to pretrain a model to learn PL and NL representation can be transferred across PLUG tasks. This approach reduces the requirement of having large-scale annotations for task-specific fine-tuning. In recent years we have seen a colossal effort to pretrain models on a massive amount of unlabeled data (*e.g.,* text, images, videos) Devlin et al. (2019); Liu et al. (2019c); Conneau and Lample (2019); Conneau et al. (2020); Li et al. (2019); Sun et al. (2019b) to transfer representation encoders across a wide variety of applications. There are a few research effort in learning general purpose PL-NL representation encoders, such as CodeBERT Feng et al. (2020) and GraphCodeBERT Guo et al. (2021) that are pretrained on a *small-scale* bimodal data (code-text pairs). Such models have been found effective for PLUG tasks, including code search, code completion, etc.

Language generation tasks such as code summarization is modeled as sequence-to-sequence learning, where an encoder learns to encode the input code and a decoder generates the target summary. Despite the effectiveness of existing methods, they do not have a pretrained decoder for language generation. Therefore, they still require a large amount of parallel data to train the decoder. To overcome this limitation, Lewis et al. (2020a) proposed denoising sequence-to-sequence pre-training where a Transformer Vaswani et al. (2017) learns to reconstruct an original text that is corrupted using an arbitrary noise function. Very recently, Lachaux et al. (2020) studied denoising pre-training using a large-scale source code collection aiming at unsupervised program translation and found the approach useful.

This raises a natural question, *can we unify pre-training for programming and natural language?* Presumably, to facilitate such pre-training, we need unlabeled NL text that is relevant to software development. Note that unlike other bimodal scenarios (*e.g.,* vision and language), PL and associated NL text share the same alphabet or uses anchor tokens (*e.g.,* "sort", "list", "tuple" as shown in Figure 6.2) that can help to learn alignment between

	Java	Python	NL
All Size	352 GB	224 GB	79 GB
All - Nb of tokens	36.4 B	28 B	6.7 B
All - Nb of documents	470 M	210 M	47 M

Table 6.4: Statistics of the data used to pre-train PLBART. "Nb of documents" refers to the number of functions in Java and Python collected from Github and the number of posts (questions and answers) in the natural language (English) from StackOverflow.

semantic spaces across languages.

We introduce PLBART (Program and Language BART), a bidirectional and autoregressive transformer pre-trained on unlabeled data across PL and NL to learn multilingual representations applicable to a broad spectrum of PLUG applications. We evaluate PLBART on code summarization, generation, translation, program repair, clone detection, and vulnerability detection tasks. Experiment results show that PLBART outperforms or rivals state-of-the-art methods, *e.g.,* CodeBERT and GraphCodeBERT, demonstrating its promise on program understanding and generation. We perform a thorough analysis to demonstrate that PLBART learns program syntax, logical data flow that is indispensable to program semantics, and excels even when limited annotations are available. We release our code[11] to foster future research.

6.3.1 Denoising Pre-training

PLBART uses denoising sequence-to-sequence pre-training to utilize unlabeled data in PL and NL. Such pre-training lets PLBART reason about language syntax and semantics. At the same time, PLBART learns to generate language coherently.

6.3.1.1 Pre-training PLBART

Data & pre-processing We pre-train PLBART on a large-collection of Java and Python functions and natural language descriptions from Github and StackOverflow, respectively. We download all the GitHub repositories associated with Java and Python

[11]https://github.com/wasiahmad/PLBART

PLBART Encoder Input	PLBART Decoder Output
Is 0 the **[MASK]** Fibonacci **[MASK]** ? <En>	<En> Is 0 the **first** Fibonacci **number** ?
public static main (String args [ ]) { date = Date ( ) ; System . out . (String . format (" Current Date : % tc " ,)) ; } <java>	<java> public static **void** main (String args [ ]) { **Date** date = **new** Date ( ) ; System . out . **printf** (String . format (" Current Date : % tc " , **date**)) ; }
def addThreeNumbers (x , y , z) : NEW_LINE INDENT return **[MASK]** <python>	<python> def addThreeNumbers (x , y , z) : NEW_LINE INDENT return **x** + **y** + **z**

Table 6.5: Example encoder inputs and decoder outputs during denoising pre-training of PLBART. We use three noising strategies: token masking, token deletion, and token infilling (shown in the three examples, respectively).

languages available on Google BigQuery.[12] We extract the Java and Python functions following the pre-processing pipeline from Lachaux et al. (2020). We collect the StackOverflow posts (include both questions and answers, exclude code snippets) by downloading the data dump (date: 7th September 2020) from stackexchange.[13] Statistics of the pre-training dataset are presented in Table 6.4. We tokenize all the data with a sentencepiece model (Kudo and Richardson, 2018) learned on 1/5'th of the pre-training data. We train sentencepiece to learn 50,000 subword tokens.

One key challenge to aggregate data from different modalities is that some modalities may have more data, such as we have 14 times more data in PL than NL. Therefore, we mix and up/down sample the data following Conneau and Lample (2019) to alleviate the bias towards PL. We sample instances for pre-training according to a multinomial distribution with probabilities $(q_1, q_2, \ldots, q_N)$:

$$q_i = \frac{1}{p_i} \cdot \frac{p_i^\alpha}{\sum_{j=1}^{N} p_j^\alpha}, p_i = \frac{n_i}{\sum_{j=1}^{N} n_j},$$

where N is the total number of languages and n_i is the total number of instances in language i. We set the smoothing parameter α to 0.3.

[12] https://console.cloud.google.com/ marketplace/details/github/github-repos

[13] https://archive.org/download/stackexchange

	PLBART Encoder Input	PLBART Decoder Input
S	def maximum (a , b , c) : NEW_LINE INDENT return max ([a , b , c]) <python>	<En> Find the maximum of three numbers
G	Find the maximum of three numbers <En>	<java> public int maximum (int a , int b , int c) { return Math . max (a , Math . max (b , c)) }
T	public int maximum (int a , int b , int c) { return Math . max (a , Math . max (b , c)) } <java>	<python> def maximum (a , b , c) : NEW_LINE INDENT return max ([a , b , c])

Table 6.6: Example inputs to the encoder and decoder for fine-tuning PLBART on sequence generation tasks: source code summarization (S), generation (G), and translation (T).

Architecture PLBART uses the same architecture as BART_{base} (Lewis et al., 2020a), it uses the sequence-to-sequence Transformer architecture (Vaswani et al., 2017), with 6 layers of encoder and 6 layers of decoder with model dimension of 768 and 12 heads (∼140M parameters). The only exception is, we include an additional layer-normalization layer on top of both the encoder and decoder following Liu et al. (2020), which is found to stabilize training with FP16 precision.

Noise function, f In denoising autoencoding, a model learns to reconstruct an input text that is corrupted by a noise function. Reconstruction of the original input requires the model to learn language syntax and semantics. In this work, we use three noising strategies: token masking, token deletion, and token infilling (Lewis et al., 2020a). According to the first two strategies, random tokens are sampled and replaced with a mask token or deleted from the input sequence. In token infilling, a number of text spans are sampled and replaced with a *single* mask token. The span lengths are drawn from a Poisson distribution ($\lambda = 3.5$). We mask 35% of the tokens in each instance.

Input/Output Format The input to the encoder is a noisy text sequence, while the input to the decoder is the original text with one position offset. A language id symbol (e.g., <java>, <python>) is appended and prepended to the encoder and decoder inputs, respectively. We provide a few examples in Table 6.5. The input instances are truncated if they exceed a maximum sequence length of 512.

Learning PLBART is pre-trained on N languages (in our case, N=3), where each language N_i has a collection of unlabeled instances $\mathcal{D}_i = \{x_1, \ldots, x_{n_i}\}$. Each instance is corrupted using the noise function f and we train PLBART to predict the original instance x from $f(x)$. Formally, PLBART is trained to maximize $\mathcal{L}_\theta$:

$$\mathcal{L}_\theta = \sum_{i=1}^{N} \sum_{j=1}^{m_i} \log P(x_j | f(x_j); \theta)$$

where m_i is the number of sampled instances in language i and the likelihood P is estimated following the standard sequence-to-sequence decoding.

Optimization We train PLBART on 8 Nvidia GeForce RTX 2080 Ti GPUs for 100K steps. The effective batch size is maintained at 2048 instances. We use Adam (ϵ = 1e-6, β_2 = 0.98) with a linear learning rate decay schedule for optimization. We started the training with dropout 0.1 and reduced it to 0.05 at 50K steps and 0 at 80K steps. This is done to help the model better fit the data (Liu et al., 2020). The total training time was approximately 276 hours (11.5 days). All experiments are done using the Fairseq library (Ott et al., 2019).

6.3.1.2 Fine-tuning PLBART

We fine-tune PLBART for two broad categories of downstream applications.

Sequence Generation PLBART has an encoder-decoder architecture where the decoder is capable of generating target sequences autoregressively. Therefore, we can directly fine-tune PLBART on sequence generation tasks, such as code summarization, generation, and translation. Unlike denoising pre-training, the source sequence is given as input to the encoder during fine-tuning, and the decoder generates the target sequence. The source and target sequence can be a piece of code or text sequence. Table 6.6 shows a few examples of input and output to and for PLBART for different generation tasks. Note that PLBART prepends a language id to the decoded sequence; it enables fine-tuning

Task	Dataset	Language	Train	Valid	Test
Summarizaion	Husain et al. (2019)	Ruby	24,927	1,400	1,261
		Javascript	58,025	3,885	3,291
		Go	167,288	7,325	8,122
		Python	251,820	13,914	14,918
		Java	164,923	5,183	10,955
		PHP	241,241	12,982	14,014
Generation	Iyer et al. (2018)	NL to Java	100,000	2,000	2,000
Translation	Code-Code (Lu et al., 2021)	Java to C#	10,300	500	1,000
		C# to Java	10,300	500	1,000
	Program Repair (Tufano et al., 2019)	Java_{small}	46,680	5,835	5,835
		Java_{medium}	52,364	6,545	6,545
Classification	Vulnerability Detection (Zhou et al., 2019)	C/C++	21,854	2,732	2,732
	Clone Detection (Wang et al., 2020)	Java	100,000	10,000	415,416

Table 6.7: Statistics of the downstream benchmark datasets.

