UNIVERSITY OF CALIFORNIA
Los Angeles

Building Reliable and Robust Natural Language Processing Models: Enhancing Understanding of Semantically Equivalent Texts

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

by

Kuan-Hao Huang

2023

ABSTRACT OF THE DISSERTATION

Building Reliable and Robust Natural Language Processing Models: Enhancing Understanding of Semantically Equivalent Texts

by

Kuan-Hao Huang
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2023
Professor Kai-Wei Chang, Chair

Recently, research in the natural language processing (NLP) domain has achieved remarkable advancements. Machines have become increasingly intelligent, achieving human performance in several NLP benchmarks. Despite its potential, recent studies demonstrate that NLP systems are not as reliable and robust as we expect and are sensitive to different levels of modifications to the input text, including word-level, syntax-level, and language-level. Those modifications do not alter the meaning of input text but would make NLP models behave very differently, which deviates from human expectations. This robustness issue results in challenges when applying NLP models to real-world applications and therefore becomes an important research question in recent years. In this thesis, we highlight the crucial role of understanding semantically equivalent texts in resolving the robustness issue of NLP models. We put emphasis on enhancing NLP models' comprehension of semantically equivalent texts while focusing specifically on improving the syntax-level robustness and language-level robustness of NLP models. The first part of this thesis focuses on enhancing syntax-level robustness by disentangling semantics and syntax when learning text representations. We propose three different ways to separate syntax from semantics: learning with paraphrase pairs, learning with unannotated texts, and learning with abstract meaning representations. By adopting these methods, NLP models become less sensitive to syntax and more robust to syntactic perturbations. In the second

part of this thesis, we improve language-level robustness by considering the zero-shot cross-lingual setting. We propose two methods to enhance zero-shot cross-lingual transfer: robust training and the utilization of generation-based models. The proposed approaches in this dissertation effectively improve the reliability and robustness of NLP models.

The dissertation of Kuan-Hao Huang is approved.

Wei Wang

Yizhou Sun

Cho-Jui Hsieh

Kai-Wei Chang, Committee Chair

University of California, Los Angeles

2023

To those who support me and trust me.

Table of Contents

List of Figures

List of Tables

Acknowledgements

First and foremost, I would like to thank my advisor, Kai-Wei Chang. Throughout the past 4.5 years, he gave me many useful and insightful pieces of advice, not limited to research but also including other decisions for my academic career. Most importantly, he provided me with a lot of freedom to do research and placed complete trust in my ability. This allowed me to explore any research topics I like and experiment with unconventional ideas. I am really grateful and enjoy the time working with Kai-Wei. I also want to express my gratitude to my mentor, Nanyun Peng. She is like my second advisor, and always gives me valuable and insightful suggestions for both research and career.

Many thanks to the members of the UCLA NLP Group and PLUS Lab, not only for your research feedback but also for your company during my Ph.D. journey. It will be an unforgettable experience in my life. I particularly want to thank my close collaborator, I-Hung Hsu. We have finished many interesting research projects together. I thoroughly enjoyed the time we spent together discussing exciting research directions and exploring various crazy ideas.

I am extremely grateful to Ruty Rinott, Anoop Kumar, Aram Galstyan, Rashmi Gangadharaiah, and Chen Li, who were my mentor during my internships at Meta AI, Amazon Alexa AI, Amazon AWS AI, and Tencent AI. They provided me with opportunities to experience doing research in the industry. I learned a lot from them.

During my Ph.D. journey, I am fortunate to have the chance to collaborate with many others, including Wasi Uddin Ahmad, Tanmay Parekh, Premkumar Natarajan, Varun Iyer, James Y. Huang, Jieyu Zhao, Muhao Chen, Fei Wang, Yixin Wan, Pei Zhou, Weijia Shi, Elizabeth Boschee, Scott Miller, Zhiyu Xie, Shuning Zhang, Wenxin Cheng, Liang Tan, Rui Hou, Sinong Wang, Amjad Almahairi, Sriram Venkatapathy, Greg Ver Steeg, and Ryan Cotterell. Many of the published work with them contribute to this thesis.

Finally, I thank those who always support me and trust me. You are the reason why I am here.

Curriculum Vitae

2010-2014	B.S. in Computer Science and Information Engineering
	National Taiwan University, Taipei, Taiwan
2014-2016	M.S. in Computer Science and Information Engineering
	National Taiwan University, Taipei, Taiwan
2019	Research Intern
	Tencent AI Lab, Bellevue, Washington, USA
2021	Applied Scientist Intern
	Amazon AWS AI, Santa Clara, California, USA
2022	Applied Scientist Intern
	Amazon Alexa AI, Manhattan Beach, California, USA
2022	Research Intern
	Meta AI, Seattle, Washington, USA
2018-2023	Research Assistant, UCLA Natural Language Processing Group
	University of California, Los Angeles, California, USA

Publications

Kuan-Hao Huang, Varun Iyer, I-Hung Hsu, Anoop Kumar, Kai-Wei Chang, and Aram Galstyan. ParaAMR: A large-scale syntactically diverse paraphrase dataset by amr back-translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

I-Hung Hsu*, Kuan-Hao Huang*, Shuning Zhang, Wenxin Cheng, Premkumar Natarajan, Kai-Wei Chang, and Nanyun Peng. TAGPRIME: A unified framework for relational structure extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023a.

I-Hung Hsu*, Zhiyu Xie*, Kuan-Hao Huang, Premkumar Natarajan, and Nanyun Peng. AMPERE: Amr-aware prefix for generation-based event argument extraction model. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023b.