PLBART in a multilingual setting (*e.g.,* code generation in multiple languages).[14]

Sequence Classification We fine-tune PLBART on sequence classification tasks following Lewis et al. (2020a). The input sequence is fed into both the encoder and decoder. For a pair of inputs, we concatenate them but insert a special token ("</s>") between them. A special token is added at the end of the input sequence. This last token's representation from the final decoder layer is fed into a linear classifier for prediction.

Optimization We fine-tune PLBART for a maximum of 100K steps on all the downstream tasks with 2500 warm-up steps. We set the maximum learning rate, effective batch size, and dropout rate to 3e-5, 32 and 0.1, respectively. The final models are selected based on the validation BLEU (in generation task) or accuracy (in classification tasks). Fine-tuning PLBART is carried out in one Nvidia GeForce RTX 2080 Ti GPU.

6.3.2 Experiments Setup

To understand PLBART's performance in a broader context, we evaluate PLBART on several tasks. Our evaluation focuses on assessing PLBART's ability to capture rich semantics in source code and associated natural language text.

6.3.2.1 Evaluation Tasks

We divide the evaluation tasks into four categories. The evaluation task datasets are summarized in Table 6.7. We use CodeXGLUE (Lu et al., 2021) provided public dataset and corresponding train-validation-test splits for all the tasks.

Code Summarization refers to the task of generating a natural language (English) summary from a piece of code. We fine-tune PLBART on summarizing source code written in six different programming languages, namely, Ruby, Javascript, Go, Python, Java, and PHP.

Code Generation is exactly the opposite of code summarization. It refers to the task of generating a code (in a target PL) from its NL description. We fine-tune PLBART on the Concode dataset (Iyer et al., 2018), where the input is a text describing class member functions in Java and class environment, the output is the target function.

Code Translation requires a model to generate an equivalent code in the target PL from the input code written in the source PL. Note that the source and target PL can be the same. Hence, we consider two types of tasks in this category.

The first task is a typical PL translation task, translating a code *i.e.,* from Java code to C#, and vice versa. In this task, the semantic meaning of the translated code should exactly match the input code. Thus, this task evaluates PLBART's understanding of program semantics and syntax across PL. The second task we consider is program repair.

[14]We do not perform multilingual fine-tuning in this work.

In this task, the input is a buggy code, and the output is a modified version of the same code which fixes the bug. This task helps us understand PLBART's ability to understand code semantics and apply semantic changes in the code.

Code Classification aims at predicting the target label given a single or a pair of source code. We evaluate PLBART on two classification tasks. The first task is clone detection, where given a pair of code, the goal is to determine whether they are clone of each other (similar to paraphrasing in NLP). The second task is detecting whether a piece of code is vulnerable. This task help us gauging PLBART's effectiveness in program understanding in an unseen PL since the code examples in this task are written in C/C++.

6.3.2.2 Evaluation Metrics

BLEU computes the n-gram overlap between a generated sequence and a collection of references. We use corpus level BLEU (Papineni et al., 2002) score for all the generation tasks, except code summarization where we use smoothed BLEU-4 score (Lin and Och, 2004) following Feng et al. (2020).

CodeBLEU is a metric for measuring the quality of the synthesized code (Ren et al., 2020). Unlike BLEU, CodeBLEU also considers grammatical and logical correctness based on the abstract syntax tree and the data-flow structure.

Exact Match (EM) evaluates if a generated sequence exactly matches the reference.

6.3.2.3 Baseline Methods

We compare PLBART with several state-of-the-art models and broadly divide them into two categories. First, the models that are trained on the evaluation tasks from scratch, and second, the models that are pre-trained on unlabeled corpora and then fine-tuned on the evaluation tasks.

Methods	Ruby	Javascript	Go	Python	Java	PHP	Overall
Seq2Seq	9.64	10.21	13.98	15.93	15.09	21.08	14.32
Transformer	11.18	11.59	16.38	15.81	16.26	22.12	15.56
RoBERTa	11.17	11.90	17.72	18.14	16.47	24.02	16.57
CodeBERT	12.16	14.90	18.07	19.06	17.65	**25.16**	17.83
PLBART	**14.11**	**15.56**	**18.91**	**19.30**	**18.45**	23.58	**18.32**

Table 6.8: Results on source code summarization, evaluated with smoothed BLEU-4 score. The baseline results are reported from Feng et al. (2020).

Training from Scratch

Seq2Seq (Luong et al., 2015b) is an LSTM based Seq2Seq model with attention mechanism. Vocabulary is constructed using byte-pair encoding.

Transformer (Vaswani et al., 2017) is the base architecture of PLBART and other pre-trained models. Transformer baseline has the same number of parameters as PLBART. Hence, a comparison with this baseline demonstrates the direct usefulness of pre-training PLBART.

Pre-trained Models

PLBART consists of an encoder and autoregressive decoder. We compare PLBART on two categories of pre-trained models. First, the encoder-only models (*e.g.,* RoBERTa, CodeBERT, and GraphCodeBERT) that are combined with a randomly initialized decoder for task-specific fine-tuning. The second category of baselines include decoder-only models (CodeGPT) that can perform generation autoregressively.

RoBERTa, RoBERTa (code) are RoBERTa (Liu et al., 2019c) model variants. While RoBERTa is pre-trained on natural language, RoBERTa (code) is pre-trained on source code from CodeSearchNet (Husain et al., 2019).

CodeBERT (Feng et al., 2020) combines masked language modeling (MLM) (Devlin et al., 2019) with replaced token detection objective (Clark et al., 2020) to pretrain a Transformer encoder.

Methods	EM	BLEU	CodeBLEU
Seq2Seq	3.05	21.31	26.39
Guo et al. (2019)	10.05	24.40	29.46
Iyer et al. (2019)	12.20	26.60	-
GPT-2	17.35	25.37	29.69
CodeGPT-2	18.25	28.69	32.71
CodeGPT-adapted	**20.10**	32.79	35.98
PLBART	18.75	**36.69**	**38.52**
PLBART_{10K}	17.25	31.40	33.32
PLBART_{20K}	18.45	34.00	35.75
PLBART_{50K}	17.70	35.02	37.11

Table 6.9: Results on text-to-code generation task using the CONCODE dataset (Iyer et al., 2018).

GraphCodeBERT (Guo et al., 2021) is a concurrent work with this research which improved CodeBERT by modeling the data flow edges between code tokens. We report GraphCodeBERT's performance directly from the paper since their implementation is not publicly available yet.

GPT-2, CodeGPT-2, and CodeGPT-adapted are GPT-style models. While GPT-2 (Radford et al., 2019) is pretrained on NL corpora, CodeGPT-2 and CodeGPT-adapted are pretrained on CodeSearchNet (Lu et al., 2021). Note that, CodeGPT-adapted starts from the GPT-2 checkpoint for pre-training.

6.3.3 Results & Analysis

We aim to address the following questions.

1. Does PLBART learn strong program and language representations from unlabeled data?
2. Does PLBART learn program characteristics, *e.g.,* syntax, style, and logical data flow?
3. How does PLBART perform in an unseen language with limited annotations?

Input text: returns the count to which the specified key is mapped in this frequency counter , or 0 if the map contains no mapping for this key .

(a) Reference Code

```
Integer function (T arg0) {
    Integer loc0 = counter.get(arg0);
    if (loc0 == null) {
        return 0 ;
    }
    return loc0;
}
```

(b) Generated Code

```
int function (T arg0) {
    Integer loc0 = counter.get(arg0);
    if (loc0 == null) {
        return 0 ;
    }
    else {
        return loc0;
    }
}
```

Figure 6.3: An example of generated code by PLBART that is syntactically and semantically valid, but does not match the reference.

6.3.3.1 Code Summarization

Table 6.8 shows the result of code summarization. PLBART outperforms the baseline methods in five out of the six programming languages with an overall average improvement of 0.49 BLEU-4 over CodeBERT. The highest improvement (∼16%) is in the Ruby language, which has the smallest amount of training examples. Unlike CodeBERT, PLBART is not pretrained on the Ruby language; however, the significant performance improvement indicates that PLBART learns better generic program semantics. In contrast, PLBART performs poorly in the PHP language. The potential reason is syntax mismatch between the pre-trained languages and PHP. Surprisingly, RoBERTa performs better than PLBART on the PHP language. We suspect that since RoBERTa is pre-trained on natural language only, it does not suffer from the syntax mismatch issue. Overall in comparison to the Transformer baseline, PLBART improves with an average of 2.76 BLEU-4, and we credit this improvement to the pre-training step.

Methods	Java to C#			C# to Java		
	BLEU	EM	CodeBLEU	BLEU	EM	CodeBLEU
Naive Copy	18.54	0	34.20	18.69	0	43.04
PBSMT	43.53	12.50	42.71	40.06	16.10	43.48
Transformer	55.84	33.00	63.74	50.47	37.90	61.59
RoBERTa (code)	77.46	56.10	83.07	71.99	57.90	80.18
CodeBERT	79.92	59.00	85.10	72.14	58.80	79.41
GraphCodeBERT	80.58	59.40	-	72.64	58.80	-
PLBART	**83.02**	**64.60**	**87.92**	**78.35**	**65.00**	**85.27**

Table 6.10: Results on source code translation using Java and C# language dataset introduced in (Lu et al., 2021). PBSMT refers to phrase-based statistical machine translation where the default settings of Moses decoder (Koehn et al., 2007) is used. The training data is tokenized using the RoBERTa (Liu et al., 2019c) tokenizer.

6.3.3.2 Code Generation

Table 6.9 shows the evaluation result on code generation from NL description. PLBART outperforms all the baselines in terms of BLEU and CodeBLEU. While CodeGPT-adapted Lu et al. (2021) achieves the best Exact Match (EM) score, PLBART outperforms CodeGPT-adapted by a large margin in terms of CodeBLEU. This result implies that PLBART generates *significantly more* syntactically and logically correct code than all the baselines.

Figure 6.3 shows an example of code generated by PLBART. The difference between the reference code and the generated code is in line 6 onward. In the reference code, `loc0` is returned, however same `loc0` is returned in an `else` block in the generated code. If we look closely, in the reference code, line 6 will be executed only if the condition in line 3 (*i.e.,* `loc0 == null`) is `false`. In the generated code, `loc0` will be returned only if the condition in line 3 is `false`, making the generated code semantically equivalent to the reference code.

To study whether PLBART learns code syntax and logical flow during pre-training or fine-tuning, we perform an ablation study where we use subset of the training examples (10K, 20K, and 50K) to fintune PLBART in this task. As table 6.9 shows, with only 10K examples, PLBART outperforms all baselines in terms of CodeBLUE. This ablation

shows that PLBART learns program syntax and data flow during pre-training, resulting in effective performance on downstream tasks even when finetuned on small number of examples.

As shown in prior works Yin and Neubig (2017); Chakraborty et al. (2020), generating syntactically and logically correct code has been a big challenge in program generation. We conjecture that PLBART's large-scale denoising sequence-to-sequence pre-training helps understand program syntax and logical flow; therefore enables PLBART to generate syntactically and logically valid code.

6.3.3.3 Code Translation

Table 6.10 presents the evaluation results on code translation. PLBART outperforms all the baselines *w.r.t.* EM, BLEU, and CodeBLEU. PLBART improves over CodeBERT by 9.5% and 10.5% when translating from Java to C# and C# to Java, respectively. Although PLBART is not pretrained on C# language, there is a significant syntactic and semantic similarity between Java and C#. Thus PLBART understands C# language syntax and semantics. However, such similarities are non-trivial, making the Naive copy and PBSMT perform very poorly in both the translation tasks.

Figure 6.4 shows an example where PLBART's generated C# code does not exactly match the reference; however, they are semantically equivalent. In the reference, the `else` block (line 4-9) is equivalent to the `else if` block (line 4-7) in the generated code. In addition, `start` is generated as function parameter and used in the function body, equivalent to `start_1` in the reference code. This further corroborates the syntactic understanding of PLBART and its ability to reason about the data flow in source code.

In the program repair task, both the input and the output are in the same language. While the input is a buggy code, the output should be the target bug-free code. Thus in this task, the exact match is the critical metric. Nevertheless, as shown in table 6.11, PLBART can generate 17.13%, and 74.03% more correct bug fixes than CodeBERT in Java_{small} and Java_{medium} datasets, respectively. On the other hand, PLBART performs

(a) Reference Code : C#

```
public bool find(int start_1){
    findPos = start_1;
    ...
    else{
        if (findPos >= _regionEnd){
            matchFound = false;
            return false;
        }
    }
    ...
}
```

(b) Generated Code : C#

```
public bool find(int start){
    findPos = start;
    ...
    else if (findPos >= _regionEnd){
        matchFound = false;
        return false;
    }
    ...
}
```

Figure 6.4: Example C# code generated by PLBART that does not exactly match the reference code.

comparably to GraphCodeBERT that uses structure-aware pre-training to learn program syntax and semantics.

6.3.3.4 Classification

In both clone detection and the vulnerability detection tasks, PLBART outperforms CodeBERT. We present the results in Table 6.12. In the vulnerability detection task, code semantics is the most critical feature Zhou et al. (2019); Chakraborty et al. (2020). Since PLBART is not pretrained on C/C++ language, its improved performance compared to the Transformer baseline is the testament that PLBART can identify semantics beyond the language syntax's specifics. Moreover, PLBART's improved performances over CodeBERT and GraphCodeBERT confirms its effectiveness in program understanding in addition to its generation ability.

We acknowledge that neither PLBART nor CodeBERT is state-of-the-art in vulnerability detection, as graph-based models perform best in this task. In this evaluation, our

Methods	Small		Medium	
	EM	BLEU	EM	BLEU
Naive Copy	0	78.06	0	90.91
Seq2Seq	10.00	76.76	2.50	72.08
Transformer	14.70	77.21	3.70	89.25
CodeBERT	16.40	77.42	5.16	**91.07**
GraphCodeBERT	17.30	**80.58**	**9.10**	72.64
PLBART	**19.21**	77.02	8.98	88.50

Table 6.11: Results on program repair (in Java).

Tasks	Vulnerability Detection	Clone Detection
Transformer	61.64	-
CodeBERT	62.08	96.5
GraphCodeBERT	-	97.1
PLBART	**63.18**	**97.2**

Table 6.12: Results on the vulnerable code detection (accuracy) and clone detection (F1 score) tasks.

goal is to study how well PLBART understands program semantics in an unseen language for a different type of task (other than the generation, *i.e.,* classification).

6.3.4 Related Work

Transformer (Vaswani et al., 2017), a sequence-to-sequence architecture that includes an encoder and decoder, has shown tremendous promise in natural language processing (NLP), computer vision, software engineering, and more. Devlin et al. (2019) first proposed to pre-train a large Transformer architecture, called BERT, to learn representations of natural language using large-scale unlabeled data in a self-supervised fashion. Later, BERT's task-independent pre-training approach is rigorously studied (Devlin et al., 2019; Liu et al., 2019c; Solaiman et al., 2019; Feng et al., 2020; Sun et al., 2019b; Li et al., 2020). While BERT-like models have shown effectiveness in learning contextualized representation, it is not very useful in generation tasks. GPT (Radford et al., 2018) style models improve upon BERT for generative tasks with autoregressive pre-training; however, unlike BERT, they are not bidirectional. Lewis et al. (2020a) introduced BART,

a denoising autoencoder that uses a bidirectional encoder and an auto-regressing decoder. Similar to BART, PLBART uses denoising pre-training to cope with generative tasks and learns multilingual representations of programming and natural language jointly.

6.4 Summary

This chapter studied representation learning using unlabeled data. Specifically, we leverage unlabeled language resources for adversarial training and denoising pre-training to induce language-agnostic encoders to improve the performances of the cross-lingual transfer in downstream tasks. To make cross-lingual dependency parsing more robust and generalizable, we presented an adversarial training framework by using English as the source language and unlabeled resources from six foreign languages. Experiments and analysis not only show improvements on cross-lingual parsing, but also demonstrates that contextual encoders successfully learns not to capture language-dependent features through adversarial training. This study opens up the opportunity to investigate the effectiveness of adversarial training for multi-source transfer parsing and other cross-lingual NLP applications.

This chapter also presents PLBART, a sizeable sequence-to-sequence model pre-trained on a large collection of unlabeled programming and natural language data that can perform program and language understanding and generation tasks. PLBART achieves state-of-the-art performance on various downstream software engineering tasks, including code summarization, code generation, and code translation. Furthermore, experiments on discriminative tasks establish PLBART's effectiveness on program understanding. We also show that PLBART learns crucial program characteristics due to pre-training, such as syntax, identifier naming conventions, data flow. In the future, we want to explore ways to fine-tune PLBART on all the downstream tasks jointly.

CHAPTER 7

Conclusion and Future Work

Cross-lingual representation learning has emerged as an indispensable ingredient to avail modern NLP applications in a broad spectrum of languages. However, it is challenging to utilize such representations in the target languages since no or limited supervision is available. This dissertation discussed challenges in cross-lingual representation learning and presented several approaches to improve the robustness and generalizability of such representations to facilitate the cross-lingual transfer. Figure 7.1 summarizes the contributions made in this dissertation.

7.1 Summary of Contributions

The world is well connected nowadays, and people seek information about events taking place around the world. Therefore, a multilingual NLP system that extracts and processes news and stories in different languages can facilitate information dissemination around the globe. Chapter 1 of this dissertation motivates the need to learn multilingual representations to build NLP systems capable of processing information provided in multiple languages. We discuss the limitations of multilingual NLP via supervised learning; it requires annotated resources in all the target languages. To remedy the lack of resources in most languages of today's world, we emphasize transfer learning by utilizing resources available in popular languages like English. We also provide the reasoning behind representation learning for cross-lingual transfer.

In chapter 2, we present the brief history of vector-based representation learning for NLP. To lay the groundwork, we discuss several approaches to learning distributed and

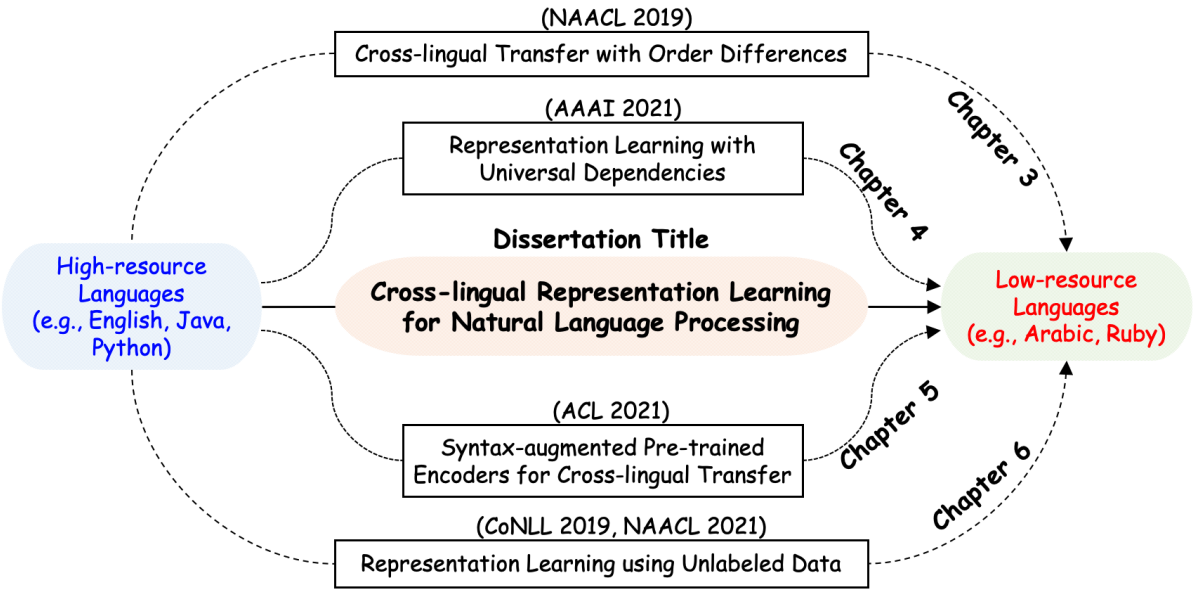


Figure 7.1: Summary of contributions made in the dissertation. Different chapters demonstrated the challenges in cross-lingual representation learning due to word order differences across languages (Chapter 3), how universal dependencies can be utilized to enhance representation learning (Chapter 4) and pre-trained multilingual encoders (Chapter 5) for cross-lingual transfer, and how such representations can be learned or improved by using unlabeled data (Chapter 6).

contextualized representations. The second half of the chapter presents cross-lingual counterparts of the representation learning approaches. Then we discuss how unlabeled monolingual resources and other available linguistic resources are utilized to facilitate cross-lingual representation learning. The chapter ended by discussing the pros and cons of training deep neural network structures to encode natural language and transfer them across languages.