Tanmay Parekh, I-Hung Hsu, Kuan-Hao Huang, Kai-Wei Chang, and Nanyun Peng. GENEVA: Benchmarking generalizability for event argument extraction with hundreds of event types and argument roles. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Yixin Wan, Kuan-Hao Huang, and Kai-Wei Chang. PIP: Parse-instructed prefix for syntactically controlled paraphrase generation. In *Findings of the Association for Computational Linguistics: ACL 2023 (ACL-Findings)*, 2023.

Kuan-Hao Huang*, Varun Iyer*, Anoop Kumar, Sriram Venkatapathy, Kai-Wei Chang, and Aram Galstyan. Unsupervised syntactically controlled paraphrase generation with abstract meaning representations. In *Findings of the Association for Computational Linguistics: EMNLP 2022 (EMNLP-Findings)*, 2022a.

Fei Wang, Kuan-Hao Huang, Anoop Kumar, Aram Galstyan, Greg Ver Steeg, and Kai-Wei Chang. Zero-shot cross-lingual sequence tagging as seq2seq generation for joint intent classification and slot filling. In *Workshop on Massively Multilingual Natural Language Understanding, EMNLP (MMNLU@EMNLP)*, 2022.

I-Hung Hsu*, Kuan-Hao Huang*, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. DEGREE: A data-efficient generation-based event extraction model. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2022.

Kuan-Hao Huang*, I-Hung Hsu*, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. Multilingual generative language models for zero-shot cross-lingual event argument extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022b.

Kuan-Hao Huang, Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. Improving zero-shot cross-lingual transfer learning via robust training. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021a.

James Y. Huang, Kuan-Hao Huang, and Kai-Wei Chang. Disentangling semantics and syntax in sentence embeddings with pre-trained language models. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021b.

Kuan-Hao Huang and Kai-Wei Chang. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Kuan-Hao Huang, Chen Li, and Kai-Wei Chang. Generating sports news from live commentary: A chinese dataset for sports game summarization. In *Proceedings of the Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (AACL)*, 2020.

Pei Zhou, Weijia Shi, Jieyu Zhao, Kuan-Hao Huang, Muhao Chen, Ryan Cotterell, and Kai-Wei Chang. Examining gender bias in languages with grammatical gender. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

CHAPTER 1

Introduction

1.1 Overview

In recent years, the field of natural language processing (NLP) has made incredible success in research. Machines have become more and more intelligent, surpassing humans in several NLP benchmarks (Bubeck et al., 2023). Despite its potential, recent studies have demonstrated that NLP systems are not as reliable and robust as we expect (Alzantot et al., 2018; Iyyer et al., 2018). Sometimes, minor modifications to the input text can lead to significant changes in the NLP system's behavior, even when the meaning of the input text does not change (Jin et al., 2020; Li et al., 2020b; Garg and Ramakrishnan, 2020). Figure 1.1 illustrates one example when applying NLP models to a smart assistant system. Users ask the smart assistant the same question in different ways but get very different responses, which deviates from users' expectations.

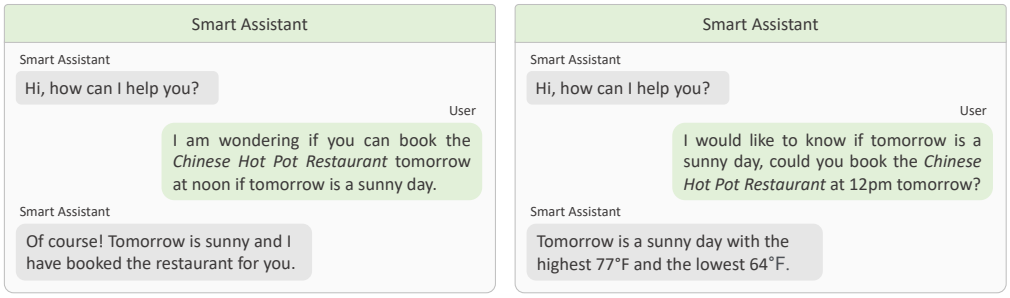


Figure 1.1: An example of how NLP systems behave differently for semantically similar texts.

One of the reasons is that existing NLP models are not robust enough to well capture the meaning of semantically similar texts. Hence, even a small modification

in the input text, such as synonym substitutions or changes in word order, can cause NLP systems to perceive it as an entirely different text, which leads to unexpected responses and behaviors. This robustness issue has become a big obstacle to making NLP techniques more realistic and hinders people from developing reliable real-world NLP applications. Several studies have pointed out that NLP systems can suffer from different levels of robustness issues that can be categorized as follows.

Word-level robustness issue. It is possible to fool NLP models' understanding of texts by replacing words with their synonyms (Alzantot et al., 2018; Li et al., 2020b; Garg and Ramakrishnan, 2020). For example, given a sentence *"I like to play volleyball."*, simply replacing *"like"* with *"love"* might change NLP models' behaviors and predictions a lot, even if the modifications do not alter the semantics of texts for humans.

Syntax-level robustness issue. Prior work has demonstrated that existing NLP models are not robust to phrase rewriting or paraphrasing (Iyyer et al., 2018; Huang and Chang, 2021). For instance, the two sentences *"Paris is the capital city of France."* and *"Paris is France's capital city."* have similar semantics and only differ only in one phrase. However, NLP models can behave very differently for the two sentences. Another example of paraphrasing is *"We will go hiking if tomorrow is a sunny day."* and *"If it is sunny tomorrow, we will go hiking."*, where two sentences have different syntactic structures but share a similar meaning. Existing NLP models may fail to have consistent behaviors.

Language-level robustness issue. The third category involves the change of languages. When we use a different language to express the same thing, NLP models cannot connect their meanings very well (Hu et al., 2020; Liang et al., 2020). For example, NLP models can give us completely different predictions for *"This restaurant is very good."* and its translation in French *"Ce resto est très bon."*

How to improve the robustness of NLP models, especially for enhancing the understanding of semantically equivalent texts, has attracted increasing attention in

recent years. It not only benefits building reliable real-world NLP systems but also helps us to develop more human-like artificial intelligence.