In chapter 3, we discuss the challenge in modeling word order to tackle the typological differences across languages. We particularly address the question, what type of neural architectures are suitable to learn transferable representations given that the source and target languages are closer or distant from each other? We perform a thorough study on the two preeminent neural architectures, Recurrent Neural Networks (RNNs) and Self-Attention mechanism as the cross-lingual representation learning encoders. We showed that the Self-Attention mechanism that is less susceptible to word order performs

better when the source and target languages are distant to each other and vice versa. We quantify the distance between two languages based on the differences in word order. We chose dependency parsing task as the test bed since word order typology significantly influences the task. To further improve cross-lingual transfer, we propose to discard directional information while encoding word positions in sentences so that the Self-Attention mechanism can adapt to the word order variances of distant target languages.

The next chapter shows that leveraging the universal dependency structure in learning contextual representations improves cross-lingual relation and event extraction. Specifically, we present a Graph Attention Transformer Encoder (GATE) to learn contextual representations by encoding the dependency structure of the input sequence. GATE modifies the self-attention mechanism in the Transformer encoder as it uses the pairwise syntactic distances between words to weigh the attention score. Experiments show that GATE is less sensitive to language word order and thus suitable to transfer across typologically diverse languages, e.g., English to Arabic.

While prior works showed that multilingual language encoder, mBERT learns compositional features during pre-training that mimick universal dependency structure, in chapter 5, we argue that it is necessary to force mBERT to embed the dependency structure while fine-tuning on the downstream tasks in the source language. We propose a fusion technique to add syntax-bias to the self-attention mechanism. The underlying idea is to guide the self-attention mechanism to attend tokens with a specific part-of-speech tag sequence or dependencies. To augment mBERT with syntax information, an auxiliary objective is adopted when mBERT performs the downstream task during fine-tuning. The chapter ends with discussion on the limitations and the scope for future works.

In chapter 6, we advocate the use of unlabeled resources to make multilingual representations robust and transferable across languages. Since there is a scarcity of annotations for low-resource languages, we can collect corpora of unlabeled sentences. Given such corpora, one fundamental research question is how we can improve the cross-lingual transferability of the language encoders? We design an adversarial training framework to make multilingual encoders language-agnostic, resulting in effective cross-lingual transfer.

We extend our study on using unlabeled data in NLP to benefit software engineering applications by jointly pre-training language models on natural and programming languages. The language model achieved state-of-the-art performances on several software engineering tasks.

7.2 Future Work

There are several research questions in cross-lingual representation learning that demand further research; we briefly discuss a few of them.

Modeling word order for transfer learning. This dissertation showed that the self-attention mechanism outperforms recurrent neural networks in cross-lingual transfer between distant language pairs, e.g., English – Arabic, English – Hindi. Our proposal of dropping the directional information while encoding word position in sentences improved transfer performances further. Many recent works proposed different positional encoding mechanisms and showed improvements in many applications, e.g., machine translation (Cooper Stickland et al., 2021; Liu et al., 2021). However, it is still an open question about how to model word positions such that the typological differences between target and source languages minimize. It is particularly challenging in the zero-shot setting where there are no labeled resources for the target languages. In such a setting, utilization of databases of structural properties of languages could benefit modeling word positions, such as WALS[1]. It has been shown that effective modeling of word order can benefit many NLP applications; however, it is still unknown how much typological differences affect different NLP applications. Since benchmark datasets are now available in a wide range of languages for many NLP applications, it is high time to study the effect of word order modeling in cross-lingual transfer.

[1]https://wals.info/

Role of language syntax in improving alignment of multilingual contextual word representations. Pre-trained multilingual language encoders, such as multilingual BERT Devlin et al. (2019) and XLM-R Conneau et al. (2020), demonstrate noteworthy performance on zero-shot cross-lingual transfer for many downstream applications. These language encoders learn a shared contextual embedding space; represent word pairs in parallel sentences with similar contextual representations. However, they lack when the source and target languages are less similar at levels of morphology, syntax, and semantics. Recent studies Cao et al. (2020); Pan et al. (2021); Dou and Neubig (2021) have shown that aligning the representations of different languages in the multilingual embedding space plays an important role in zero-shot cross-lingual transfer learning. Most of these works use parallel data to further fine-tune the encoders to learn language alignment. Since languages have universal dependency structure, it would interesting to investigate the role of language syntax in learning cross-lingual alignment.

Representation learning across domains. The challenges in cross-lingual representation learning are not limited to tackling the differences between languages at levels of morphology, syntax, and semantics. A big challenge in natural language processing is understanding the use of language in different domains, such as social media. In social networks, often a user uses *code-mixed* language; mixed up two or more languages in the same conversation. Also, non-English users often write sentences in their native language but using the English alphabet. For example, "Se tar kajer prothom ongsho sesh koreche." (translates to "He finished the part of his work."). Different domains pose different challenges in learning representations, and cross-lingual representation learning techniques should account for those challenges too.

Adel, H., Bryl, A., Weiss, D., and Severyn, A. (2018). Adversarial neural networks for cross-lingual sequence tagging. *arXiv preprint arXiv:1808.04736*. 86, 88

Agić, Ž., Tiedemann, J., Dobrovoljc, K., Krek, S., Merkler, D., and Može, S. (2014). Cross-lingual dependency parsing of related languages with rich morphosyntactic tagsets. In *EMNLP 2014 Workshop on Language Technology for Closely Related Languages and Language Variants*. 30, 88

Ahmad, W., Chakraborty, S., Ray, B., and Chang, K.-W. (2021a). Unified pre-training for program understanding and generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2668, Online. Association for Computational Linguistics. 5, 76

Ahmad, W., Li, H., Chang, K.-W., and Mehdad, Y. (2021b). Syntax-augmented multilingual BERT for cross-lingual transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4538–4554, Online. Association for Computational Linguistics. 4

Ahmad, W., Zhang, Z., Ma, X., Hovy, E., Chang, K.-W., and Peng, N. (2019a). On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2440–2452, Minneapolis, Minnesota. Association for Computational Linguistics. 4, 34, 46, 48, 53, 58

Ahmad, W. U., Peng, N., and Chang, K.-W. (2021c). GATE: graph attention transformer encoder for cross-lingual relation and event extraction. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*. 4, 74, 75

Ahmad, W. U., Zhang, Z., Ma, X., Chang, K.-W., and Peng, N. (2019b). Cross-lingual dependency parsing with unlabeled auxiliary languages. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 372–382, Hong Kong, China. Association for Computational Linguistics. 5, 53, 76

Ahmad, W. U., Zhang, Z., Ma, Z., Hovy, E., Chang, K.-W., and Peng, N. (2019c). On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. xi, 79, 83, 85, 88

Ahn, D. (2006). The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics. 52

Alsentzer, E., Murphy, J. R., Boag, W., Weng, W.-H., Jin, D., Naumann, T., and McDermott, M. (2019). Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*. 11

Ammar, W., Mulcaire, G., Ballesteros, M., Dyer, C., and Smith, N. (2016a). Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444. 15, 31

Ammar, W., Mulcaire, G., Tsvetkov, Y., Lample, G., Dyer, C., and Smith, N. A. (2016b). Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*. 13

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 214–223. PMLR. 87

Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. (2018). Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495. 9

Artetxe, M., Labaka, G., and Agirre, E. (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. 12

Artetxe, M., Ruder, S., and Yogatama, D. (2020). On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics. 67, 69

Aufrant, L., Wisniewski, G., and Yvon, F. (2015). Zero-resource dependency parsing: Boosting delexicalized cross-lingual transfer with linguistic knowledge. In *COLING 2016, the 26th International Conference on Computational Linguistics*, pages 119–130. The COLING 2016 Organizing Committee. 88

Aufrant, L., Wisniewski, G., and Yvon, F. (2016). Zero-resource dependency parsing: Boosting delexicalized cross-lingual transfer with linguistic knowledge. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 119–130, Osaka, Japan. The COLING 2016 Organizing Committee. 15, 31

Bekoulis, G., Deleu, J., Demeester, T., and Develder, C. (2018). Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2830–2836, Brussels, Belgium. Association for Computational Linguistics. 52

Beltagy, I., Lo, K., and Cohan, A. (2019). Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural*

Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3606–3611. 11

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155. 7, 8

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017a). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146. 7, 9

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017b). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146. 21, 83

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics. 58

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311. 12

Bugliarello, E. and Okazaki, N. (2020). Enhancing machine translation with dependency-aware self-attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1618–1627, Online. Association for Computational Linguistics. 34, 73, 74

Buys, J. and Botha, J. A. (2016). Cross-lingual morphological tagging for low-resource languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1954–1964, Berlin, Germany. Association for Computational Linguistics. 30

Cao, S., Kitaev, N., and Klein, D. (2020). Multilingual alignment of contextual word representations. In *8th International Conference on Learning Representations (ICLR)*. 111

Chakraborty, S., Ding, Y., Allamanis, M., and Ray, B. (2020). Codit: Code editing with tree-based neural models. *IEEE Transactions on Software Engineering*, pages 1–1. 103

Chakraborty, S., Krishna, R., Ding, Y., and Ray, B. (2020). Deep learning based vulnerability detection: Are we there yet? *arXiv preprint arXiv:2009.07235*. 104

Chen, H., Huang, S., Chiang, D., and Chen, J. (2017). Improved neural machine translation with a syntax-aware encoder and decoder. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1936–1945, Vancouver, Canada. Association for Computational Linguistics. 74

Chen, Q., Zhuo, Z., and Wang, W. (2019a). Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*. 68

Chen, X. and Cardie, C. (2018). 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*, pages 1226–1240. 88

Chen, X., Sun, Y., Athiwaratkun, B., Cardie, C., and Weinberger, K. (2018). Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570. 88

Chen, Y., Xu, L., Liu, K., Zeng, D., and Zhao, J. (2015). Event extraction via dynamic multi-pooling convolutional neural 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 167–176, Beijing, China. Association for Computational Linguistics. 52

Chen, Z. and Ji, H. (2009). Can one language bootstrap the other: A case study on event extraction. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing*, pages 66–74, Boulder, Colorado. Association for Computational Linguistics. 53

Chen, Z.-Y., Chang, C.-H., Chen, Y.-P., Nayak, J., and Ku, L.-W. (2019b). UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 345–356, Minneapolis, Minnesota. Association for Computational Linguistics. 33

Chi, E. A., Hewitt, J., and Manning, C. D. (2020). Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics. 59

Clark, C. and Gardner, M. (2018). Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics. 68

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*. 99

Cohen, S. B., Das, D., and Smith, N. A. (2011). Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 50–61, Edinburgh, Scotland, UK. Association for Computational Linguistics. 88

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM. 7, 8

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537. 7, 8

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*. x, 13, 37, 45, 50

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics. 58, 91, 111

Conneau, A. and Lample, G. (2019). Cross-lingual language model pretraining. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 7059–7069. Curran Associates, Inc. 91, 93

Conneau, A., Rinott, R., Lample, G., Williams, A., Bowman, S., Schwenk, H., and Stoyanov, V. (2018). XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics. xvi, 58, 59, 67, 69

Cooper Stickland, A., Li, X., and Ghazvininejad, M. (2021). Recipes for adapting pre-trained monolingual and multilingual models to machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3440–3453, Online. Association for Computational Linguistics. 110

Cotterell, R. and Duh, K. (2017). Low-resource named entity recognition with cross-lingual, character-level neural conditional random fields. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 91–96. 30

Deguchi, H., Tamura, A., and Ninomiya, T. (2019). Dependency-based self-attention for transformer NMT. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 239–246, Varna, Bulgaria. INCOMA Ltd. 34, 74

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 11, 13, 83, 89

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics. x, 4, 37, 45, 50, 58, 61, 91, 99, 105, 111

Dou, Z.-Y. and Neubig, G. (2021). Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online. Association for Computational Linguistics. 111

Dozat, T. and Manning, C. D. (2017). Deep biaffine attention for neural dependency parsing. *Internation Conference on Learning Representations*. 19, 20, 21, 79, 81

Dryer, M. S. (2007). Word order. *Language typology and syntactic description*, 1:61–131. 16

Dryer, M. S. and Haspelmath, M., editors (2013a). *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig. 16

Dryer, M. S. and Haspelmath, M. (2013b). The world atlas of language structures online. 48

Duong, L., Cohn, T., Bird, S., and Cook, P. (2015). Cross-lingual transfer for unsupervised dependency parsing without parallel data. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 113–122, Beijing, China. Association for Computational Linguistics. 88

Duong, L., Kanayama, H., Ma, T., Bird, S., and Cohn, T. (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. 13

Dyer, C., Chahuneau, V., and Smith, N. A. (2013). A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics. 12

Eisner, J. M. (1996). Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 340–345. Association for Computational Linguistics. 20

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211. 8

Faruqui, M. and Dyer, C. (2014). Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter*

of the Association for Computational Linguistics, pages 462–471, Gothenburg, Sweden. Association for Computational Linguistics. 12

Faruqui, M. and Kumar, S. (2015). Multilingual open relation extraction using cross-lingual projection. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1351–1356, Denver, Colorado. Association for Computational Linguistics. 53

Feng, Z., Guo, D., Tang, D., Duan, N., Feng, X., Gong, M., Shou, L., Qin, B., Liu, T., Jiang, D., and Zhou, M. (2020). CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, Online. Association for Computational Linguistics. xii, 91, 98, 99, 105

Fu, T.-J., Li, P.-H., and Ma, W.-Y. (2019). GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of ACL*, pages 1409–1418. 34

Gamallo, P., Garcia, M., and Fernández-Lanza, S. (2012). Dependency-based open information extraction. In *Proceedings of the joint workshop on unsupervised and semi-supervised learning in NLP*, pages 10–18. Association for Computational Linguistics. 15

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030. 88

Glavaš, G., Litschko, R., Ruder, S., and Vulić, I. (2019). How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 710–721. 89

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680. 77, 87

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *Internation Conference on Learning Representations*. 87

Gouws, S., Bengio, Y., and Corrado, G. (2015). Bilbowa: Fast bilingual distributed representations without word alignments. In *International Conference on Machine Learning*, pages 748–756. PMLR. 11

Gui, T., Zhang, Q., Huang, H., Peng, M., and Huang, X. (2017). Part-of-speech tagging for twitter with adversarial neural networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2411–2420. 88

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777. 87

Guo, D., Ren, S., Lu, S., Feng, Z., Tang, D., Liu, S., Zhou, L., Duan, N., Yin, J., Jiang, D., et al. (2021). Graphcodebert: Pre-training code representations with data flow. In *International Conference on Learning Representations*. 91, 100

Guo, D., Tang, D., Duan, N., Zhou, M., and Yin, J. (2019). Coupling retrieval and meta-learning for context-dependent semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 855–866, Florence, Italy. Association for Computational Linguistics. 100

Guo, J., Che, W., Yarowsky, D., Wang, H., and Liu, T. (2015). Cross-lingual dependency parsing based on distributed representations. 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)*, volume 1, pages 1234–1244. 14, 22, 30, 31, 88

Guo, J., Che, W., Yarowsky, D., Wang, H., and Liu, T. (2016). A representation learning framework for multi-source transfer parsing. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2734–2740. 14, 31

Han, R., Ning, Q., and Peng, N. (2019). Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of EMNLP*, pages 434–444. 53

Han, R., Zhou, Y., and Peng, N. (2020). Domain knowledge empowered structured neural net for end-to-end event temporal relation extraction. In *Proceedings of EMNLP*, pages 5717–5729. 53

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162. 7

Hashimoto, K., xiong, c., Tsuruoka, Y., and Socher, R. (2017). A joint many-task model: Growing a neural network for multiple nlp tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933. Association for Computational Linguistics. 15

He, J., Zhang, Z., Berg-Kiripatrick, T., and Neubig, G. (2019). Cross-lingual syntactic transfer through unsupervised adaptation of invertible projections. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3211–3223. 88

Hewitt, J. and Manning, C. D. (2019). A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics. 59, 65, 66

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780. 4, 10, 19, 45

Hong, Y., Zhang, J., Ma, B., Yao, J., Zhou, G., and Zhu, Q. (2011). Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127–1136, Portland, Oregon, USA. Association for Computational Linguistics. 52

Hsi, A., Yang, Y., Carbonell, J., and Xu, R. (2016). Leveraging multilingual training for limited resource event extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1201–1210, Osaka, Japan. The COLING 2016 Organizing Committee. 53

Hu, J., Ruder, S., Siddhant, A., Neubig, G., Firat, O., and Johnson, M. (2020). XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4411–4421. PMLR. xi, 68, 69

Huang, B. and Carley, K. (2019). Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5469–5477, Hong Kong, China. Association for Computational Linguistics. 74

Huang, E., Socher, R., Manning, C., and Ng, A. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882, Jeju Island, Korea. Association for Computational Linguistics. 9

Huang, J.-T., Li, J., Yu, D., Deng, L., and Gong, Y. (2013). Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7304–7308. IEEE. 30

Huang, K.-H., Yang, M., and Peng, N. (2020). Biomedical event extraction with hierarchical knowledge graphs. In *Findings of ACL: EMNLP 2020*. 52

Huang, L., Ji, H., Cho, K., Dagan, I., Riedel, S., and Voss, C. (2018). Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics. 33

Husain, H., Wu, H.-H., Gazit, T., Allamanis, M., and Brockschmidt, M. (2019). Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*. 96, 99

Iyer, S., Cheung, A., and Zettlemoyer, L. (2019). Learning programmatic idioms for scalable semantic parsing. In *Proceedings of the 2019 Conference on Empirical*

Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 5426–5435, Hong Kong, China. Association for Computational Linguistics. 100

Iyer, S., Konstas, I., Cheung, A., and Zettlemoyer, L. (2018). Mapping language to code in programmatic context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1652, Brussels, Belgium. Association for Computational Linguistics. xii, 96, 97, 100

Jawahar, G., Sagot, B., and Seddah, D. (2019). What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics. 59

Ji, H. and Grishman, R. (2008). Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics. 43, 52

Jie, Z., Muis, A. O., and Lu, W. (2017). Efficient dependency-guided named entity recognition. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3457–3465. 15

Joty, S., Nakov, P., Màrquez, L., and Jaradat, I. (2017). Cross-language learning with adversarial neural networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 226–237. 31, 88

Joulin, A., Bojanowski, P., Mikolov, T., Jégou, H., and Grave, E. (2018). Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2984, Brussels, Belgium. Association for Computational Linguistics. x, 44, 50

K, K., Wang, Z., Mayhew, S., and Roth, D. (2020). Cross-lingual ability of multilingual bert: An empirical study. In *International Conference on Learning Representations*. 58

Kann, K., Cotterell, R., and Schütze, H. (2017). One-shot neural cross-lingual transfer for paradigm completion. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, page 1993–2003. 30