1.2 Contributions and Structure of the Thesis

The main contribution of this thesis is to enhance NLP models' understanding of semantically equivalent texts and improve the robustness of NLP models. We particularly focus on syntax-level robustness and language-level robustness.

In the first part of this thesis, we discuss the syntax-level robustness and argue that the existing commonly used text representations (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019b) are strongly affected by syntax. As a result, NLP models can have a high sensitivity to syntactic changes within texts, leading to inconsistent behaviors when encountering semantically equivalent texts with syntactic differences. This motivates us to *separate* semantics and syntax of texts. We discuss different ways to encode texts into two separate embeddings: semantic embeddings and syntactic embeddings. In Chapter 2, we demonstrate that disentangled semantic representations and syntactic representations can be learned from *annotated paraphrase pairs*. Chapter 3 shows that we can separate semantics and syntax without using annotating paraphrase pairs but using a large number of *unannotated texts* instead. Chapter 4 further improves the quality of disentanglement of semantics and syntax by introducing *abstract meaning representations* (Banarescu et al., 2013).

In the second part, we focus on the language-level robustness. We first introduce the relation between zero-shot cross-lingual transfer and language-level robustness. In Chapter 5, we show that *robust training techniques* can improve the performance of zero-shot cross-lingual transfer without modifying pre-trained multilingual text representations, leading to better language-level robustness. Chapter 6 reveals that *generation-based models* can create a language-agnostic space to align knowledge from different languages, resulting in better performance than traditional classification-based models on zero-shot cross-lingual transfer.

Finally, we summarize our conclusions and discuss potentual future research directions in Chapter 7.

1.3 Other Publications

During my Ph.D. study, I have published other research work related to the topics of information extraction, text generation, and bias analysis. For the field of information extraction, we demonstrate the power of generation-based models for even extraction (Hsu et al., 2022, 2023b), show the effectiveness of priming on relational structure extraction (Hsu et al., 2023a), as well as propose new resources for event argument extraction (Parekh et al., 2023). For text generation, we study how to leverage additional knowledge to guide generation (Wan et al., 2023) and construct new datasets for diverse generation (Huang et al., 2023) and summarization (Huang et al., 2020). We also analyze the potential gender bias in NLP models and provide solutions Zhou et al. (2019).

Part I

Syntax-Level Robustness

Text representations can be viewed as the way how NLP models understand texts. However, we observe that existing commonly used text representations, such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), and Sentence-BERT Reimers and Gurevych (2019), are strongly affected by syntax. This makes NLP models sensitive to syntactic changes within texts. For example, let us consider two pairs of sentences: (1) *"We will go hiking if tomorrow is a sunny day"* and *"If it is sunny tomorrow, we will go hiking."*, where the two sentences are semantically similar but syntactically different, and (2) *"We will go hiking if tomorrow is a sunny day."* and *"We will go swimming if tomorrow is a sunny day."*, where the two sentences are semantically different but syntactically similar. In our preliminary experiment, there is a 50% of chance that existing text representations will give us higher text similarity for the second pair, which deviates from human understanding. We argue that the existing text representations are strongly affected by syntax and therefore propose to *separate* semantics and syntax of texts. Specifically, we encode a text into disentangled semantic embeddings and syntactic embeddings. By separating syntactic information from semantic embeddings, the learned text representations can capture semantics better without being affected by syntax too much.

In Chapter 2, 3, and 4, we will introduce three different ways to learn such disentanglement of semantics and syntax: learning with paraphrase pairs, learning with unannotated texts, learning with abstract meaning representations.

CHAPTER 2

Improving Syntax-Level Robustness with Paraphrase Pairs

2.1 Introduction

Semantic sentence embedding models encode sentences into fixed-length vectors based on their semantic relatedness with each other. If two sentences are more semantically related, their corresponding sentence embeddings are closer. As sentence embeddings can be used to measures semantic relatedness without requiring supervised data, they have been used in many applications, such as semantic textual similarity (Agirre et al., 2016), question answering (Nakov et al., 2017), and natural language inference (Artetxe and Schwenk, 2019).

Recent years have seen huge success of pre-trained language models across a wide range of NLP tasks (Devlin et al., 2019; Lewis et al., 2020a). However, several studies (Reimers and Gurevych, 2019; Li et al., 2020a) have found that sentence embeddings from pre-trained language models perform poorly on semantic similarity tasks when the models are not fine-tuned on task-specific data. Meanwhile, Goldberg (2019) shows that BERT without fine-tuning performs surprisingly well on syntactic tasks. Hence, we posit that these contextual representations from pre-trained language models without fine-tuning capture entangled semantic and syntactic information, and therefore are not suitable for sentence-level semantic tasks.

Ideally, the semantic embedding of a sentence should not encode its syntax, and two semantically similar sentences should have close semantic embeddings regardless of their syntactic differences. While various models (Conneau et al., 2017; Cer et al., 2018; Reimers and Gurevych, 2019) have been proposed to improve the performance

of sentence embeddings on downstream semantic tasks, most of these approaches do not attempt to separate syntactic information from sentence embeddings.