Kim, J.-K., Kim, Y.-B., Sarikaya, R., and Fosler-Lussier, E. (2017). Cross-lingual transfer learning for pos tagging without cross-lingual resources. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2832–2838. 14, 30

Kim, S., Jeong, M., Lee, J., and Lee, G. G. (2010a). A cross-lingual annotation projection approach for relation detection. In *Proceedings of COLING*, pages 564–571. 53

Kim, S., Jeong, M., Lee, J., and Lee, G. G. (2010b). A cross-lingual annotation projection approach for relation detection. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 564–571, Beijing, China. Coling 2010 Organizing Committee. 53

Kim, S. and Lee, G. G. (2012). A graph-based cross-lingual projection approach for weakly supervised relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 48–53, Jeju Island, Korea. Association for Computational Linguistics. 53

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations*. 83

Kiperwasser, E. and Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327. 19

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *ICLR*. 34, 38, 45, 53

Kitaev, N. and Klein, D. (2018). Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686. Association for Computational Linguistics. 15

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. (2017). OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demo*. 46

Klementiev, A., Titov, I., and Bhattarai, B. (2012). Inducing crosslingual distributed representations of words. In *Proceedings of COLING 2012*, pages 1459–1474, Mumbai, India. The COLING 2012 Organizing Committee. 11, 12

Kočiský, T., Hermann, K. M., and Blunsom, P. (2014). Learning bilingual word representations by marginalizing alignments. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 224–229, Baltimore, Maryland. Association for Computational Linguistics. 12

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics. xii, 102

Kozhevnikov, M. and Titov, I. (2013). Cross-lingual transfer of semantic role labeling models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1200, Sofia, Bulgaria. Association for Computational Linguistics. 75

Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics. 93

Kundu, G., Sil, A., Florian, R., and Hamza, W. (2018). Neural cross-lingual coreference resolution and its application to entity linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 395–400. Association for Computational Linguistics. 30

Lachaux, M.-A., Roziere, B., Chanussot, L., and Lample, G. (2020). Unsupervised translation of programming languages. In *Advances in Neural Information Processing Systems*, volume 33, pages 20601–20611. Curran Associates, Inc. 91, 93

Lample, G. and Conneau, A. (2019). Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*. 13

Lample, G., Conneau, A., Denoyer, L., Jégou, H., et al. (2018). Word translation without parallel data. In *Internation Conference on Learning Representations*. 88

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324. 8

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020a). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics. 13, 91, 94, 96, 105

Lewis, P., Oguz, B., Rinott, R., Riedel, S., and Schwenk, H. (2020b). MLQA: Evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics. xi, xvi, 58, 59, 60, 67, 69, 71

Li, H., Arora, A., Chen, S., Gupta, A., Gupta, S., and Mehdad, Y. (2021). MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics. 68, 69

Li, J., Monroe, W., Shi, T., Jean, S., Ritter, A., and Jurafsky, D. (2017). Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark. Association for Computational Linguistics. 88

Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., and Chang, K.-W. (2019). Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*. 91

Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., and Chang, K.-W. (2020). What does BERT with vision look at? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5265–5275, Online. Association for Computational Linguistics. 105

Li, Q. and Ji, H. (2014). Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 402–412, Baltimore, Maryland. Association for Computational Linguistics. 43, 52

Li, Q., Ji, H., Hong, Y., and Li, S. (2014). Constructing information networks using one single model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1846–1851, Doha, Qatar. Association for Computational Linguistics. 52

Li, Q., Ji, H., and Huang, L. (2013). Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics. 43, 52

Liang, Y., Duan, N., Gong, Y., Wu, N., Guo, F., Qi, W., Gong, M., Shou, L., Jiang, D., Cao, G., Fan, X., Zhang, R., Agrawal, R., Cui, E., Wei, S., Bharti, T., Qiao, Y., Chen, J.-H., Wu, W., Liu, S., Yang, F., Campos, D., Majumder, R., and Zhou, M. (2020). XGLUE: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics. xi, 67, 69

Liao, S. and Grishman, R. (2010). Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797, Uppsala, Sweden. Association for Computational Linguistics. 52

Liao, S. and Grishman, R. (2011). Acquiring topic features to improve event extraction: in pre-selected and balanced collections. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 9–16, Hissar, Bulgaria. Association for Computational Linguistics. 52

Lin, C.-Y. and Och, F. J. (2004). ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 501–507, Geneva, Switzerland. COLING. 98

Lin, Y., Liu, Z., and Sun, M. (2017). Neural relation extraction with multi-lingual attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–43, Vancouver, Canada. Association for Computational Linguistics. 53

Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*, page 2181–2187. 33

Litschko, R., Glavaš, G., Ponzetto, S. P., and Vulić, I. (2018). Unsupervised cross-lingual information retrieval using monolingual data only. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pages 1253–1256. 30

Littell, P., Mortensen, D. R., Lin, K., Kairis, K., Turner, C., and Levin, L. (2017). Uriel and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 8–14, Valencia, Spain. Association for Computational Linguistics. 16

Liu, D., Niehues, J., Cross, J., Guzmán, F., and Li, X. (2021). Improving zero-shot translation by disentangling positional information. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1259–1273, Online. Association for Computational Linguistics. 110

Liu, H. (2010). Dependency direction as a means of word-order typology: A method based on dependency treebanks. *Lingua*, 120(6):1567–1578. 16, 17

Liu, J., Chen, Y., Liu, K., and Zhao, J. (2018a). Event detection via gated multilingual attention mechanism. In *Proceedings of AAAI*, page 4865–4872. 52, 53

Liu, J., Chen, Y., Liu, K., and Zhao, J. (2019a). Neural cross-lingual event detection with minimal parallel resources. In *Proceedings of EMNLP-IJCNLP*, pages 738–748. 34, 36, 44, 52, 53

Liu, J., Chen, Y., Liu, K., and Zhao, J. (2019b). Neural cross-lingual event detection with minimal parallel resources. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 738–748, Hong Kong, China. Association for Computational Linguistics. 75

Liu, P., Qiu, X., and Huang, X. (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. 88

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., and Shazeer, N. (2018b). Generating wikipedia by summarizing long sequences. *Internation Conference on Learning Representations*. 15

Liu, X., Luo, Z., and Huang, H. (2018c). Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference*

on Empirical Methods in Natural Language Processing, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics. 33, 34, 52

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742. 13, 94, 95

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019c). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*. xii, 13, 91, 99, 102, 105

Lu, A., Wang, W., Bansal, M., Gimpel, K., and Livescu, K. (2015). Deep multilingual correlation for improved word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 250–256, Denver, Colorado. Association for Computational Linguistics. 12

Lu, S., Guo, D., Ren, S., Huang, J., Svyatkovskiy, A., Blanco, A., Clement, C., Drain, D., Jiang, D., Tang, D., et al. (2021). Codexglue: A machine learning benchmark dataset for code understanding and generation. *arXiv preprint arXiv:2102.04664*. xii, 96, 97, 100, 102

Lu, W. and Nguyen, T. H. (2018). Similar but not the same: Word sense disambiguation improves event detection via neural representation matching. In *Proceedings of EMNLP*, pages 4822–4828. 52

Luong, T., Pham, H., and Manning, C. D. (2015a). Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, Denver, Colorado. Association for Computational Linguistics. 12

Luong, T., Pham, H., and Manning, C. D. (2015b). 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. Association for Computational Linguistics. 99

Ma, X., Hu, Z., Liu, J., Peng, N., Neubig, G., and Hovy, E. (2018). Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 20, 21, 79, 81

Ma, X. and Xia, F. (2014a). Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization. In *Proceedings of ACL 2014*, pages 1337–1348, Baltimore, Maryland. 30

Ma, X. and Xia, F. (2014b). Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1337–1348. 88

McCann, B., Bradbury, J., Xiong, C., and Socher, R. (2017). Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305. 10

McClosky, D., Surdeanu, M., and Manning, C. D. (2011). Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1626–1635. Association for Computational Linguistics. 15

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of ACL-2005*, pages 91–98, Ann Arbor, Michigan, USA. 20

McDonald, R., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K., Petrov, S., Zhang, H., Täckström, O., et al. (2013). Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 92–97. 31

McDonald, R., Petrov, S., and Hall, K. (2011). Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the conference on empirical methods in natural language processing*, pages 62–72. Association for Computational Linguistics. 31

Meng, T., Peng, N., and Chang, K.-W. (2019). Target language-aware constrained inference for cross-lingual dependency parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1117–1128, Hong Kong, China. Association for Computational Linguistics. 53

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. 7, 8

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*. 8, 15

Mikolov, T., Le, Q. V., and Sutskever, I. (2013b). Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*. 11, 12

Miwa, M. and Bansal, M. (2016). End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics. 52

Miyato, T., Dai, A. M., and Goodfellow, I. (2017). Adversarial training methods for semi-supervised text classification. In *Internation Conference on Learning Representations*. 88

Naseem, T., Barzilay, R., and Globerson, A. (2012). Selective sharing for multilingual dependency parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 629–637, Jeju Island, Korea. Association for Computational Linguistics. 15, 31

Nguyen, T. H., Cho, K., and Grishman, R. (2016). Joint event extraction via recurrent neural networks. In *Proceedings of NAACL*, pages 300–309. 52

Nguyen, T. H. and Grishman, R. (2015). Event detection and domain adaptation with convolutional neural networks. In *Proceedings of EMNLP-IJCNLP*, pages 365–371. 52

Nguyen, T. H. and Grishman, R. (2018). Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of AAAI*. 52

Ni, J. and Florian, R. (2019). Neural cross-lingual relation extraction based on bilingual word embedding mapping. In *Proceedings of EMNLP-IJCNLP*, pages 399–409. 36, 45, 53