To this end, we propose ParaBART, a semantic sentence embedding model that learns to disentangle semantics and syntax in sentence embeddings. Our model is built upon BART (Lewis et al., 2020a), a sequence-to-sequence Transformer (Vaswani et al., 2017) model pre-trained with self-denoising objectives. Parallel paraphrase data is a good source of learning the distinction between semantics and syntax, as paraphrase pairs naturally share the same meaning but often differ in syntax. Taking advantage of this fact, ParaBART is trained to perform syntax-guided paraphrasing, where a source sentence containing the desired semantics and a parse tree specifying the desired syntax are given as inputs. In order to generate a paraphrase that follows the given syntax, ParaBART uses separate encoders to learn disentangled semantic and syntactic representations from their respective inputs. In this way, the disentangled representations capture sufficient semantic and syntactic information needed for paraphrase generation. The semantic encoder is also encouraged to ignore the syntax of the source sentence, as the desired syntax is already provided by the syntax input.

ParaBART achieves strong performance across unsupervised semantic textual similarity tasks. Furthermore, semantic embeddings learned by ParaBART contain significantly less syntactic information as suggested by probing results, and yield robust performance on datasets with syntactic variation. Our source code is available at https://github.com/uclanlp/ParaBART.

2.2 Related Work

Various sentence embedding models have been proposed in recent years. Most of these models utilize supervision from parallel data (Wieting and Gimpel, 2018; Artetxe and Schwenk, 2019; Wieting et al., 2019, 2020), natural language inference data (Conneau et al., 2017; Cer et al., 2018; Reimers and Gurevych, 2019), or a combination of both (Subramanian et al., 2018).

Many efforts towards controlled text generation have been focused on learning disentangled sentence representations (Hu et al., 2017; Fu et al., 2018; John et al., 2019). In the context of disentangling semantics and syntax, Bao et al. (2019) and

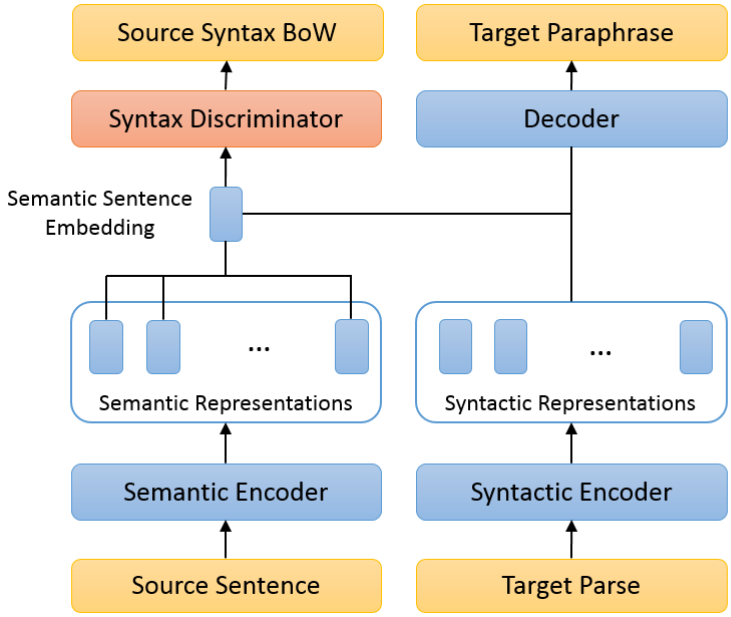


Figure 2.1: An overview of ParaBART. The model extracts semantic and syntactic representations from a source sentence and a target parse respectively, and uses both the semantic sentence embedding and the target syntactic representations to generate the target paraphrase. ParaBART is trained in an adversarial setting, with the syntax discriminator (red) trying to decode the source syntax from the semantic embedding, and the paraphrasing model (blue) trying to fool the syntax discriminator and generate the target paraphrase at the same time.

Chen et al. (2019a) utilize variational autoencoders to learn two latent variables for semantics and syntax. In contrast, we use the outputs of a constituency parser to learn purely syntactic representations, and facilitate the usage of powerful pre-trained language models as semantic encoders.

Our approach is also related to prior work on syntax-controlled paraphrase generation (Iyyer et al., 2018; Kumar et al., 2020; Goyal and Durrett, 2020; Huang and Chang, 2021). While these approaches focus on generating high-quality paraphrases that conform to the desired syntax, we are interested in how semantic and syntactic information can be disentangled and how to obtain good semantic sentence embeddings.

2.3 Proposed Model – ParaBART

Our goal is to build a semantic sentence embedding model that learns to separate syntax from semantic embeddings. ParaBART is trained to generate syntax-guided paraphrases, where the model attempts to only extract the semantic part from the input sentence, and combine it with a different syntax specified by the additional syntax input in the form of a constituency parse tree.

Figure 2.1 outlines the proposed model, which consists of a semantic encoder that learns the semantics of a source sentence, a syntactic encoder that encodes the desired syntax of a paraphrase, and a decoder that generates a corresponding paraphrase. Additionally, we add a syntax discriminator to adversarially remove syntactic information from the semantic embeddings.

Given a source sentence S_1 and a target constituency parse tree P_2, ParaBART is trained to generate a paraphrase S_2 that shares the semantics of S_1 and conforms to the syntax specified by P_2. Semantics and syntax are two key aspects that determine how a sentence is generated. Our model learns purely syntactic representations from the output trees generated by a constituency parser, and extracts the semantic embedding directly from the source sentence. The syntax discriminator and the syntactic encoder are designed to remove source syntax and provide target syntax, thus encouraging the semantic encoder to only capture source semantics.