Nivre, J., Abrams, M., Agić, Ž., Ahrenberg, L., Aleksandravičiūtė, G., Antonsen, L., Aplonova, K., Aranzabe, M. J., Arutie, G., Asahara, M., Ateyah, L., Attia, M., Atutxa, A., Augustinus, L., Badmaeva, E., Ballesteros, M., Banerjee, E., Bank, S., Barbu Mititelu, V., Basmov, V., Bauer, J., Bellato, S., Bengoetxea, K., Berzak, Y., Bhat, I. A., Bhat, R. A., Biagetti, E., Bick, E., Bielinskienė, A., Blokland, R., Bobicev, V., Boizou, L., Borges Völker, E., Börstell, C., Bosco, C., Bouma, G., Bowman, S., Boyd, A., Brokaitė, K., Burchardt, A., Candito, M., Caron, B., Caron, G., Cebiroğlu Eryiğit, G., Cecchini, F. M., Celano, G. G. A., Čéplö, S., Cetin, S., Chalub, F., Choi, J., Cho, Y., Chun, J., Cinková, S., Collomb, A., Çöltekin, Ç., Connor, M., Courtin, M., Davidson, E., de Marneffe, M.-C., de Paiva, V., Diaz de Ilarraza, A., Dickerson, C., Dione, B., Dirix, P., Dobrovoljc, K., Dozat, T., Droganova, K., Dwivedi, P., Eckhoff, H., Eli, M., Elkahky, A., Ephrem, B., Erjavec, T., Etienne, A., Farkas, R., Fernandez Alcalde, H., Foster, J., Freitas, C., Fujita, K., Gajdošová, K., Galbraith, D., Garcia, M., Gärdenfors, M., Garza, S., Gerdes, K., Ginter, F., Goenaga, I., Gojenola, K., Gökırmak, M., Goldberg, Y., Gómez Guinovart, X., González Saavedra, B., Grioni, M., Grūzītis, N., Guillaume, B., Guillot-Barbance, C., Habash, N., Hajič, J., Hajič jr., J., Hà Mỹ, L., Han, N.-R., Harris, K., Haug, D., Heinecke, J., Hennig, F., Hladká, B., Hlaváčová, J., Hociung, F., Hohle, P., Hwang, J., Ikeda, T., Ion, R., Irimia, E., Ishola, Ọ., Jelínek, T., Johannsen, A., Jørgensen, F., Kaşıkara, H., Kaasen, A., Kahane, S., Kanayama, H., Kanerva, J., Katz, B., Kayadelen, T., Kenney, J., Kettnerová, V., Kirchner, J., Köhn, A., Kopacewicz, K., Kotsyba, N., Kovalevskaitė, J., Krek, S., Kwak, S., Laippala, V., Lambertino, L., Lam, L., Lando, T., Larasati, S. D., Lavrentiev, A., Lee, J., Lê Hồng, P., Lenci, A., Lertpradit, S., Leung, H., Li, C. Y., Li, J., Li, K., Lim, K., Li, Y., Ljubešić, N., Loginova, O., Lyashevskaya, O., Lynn, T., Macketanz, V., Makazhanov, A., Mandl, M., Manning, C., Manurung, R., Mărănduc, C., Mareček, D., Marheinecke, K., Martínez Alonso, H., Martins, A., Mašek, J., Matsumoto, Y., McDonald, R., McGuinness, S., Mendonça, G., Miekka, N., Misirpashayeva, M., Missilä, A., Mititelu, C., Miyao, Y., Montemagni, S., More, A., Moreno Romero, L., Mori, K. S., Morioka, T., Mori, S., Moro, S., Mortensen, B., Moskalevskyi, B., Muischnek, K.,

Murawaki, Y., Müürisep, K., Nainwani, P., Navarro Horñiacek, J. I., Nedoluzhko, A., Nešpore-Bērzkalne, G., Nguyễn Thị, L., Nguyễn Thị Minh, H., Nikaido, Y., Nikolaev, V., Nitisaroj, R., Nurmi, H., Ojala, S., Olúòkun, A., Omura, M., Osenova, P., Östling, R., Øvrelid, L., Partanen, N., Pascual, E., Passarotti, M., Patejuk, A., Paulino-Passos, G., Peljak-Łapińska, A., Peng, S., Perez, C.-A., Perrier, G., Petrova, D., Petrov, S., Piitulainen, J., Pirinen, T. A., Pitler, E., Plank, B., Poibeau, T., Popel, M., Pretkalniņa, L., Prévost, S., Prokopidis, P., Przepiórkowski, A., Puolakainen, T., Pyysalo, S., Rääbis, A., Rademaker, A., Ramasamy, L., Rama, T., Ramisch, C., Ravishankar, V., Real, L., Reddy, S., Rehm, G., Rießler, M., Rimkutė, E., Rinaldi, L., Rituma, L., Rocha, L., Romanenko, M., Rosa, R., Rovati, D., Roșca, V., Rudina, O., Rueter, J., Sadde, S., Sagot, B., Saleh, S., Salomoni, A., Samardžić, T., Samson, S., Sanguinetti, M., Särg, D., Saulīte, B., Sawanakunanon, Y., Schneider, N., Schuster, S., Seddah, D., Seeker, W., Seraji, M., Shen, M., Shimada, A., Shirasu, H., Shohibussirri, M., Sichinava, D., Silveira, N., Simi, M., Simionescu, R., Simkó, K., Šimková, M., Simov, K., Smith, A., Soares-Bastos, I., Spadine, C., Stella, A., Straka, M., Strnadová, J., Suhr, A., Sulubacak, U., Suzuki, S., Szántó, Z., Taji, D., Takahashi, Y., Tamburini, F., Tanaka, T., Tellier, I., Thomas, G., Torga, L., Trosterud, T., Trukhina, A., Tsarfaty, R., Tyers, F., Uematsu, S., Urešová, Z., Uria, L., Uszkoreit, H., Vajjala, S., van Niekerk, D., van Noord, G., Varga, V., Villemonte de la Clergerie, E., Vincze, V., Wallin, L., Walsh, A., Wang, J. X., Washington, J. N., Wendt, M., Williams, S., Wirén, M., Wittern, C., Woldemariam, T., Wong, T.-s., Wróblewska, A., Yako, M., Yamazaki, N., Yan, C., Yasuoka, K., Yavrumyan, M. M., Yu, Z., Žabokrtský, Z., Zeldes, A., Zeman, D., Zhang, M., and Zhu, H. (2019). Universal dependencies 2.4. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. 66

Nivre, J., Abrams, M., Agić, Ž., and et al. (2018). Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. 15, 16, 83

Nivre, J., De Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R. T., Petrov, S., Pyysalo, S., Silveira, N., et al. (2016). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. 78

Östling, R. (2015). Word order typology through multilingual word alignment. 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)*, volume 2, pages 205–211. 17

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics. 95

Pan, L., Hang, C.-W., Qi, H., Shah, A., Potdar, S., and Yu, M. (2021). Multilingual BERT post-pretraining alignment. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 210–219, Online. Association for Computational Linguistics. 111

Pan, X., Zhang, B., May, J., Nothman, J., Knight, K., and Ji, H. (2017). Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics. 67, 69

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. 98

Pawlik, M. and Augsten, N. (2015). Efficient computation of the tree edit distance. *ACM Transactions on Database Systems (TODS)*, 40(1):1–40. 50

Pawlik, M. and Augsten, N. (2016). Tree edit distance: Robust and memory-efficient. *Information Systems*, pages 157–173. 50

Peng, N., Poon, H., Quirk, C., Toutanova, K., and Yih, W.-t. (2017). Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115. 52

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*, pages 1532–1543. 7, 8

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018a). 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. Association for Computational Linguistics. 15

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018b). Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 10

Petrov, S., Das, D., and McDonald, R. (2012). A universal part-of-speech tagset. In *Proceedings of LREC-2012*, pages 2089–2096, Istanbul, Turkey. 31

Pires, T., Schlinger, E., and Garrette, D. (2019). How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics. 58

Pražák, O. and Konopík, M. (2017). Cross-lingual SRL based upon Universal Dependencies. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 592–600, Varna, Bulgaria. INCOMA Ltd. 75

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. (2020). Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics. 73

Qian, L., Hui, H., Hu, Y., Zhou, G., and Zhu, Q. (2014). Bilingual active learning for relation classification via pseudo parallel corpora. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Baltimore, Maryland. Association for Computational Linguistics. 53

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*. 11, 105

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9. 100

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics. 58, 67

Rasooli, M. S. and Collins, M. (2015). Density-driven cross-lingual transfer of dependency parsers. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 328–338. 88

Rasooli, M. S. and Collins, M. (2019a). Low-resource syntactic transfer with unsupervised source reordering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3845–3856, Minneapolis, Minnesota. Association for Computational Linguistics. 53

Rasooli, M. S. and Collins, M. (2019b). Low-resource syntactic transfer with unsupervised source reordering. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 88

Reisinger, J. and Mooney, R. J. (2010). Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117, Los Angeles, California. Association for Computational Linguistics. 9

Ren, S., Guo, D., Lu, S., Zhou, L., Liu, S., Tang, D., Zhou, M., Blanco, A., and Ma, S. (2020). Codebleu: a method for automatic evaluation of code synthesis. *arXiv preprint arXiv:2009.10297*. 98

Ruder, S., Søgaard, A., and Vulic, I. (2017). A survey of cross-lingual embedding models. *arXiv preprint arXiv:1706.04902*. 89

Ruder, S., Vulić, I., and Søgaard, A. (2019). A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631. 13

Sachan, D., Zhang, Y., Qi, P., and Hamilton, W. L. (2021). Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661, Online. Association for Computational Linguistics. 74

Sasaki, S., Sun, S., Schamoni, S., Duh, K., and Inui, K. (2018). Cross-lingual learning-to-rank with shared representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 458–463. 30

Sato, M., Manabe, H., Noji, H., and Matsumoto, Y. (2017). Adversarial training for cross-domain universal dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 71–79. 88

Schlichtkrull, M. and Søgaard, A. (2017). Cross-lingual dependency parsing with late decoding for truly low-resource languages. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 220–229. Association for Computational Linguistics. 88

Shaw, P., Uszkoreit, J., and Vaswani, A. (2018a). Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468. Association for Computational Linguistics. 19, 24, 79

Shaw, P., Uszkoreit, J., and Vaswani, A. (2018b). Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics. 45

Sil, A., Kundu, G., Florian, R., and Hamza, W. (2018). Neural cross-lingual entity linking. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5464–5472. 14, 30

Smith, S. L., Turban, D. H., Hamblin, S., and Hammerla, N. Y. (2017). Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *Internation Conference on Learning Representations*. 12, 13, 18, 21, 83

Søgaard, A. (2011). Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 682–686. Association for Computational Linguistics. 31

Solaiman, I., Brundage, M., Clark, J., Askell, A., Herbert-Voss, A., Wu, J., Radford, A., Krueger, G., Kim, J. W., Kreps, S., et al. (2019). Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*. 105

Straka, M. and Straková, J. (2017). Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics. 37, 73

Strubell, E., Verga, P., Andor, D., Weiss, D., and McCallum, A. (2018). Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics. 74

Subburathinam, A., Lu, D., Ji, H., May, J., Chang, S.-F., Sil, A., and Voss, C. (2019a). Cross-lingual structure transfer for relation and event extraction. In *Proceedings of EMNLP-IJCNLP*, pages 313–325. 33, 34, 36, 37, 38, 41, 43, 45, 51, 52, 53

Subburathinam, A., Lu, D., Ji, H., May, J., Chang, S.-F., Sil, A., and Voss, C. (2019b). Cross-lingual structure transfer for relation and event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 313–325, Hong Kong, China. Association for Computational Linguistics. 75

Sun, C., Gong, Y., Wu, Y., Gong, M., Jiang, D., Lan, M., Sun, S., and Duan, N. (2019a). Joint type inference on entities and relations via graph convolutional networks. In *Proceedings of ACL*, pages 1361–1370. 34

Sun, C., Myers, A., Vondrick, C., Murphy, K., and Schmid, C. (2019b). Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473. 91, 105

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112. 15

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In *Internation Conference on Learning Representations*. 87

Täckström, O., McDonald, R., and Nivre, J. (2013). Target language adaptation of discriminative transfer parsers. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1061–1071, Atlanta, Georgia. Association for Computational Linguistics. 15

Täckström, O., McDonald, R., and Nivre, J. (2013). Target language adaptation of discriminative transfer parsers. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1061–1071. Association for Computational Linguistics. 31, 88

Täckström, O., McDonald, R., and Uszkoreit, J. (2012a). 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, Montréal, Canada. Association for Computational Linguistics. 12

Täckström, O., McDonald, R., and Uszkoreit, J. (2012b). 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. 31

Tang, H., Ji, D., Li, C., and Zhou, Q. (2020). Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of ACL*, pages 6578–6588. 34

Tiedemann, J. (2015). Cross-lingual dependency parsing with universal dependencies and predicted pos labels. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 340–349. 30, 88

Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31. 84

Tjong Kim Sang, E. F. (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)*. 67, 69

Tjong Kim Sang, E. F. and De Meulder, F. (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*, pages 142–147. 67, 69

Tufano, M., Watson, C., Bavota, G., Penta, M. D., White, M., and Poshyvanyk, D. (2019). An empirical study on learning bug-fixing patches in the wild via neural machine translation. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 28(4):1–29. 96

Turian, J., Ratinov, L., and Bengio, Y. (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. 7

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008. 4, 15, 18, 34, 35, 37, 45, 51, 53, 58, 61, 74, 91, 94, 99, 105

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph Attention Networks. In *International Conference on Learning Representations*. 60, 61, 62, 75

Vulić, I. and Moens, M.-F. (2015a). 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. 11

Vulić, I. and Moens, M.-F. (2015b). Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 363–372. ACM. 30

Walker, C., Strassel, S., Medero, J., and Maeda, K. (2006). Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57. 35, 43

Wang, C., Wu, S., and Liu, S. (2019a). Source dependency-aware transformer with supervised self-attention. *arXiv preprint arXiv:1909.02273*. 74

Wang, D. and Eisner, J. (2017). Fine-grained prediction of syntactic typology: Discovering latent structure with supervised learning. *Transactions of the Association for Computational Linguistics*, 5:147–161. 17

Wang, D. and Eisner, J. (2018a). Surface statistics of an unknown language indicate how to parse it. *Transactions of the Association for Computational Linguistics (TACL)*. 31

Wang, D. and Eisner, J. (2018b). Synthetic data made to order: The case of parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1325–1337. 31

Wang, W., Li, G., Ma, B., Xia, X., and Jin, Z. (2020). Detecting code clones with graph neural network and flow-augmented abstract syntax tree. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 261–271. IEEE. 96

Wang, W. and Pan, S. J. (2018). Transition-based adversarial network for cross-lingual aspect extraction. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4475–4481. International Joint Conferences on Artificial Intelligence Organization. 88

Wang, X., Han, X., Lin, Y., Liu, Z., and Sun, M. (2018). Adversarial multi-lingual neural relation extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1156–1166, Santa Fe, New Mexico, USA. Association for Computational Linguistics. 53

Wang, X., Tu, Z., Wang, L., and Shi, S. (2019b). Self-attention with structural position representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1403–1409, Hong Kong, China. Association for Computational Linguistics. x, 47, 48, 74

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics. 68

Wu, S. and Dredze, M. (2019a). Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics. 58, 59

Wu, S. and Dredze, M. (2019b). Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 83, 89

Wu, S., Zhou, M., and Zhang, D. (2017a). Improved neural machine translation with source syntax. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4179–4185. 74, 75

Wu, Y., Bamman, D., and Russell, S. (2017b). Adversarial training for relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1778–1783. 88

Xiao, M. and Guo, Y. (2014). Distributed word representation learning for cross-lingual dependency parsing. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 119–129. 14, 31, 88

Xie, J., Yang, Z., Neubig, G., Smith, N. A., and Carbonell, J. (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. 30

Xie, Z., Zhu, R., Zhao, K., Liu, J., Zhou, G., and Huang, J. X. (2020). A contextual alignment enhanced cross graph attention network for cross-lingual entity alignment. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5918–5928, Barcelona, Spain (Online). International Committee on Computational Linguistics. 75

Xu, R. and Yang, Y. (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. Association for Computational Linguistics. 88

Xu, W., Haider, B., and Mansour, S. (2020). End-to-end slot alignment and recognition for cross-lingual NLU. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5052–5063, Online. Association for Computational Linguistics. 68, 69

Xu, Y., Du, J., Dai, L.-R., and Lee, C.-H. (2014). Cross-language transfer learning for deep neural network based speech enhancement. In *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*, pages 336–340. IEEE. 30

Yang, B. and Mitchell, T. M. (2016). Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics. 53

Yang, Y., Zhang, Y., Tar, C., and Baldridge, J. (2019a). PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692, Hong Kong, China. Association for Computational Linguistics. 67, 69

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019b). Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763. 13

Yang, Z., Salakhutdinov, R., and Cohen, W. (2016). Multi-task cross-lingual sequence tagging from scratch. *arXiv preprint arXiv:1603.06270*. 30

Yasunaga, M., Kasai, J., and Radev, D. (2018). Robust multilingual part-of-speech tagging via adversarial training. 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 976–986, New Orleans, Louisiana. Association for Computational Linguistics. 88

Yin, P. and Neubig, G. (2017). A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Vancouver, Canada. Association for Computational Linguistics. 103

Zeman, D., Hajič, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J., and Petrov, S. (2018). Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21. Association for Computational Linguistics. 21

Zeman, D. and Resnik, P. (2008). Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*. 31

Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics. 52

Zhang, C., Li, Q., and Song, D. (2019a). Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4568–4578, Hong Kong, China. Association for Computational Linguistics. 34

Zhang, J., Zhou, W., Hong, Y., Yao, J., and Zhang, M. (2018a). Using entity relation to improve event detection via attention mechanism. In *CCF International Conference on NLPCC*, pages 171–183. 52

Zhang, M., Liu, Y., Luan, H., and Sun, M. (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. 88

Zhang, T., Ji, H., and Sil, A. (2019b). Joint entity and event extraction with generative adversarial imitation learning. *Data Intelligence*, pages 99–120. 52

Zhang, Y. and Barzilay, R. (2015). Hierarchical low-rank tensors for multilingual transfer parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1857–1867, Lisbon, Portugal. Association for Computational Linguistics. 15, 31

Zhang, Y., Qi, P., and Manning, C. D. (2018b). Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215, Brussels, Belgium. Association for Computational Linguistics. 34, 38, 41, 46, 52

Zhang, Y., Wang, R., and Si, L. (2019c). Syntax-enhanced self-attention-based semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 616–626, Hong Kong, China. Association for Computational Linguistics. 74, 75

Zhang, Z., Wu, Y., Zhou, J., Duan, S., Zhao, H., and Wang, R. (2020). SG-Net: Syntax-guided machine reading comprehension. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*. 74

Zhou, H., Chen, L., Shi, F., and Huang, D. (2015). Learning bilingual sentiment word embeddings for cross-language sentiment 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)*, volume 1, pages 430–440. 31

Zhou, J. T., Pan, S. J., Tsang, I. W., and Ho, S.-S. (2016a). Transfer learning for cross-language text categorization through active correspondences construction. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2400–2406. 14

Zhou, X., Wan, X., and Xiao, J. (2016b). 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)*, volume 1, pages 1403–1412. 31

Zhou, Y., Liu, S., Siow, J., Du, X., and Liu, Y. (2019). Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32, pages 10197–10207. Curran Associates, Inc. 96, 104

Zhu, Z., Li, S., Zhou, G., and Xia, R. (2014). Bilingual event extraction: a case study on trigger type determination. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 842–847, Baltimore, Maryland. Association for Computational Linguistics. 53

Zoph, B., Yuret, D., May, J., and Knight, K. (2016). Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics. 14

Zou, B., Xu, Z., Hong, Y., and Zhou, G. (2018a). Adversarial feature adaptation for cross-lingual relation classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 437–448, Santa Fe, New Mexico, USA. Association for Computational Linguistics. 53

Zou, B., Xu, Z., Hong, Y., and Zhou, G. (2018b). Adversarial feature adaptation for cross-lingual relation classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 437–448. 88

Zou, W. Y., Socher, R., Cer, D., and Manning, C. D. (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. Association for Computational Linguistics. 11, 12