Semantic encoder. The semantic encoder E_{sem} is a Transformer encoder that embeds a sentence $S = (s^{(1)}, ..., s^{(m)})$ into contextual semantic representations:

$$U = (\mathbf{u}^{(1)}, ..., \mathbf{u}^{(m)}) = E_{sem}\left((s^{(1)}, ..., s^{(m)})\right).$$

Then, we take the mean of these contextual representations $\mathbf{u}^{(i)}$ to get a fixed-length semantic sentence embedding

$$\bar{\mathbf{u}} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{u}^{(i)}.$$

Syntactic encoder. The syntactic encoder E_{syn} is a Transformer encoder that takes a linearized constituency parse tree $P = (p^{(1)}, ..., p^{(n)})$ and converts it into

contextual syntactic representations

$$V = (\mathbf{v}^{(1)}, ..., \mathbf{v}^{(n)}) = E_{syn}\left((p^{(1)}, ..., p^{(n)})\right).$$

For example, the linearized parse tree of the sentence "This book is good." is "(S (NP (DT) (NN)) (VP (VBZ) (ADJP)) (.))". Such input sequence preserves the tree structure, allowing the syntactic encoder to capture the exact syntax needed for decoding.

Decoder. The decoder D_{dec} uses the semantic sentence embedding $\bar{\mathbf{u}}$ and the contextual syntactic representations V to generate a paraphrase that shares semantics with the source sentence while following the syntax of the given parse tree. In other words,

$$(y^{(1)}, ..., y^{(l)}) = D_{dec}\left(\text{Concat}(\bar{\mathbf{u}}, V)\right).$$

During training, given a source sentence S_1, a target parse tree P_2 and a target paraphrase $S_2 = (s_2^1, ..., s_2^l)$, we minimize the following *paraphrase generation loss*:

$$\mathcal{L}_{para} = -\sum_{i=1}^{l} \log P(y^{(i)} = s_2^{(i)} | S_1, P_2).$$

Since the syntactic representations do not contain semantics, the semantic encoder needs to accurately capture the semantics of the source sentence for a paraphrase to be generated. Meanwhile, the full syntactic structure of the target is provided by the syntactic encoder, thus encouraging the semantic encoder to ignore the source syntax.

Syntax discriminator. To further encourage the disentanglement of semantics and syntax, we employ a syntax discriminator to adversarially remove syntactic information from semantic embeddings. We first train the syntax discriminator to predict the syntax from its semantic embedding, and then train the semantic encoder to "fool" the syntax discriminator such that the source syntax cannot be predicted from the semantic embedding.

More specifically, we adopt a simplified approach similar to John et al. (2019) by encoding source syntax as a Bag-of-Words vector $\mathbf{h}$ of its constituency parse tree. For

any given source parse tree, this vector contains the count of occurrences of every constituent tag, divided by the total number of constituents in the parse tree. Given the semantic sentence embedding $\bar{\mathbf{u}}$, our linear syntax discriminator D_{dis} predicts $\mathbf{h}$ by

$$\mathbf{y}_h = D_{dis}(\bar{\mathbf{u}}) = \text{softmax}(\mathbf{W}\bar{\mathbf{u}} + \mathbf{b})$$

with the following *adversarial loss*:

$$\mathcal{L}_{adv} = -\sum_{t \in T} \mathbf{h}(t) \log(\mathbf{y}_h(t)),$$

where T denotes the set of all constituent tags.

Training. We adversarially train E_{sem}, E_{syn}, D_{dec}, and D_{dis} with the following objective:

$$\min_{E_{sem}, E_{syn}, D_{dec}} \left(\max_{D_{dis}} \left(\mathcal{L}_{para} - \lambda_{adv} \mathcal{L}_{adv} \right) \right),$$

where λ_{adv} is a hyperparameter to balance loss terms. In each iteration, we update the D_{dis} by considering the inner optimization, and then update E_{sem}, E_{syn} and D_{dec} by considering the outer optimization.

2.4 Experiments

In this section, we demonstrate that ParaBART is capable of learning semantic sentence embeddings that capture semantic similarity, contain less syntactic information, and yield robust performance against syntactic variation on semantic tasks.

2.4.1 Setup

Datasets. We use the ParaNMT-50M dataset released by Wieting and Gimpel (2018), which can be obtained from https://github.com/jwieting/para-nmt-50m. We sample 1 million English paraphrase pairs from ParaNMT-50M, and split this dataset into 5000 pairs as the validation set and the rest as our training set. STS and syntactic probing datasets are directly taken from SentEval, which can be accessed from https://github.com/facebookresearch/SentEval. Quora Question Pairs are

downloaded from the official GLUE Benchmark website (https://gluebenchmark.com/).

Word dropout. We observe that some paraphrase pairs in our training set contain many overlapping words, which means our model can learn to generate the target paraphrase by just copying words from a source sentence without fully understanding the semantics of the sentence. To alleviate this issue, we apply word dropout (Iyyer et al., 2015) that randomly masks a portion of the input tokens. We don't apply word dropout to syntactic inputs, as these inputs are designed to provide the exact syntactic structure of the paraphrase and encourage disentanglement of syntactic and semantic representations. We set the word dropout probability to 0.2 for all our models.

Hyperparameter search. Hyperparameters of ParaBART are tuned manually based on the paraphrase generation loss on the validation set. Specifically, the weight of adversarial loss is tuned within {0.1, 0.2, 0.5, 1.0}. Word dropout is selected from {0.0, 0.1, 0.2, 0.4}. Learning rate is tuned within $\{1,2,5,10\}\times10^{-5}$.

None of the previous models we compare in this work involves any hyperparameter search. The results for BGT are taken from Wieting et al. (2020). For all other sentence embedding models, we use the trained model provided by their respective authors. These models include InferSent[1], USE[2], Sentence-$\text{BERT}_{\text{base}}$[3] and VGVAE[4].

Baselines. We compare our model with other sentence embeddings models, including InferSent (Conneau et al., 2017), Universal Sentence Encoder (USE) (Cer et al., 2018), Sentence-$\text{BERT}_{\text{base}}$ (Reimers and Gurevych, 2019), VGVAE (Chen et al., 2019a), and BGT (Wieting et al., 2020). We also include mean-pooled $\text{BERT}_{\text{base}}$ and $\text{BART}_{\text{base}}$ embeddings. In addition to ParaBART, we consider two model ablations: ParaBART without adversarial loss, and ParaBART without syntactic guidance and adversarial loss.

[1] https://github.com/facebookresearch/InferSent
[2] https://tfhub.dev/google/universal-sentence-encoder-large/2
[3] https://github.com/UKPLab/sentence-transformers
[4] https://github.com/mingdachen/syntactic-template-g

Model	STS12	STS13	STS14	STS15	STS16	STS-B	Avg.
Avg. BERT embeddings	46.9	52.8	57.2	63.5	64.5	47.9	55.5
Avg. BART embeddings	50.8	42.8	56.1	63.9	59.5	52.0	54.2
InferSent	59.3	59.0	70.0	71.5	71.5	70.0	66.9
VGVAE	61.8	62.2	69.2	72.5	67.8	74.2	68.0
USE	61.4	63.5	70.6	74.3	73.9	74.2	69.7
Sentence-BERT	64.6	67.5	73.2	74.3	70.1	74.1	70.6
BGT	**68.9**	62.2*	75.9	79.4	79.3	-	-
ParaBART	68.4	**71.1**	**76.4**	80.7	**80.1**	78.5	**75.9**
- w/o AL	67.5	70.0	75.8	**80.9**	80.0	**78.7**	75.5
- w/o AL and SG	66.4	65.3	73.6	80.0	78.6	75.4	73.2

Table 2.1: Pearson's r (in percentage) between cosine similarity of sentence embeddings and gold labels on STS tasks from 2012 to 2016 and STS Benchmark test set. BGT results are taken from Wieting et al. (2020). AL and SG denote adversarial loss and syntactic guidance, respectively. *BGT is evaluated on an additional dataset from STS13, which is not included in the standard SentEval toolkit.

2.4.2 Semantic Textual Similarity

We evaluate our semantic sentence embeddings on the unsupervised Semantic Textual Similarity (STS) tasks from SemEval 2012 to 2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016) and STS Benchmark test set (Cer et al., 2017), where the goal is to predict a continuous-valued score between 0 and 5 indicating how similar the meanings of a sentence pair are. For all models, we compute the cosine similarity of embedding vectors as the semantic similarity measure. We use the standard SentEval toolkit (Conneau and Kiela, 2018) for evaluation and report average Pearson correlation over all domains.

As shown in Table 2.1, both average BERT embeddings and average BART embeddings perform poorly on STS tasks, as the entanglement of semantic and syntactic information leads to low correlation with semantic similarity. Training ParaBART on paraphrase data substantially improves the correlation. With the addition of syntactic guidance and adversarial loss, ParaBART achieves the best overall performance across STS tasks, showing the effectiveness of our approach.

Model	**BShift**	**TreeDepth**	**TopConst**
Avg. BART embed.	90.5	47.8	80.1
ParaBART	**72.4**	**33.9**	**67.2**
- w/o AL	75.4	36.6	71.7
- w/o AL and SG	83.3	46.5	83.1

Table 2.2: Results on syntactic probing tasks. Semantic embeddings with lower accuracy on downstream syntactic tasks contain less syntactic information, suggesting better disentanglement of semantics and syntax. AL and SG denote adversarial loss and syntactic guidance, respectively.

2.4.3 Syntactic Probing

To better understand how well our model learns to disentangle syntactic information from semantic embeddings, we probe our semantic sentence embeddings with downstream syntactic tasks. Following Conneau et al. (2018a), we investigate to what degree our semantic sentence embeddings can be used to identify bigram word reordering (BShift), estimate parse tree depth (TreeDepth), and predict parse tree top-level constituents (TopConst). Top-level constituents are defined as the group of constituency parse tree nodes immediately below the sentence (S) node. We use the datasets provided by SentEval (Conneau and Kiela, 2018) to train a Multi-Layer Perceptron classifier with a single 50-neuron hidden layer on top of semantic sentence embeddings, and report accuracy on all tasks.

As shown in Table 2.2, sentence embeddings pooled from pre-trained BART model contain rich syntactic information that can be used to accurately predict syntactic properties including word order and top-level constituents. The disentanglement induced by ParaBART is evident, lowering the accuracy of downstream syntactic tasks by more than 10 points compared to pre-trained BART embeddings and ParaBART without adversarial loss and syntactic guidance. The results suggest that the semantic sentence embeddings learned by ParaBART indeed contain less syntactic information.

2.4.4 Robustness Against Syntactic Variation

Intuitively, semantic sentence embedding models that learn to disentangle semantics and syntax are expected to yield more robust performance on datasets with high

QQP-Easy
What are the essential skills of the project management? What are the essential skills of a project manager?
QQP-Hard
Is there a reason why we should travel alone? What are some reasons to travel alone?

Table 2.3: Examples of paraphrase pairs from *QQP-Easy* and *QQP-Hard*.

Model	**QQP-Easy**	**QQP-Hard**
Avg. BART embed.	72.3	64.1
InferSent	72.1	67.5
VGVAE	71.5	67.1
USE	**80.7**	72.4
Sentence-BERT	74.3	70.7
ParaBART	76.5	**72.7**
- w/o AL	76.8	72.1
- w/o AL and SG	76.1	69.9

Table 2.4: Results on *QQP-Easy* and *QQP-Hard*. For every model we report the highest accuracy after finding the best threshold. AL and SG denote adversarial loss and syntactic guidance, respectively.

syntactic variation. We consider the task of paraphrase detection on Quora Question Pairs (Iyer et al., 2017) dev set as a testbed for evaluating model robustness. We categorize paraphrase pairs based on whether they share the same top-level constituents. We randomly sample 1,000 paraphrase pairs from each of the two classes, combined with a common set of 1,000 randomly sampled non-paraphrase pairs, to create two datasets *QQP-Easy* and *QQP-Hard*. Paraphrase pairs from *QQP-Hard* are generally harder to identify as they are much more syntactically different compared to those from *QQP-Easy*. Table 2.3 shows some examples from these two datasets. We evaluate semantic sentence embeddings on these datasets in an unsupervised manner by computing the cosine similarity as the semantic similarity measure. We search for the best threshold between -1 and 1 with a step size of 0.01 on each dataset, and report the highest accuracy. The results are shown in Table 2.4.

While Universal Sentence Encoder scores much higher than other models on *QQP-Easy*, its performance degrades significantly on *QQP-Hard*. In comparison,

ParaBART demonstrates better robustness against syntactic variation, and surpasses USE to become the best model on the more syntactically diverse *QQP-Hard*. It is worth mentioning that even pre-trained BART embeddings give decent results on *QQP-Easy*, suggesting large overlaps between paraphrase pairs from *QQP-Easy*. On the other hand, the poor performance of pre-trained BART embeddings on a more syntactically diverse dataset like *QQP-Hard* clearly shows its incompetence as semantic sentence embeddings.

2.5 Summary

In this paper, we present ParaBART, a semantic sentence embedding model that learns to disentangle semantics and syntax in sentence embeddings from pre-trained language models. Experiments show that our semantic sentence embeddings yield strong performance on unsupervised semantic similarity tasks. Further investigation demonstrates the effectiveness of disentanglement, and robustness of our semantic sentence embeddings against syntactic variation on downstream semantic tasks.

CHAPTER 3

Improving Syntax-Level Robustness with Unannotated Texts

3.1 Introduction

Paraphrase generation (McKeown, 1983) is a long-lasting task in natural language processing (NLP) and has been greatly improved by recently developed machine learning approaches and large data collections. Paraphrase generation demonstrates the potential of machines in semantic abstraction and sentence reorganization and has already been applied to many NLP downstream applications, such as question answering (Yu et al., 2018), chatbot engines (Yan et al., 2016), and sentence simplification (Zhao et al., 2018).

In recent years, various approaches have been proposed to train sequence-to-sequence (seq2seq) models on a large number of annotated paraphrase pairs (Prakash et al., 2016; Mallinson et al., 2017; Cao et al., 2017; Egonmwan and Chali, 2019). Some of them control the syntax of output sentences to improve the diversity of paraphrase generation (Iyyer et al., 2018; Goyal and Durrett, 2020; Kumar et al., 2020). However, collecting annotated pairs is expensive and induces challenges for some languages and domains. On the contrary, unsupervised approaches build paraphrase models without using parallel corpora (Li et al., 2018; Roy and Grangier, 2019; Zhang et al., 2019a). Most of them are based on the variational autoencoder (Bowman et al., 2016) or back-translation (Mallinson et al., 2017; Wieting and Gimpel, 2018; Hu et al., 2019). Nevertheless, without the consideration of controlling syntax, their generated paraphrases are often similar to the source sentences and are not diverse in syntax.

This chapter presents a pioneering study on syntactically controlled paraphrase

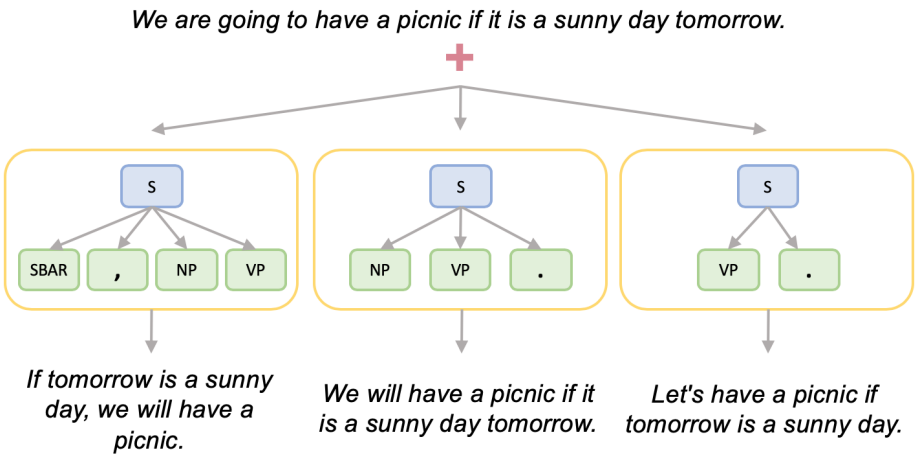


Figure 3.1: Paraphrase generation with syntactic control. Given a source sentence and a target syntactic specification (either a full parse tree or top levels of a parse tree), the model is expected to generate a paraphrase with the syntax following the given specification.

generation based on disentangling semantics and syntax. We aim to disentangle one sentence into two parts: 1) the semantic part and 2) the syntactic part. The semantic aspect focuses on the meaning of the sentence, while the syntactic part represents the grammatical structure. When two sentences are paraphrased, their semantic aspects are supposed to be similar, while their syntactic parts should be different. To generate a syntactically different paraphrase of one sentence, we can keep its semantic part unchanged and modify its syntactic part.

Based on this idea, we propose **Syn**tactically Controlled **P**araphrase **G**enerator (SynPG)[1], a Transformer-based model (Vaswani et al., 2017) that can generate syntactically different paraphrases of one source sentence based on some target syntactic parses. SynPG consists of a semantic encoder, a syntactic encoder, and a decoder. The semantic encoder considers the source sentence as a bag of words without ordering and learns a contextualized embedding containing only the semantic information. The syntactic encoder embeds the target parse into a contextualized embedding including only the syntactic information. Then, the decoder combines the two representations and generates a paraphrase sentence. The design of disentangling semantics and syntax enables SynPG to learn the association between words and parses and be trained

[1]Our code and the pretrained models are available at https://github.com/uclanlp/synpg

by reconstructing the source sentence given its unordered words and its parse. Therefore, we do not require any annotated paraphrase pairs but only unannotated texts to train SynPG.

We verify SynPG on four paraphrase datasets: ParaNMT-50M (Wieting and Gimpel, 2018), Quora (Iyer et al., 2017), PAN (Madnani et al., 2012), and MRPC (Dolan et al., 2004). The experimental results reveal that when being provided with the syntactic structures of the target sentences, SynPG can generate paraphrases with the syntax more similar to the ground truth than the unsupervised baselines. The human evaluation results indicate that SynPG achieves competitive paraphrase quality to other baselines while its generated paraphrases are more accurate in following the syntactic specifications. In addition, we show that when the training data is large enough, the performance of SynPG is competitive or even better than supervised approaches. Finally, we demonstrate that the syntactically controlled paraphrases generated by SynPG can be used for data augmentation to defense syntactically adversarial attack (Iyyer et al., 2018) and improve the robustness of NLP models.

3.2 Unsupervised Paraphrase Generation

We aim to train a paraphrase model without using annotated paraphrase pairs. Given a source sentence $\mathbf{x} = (x_1, x_2, ..., x_n)$, our goal is to generate a paraphrase sentence $\mathbf{y} = (y_1, y_2, ..., y_m)$ that is expected to maintain the same meaning of $\mathbf{x}$ but has a different syntactic structure from $\mathbf{x}$.

Syntactic control. Motivated by previous work (Iyyer et al., 2018; Zhang et al., 2019a; Kumar et al., 2020), we allow our model to access additional syntactic specifications as the control signals to guide the paraphrase generation. More specifically, in addition to the source sentence $\mathbf{x}$, we give the model a target constituency parse $\mathbf{p}$ as another input. Given the input $(\mathbf{x}, \mathbf{p})$, the model is expected to generate a paraphrase $\mathbf{y}$ that is semantically similar to the source sentence $\mathbf{x}$ and syntactically follows the target parse $\mathbf{p}$. In the following discussions, we assume the target parse $\mathbf{p}$ to be a full constituency parse tree. Later on, in Section 3.2.3, we will relax the syntax guidance to be a *template*, which is defined as the top two levels of a full parse

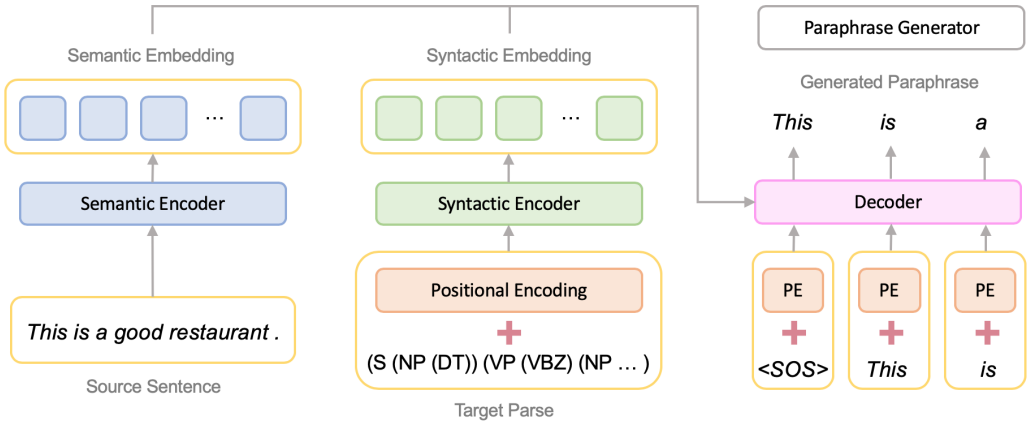


Figure 3.2: SynPG embeds the source sentence and the target parse into a semantic embedding and a syntactic embedding, respectively. Then, SynPG generates a paraphrase sentence based on the two embeddings.

tree. We expect that a successful model can control the syntax of output sentences and generate syntactically different paraphrases based on different target parses, as illustrated in Figure 3.1.

Similar to previous work (Iyyer et al., 2018; Zhang et al., 2019a), we linearize the constituency parse tree to a sequence. For example, the linearized parse of the sentence "*He eats apples.*" is `(S(NP(PRP))(VP(VBZ)(NP(NNS)))(.))`. Accordingly, a parse tree can be considered as a sentence $\mathbf{p} = (p_1, p_2, ..., p_k)$, where the tokens in $\mathbf{p}$ are non-terminal symbols and parentheses.

3.2.1 Proposed Model

Our main idea is to disentangle a sentence into the semantic part and the syntactic part. Once the model learns the disentanglement, it can generate a syntactically different paraphrase of one given sentence by keeping its semantic part unchanged and modifying only the syntactic part.

Figure 3.2 illustrates the proposed paraphrase model called SynPG, a seq2seq model consisting of a semantic encoder, a syntactic encoder, and a decoder. The semantic encoder captures only the semantic information of the source sentence $\mathbf{x}$, while the syntactic encoder extracts only the syntactic information from the target parse $\mathbf{p}$. The decoder then combines the encoded semantic and syntactic information and generates a paraphrase $\mathbf{y}$. We discuss the details of SynPG in the following.

Semantic encoder. The semantic encoder embeds a source sentence $\mathbf{x}$ into a contextualized semantic embedding $\mathbf{z}_{sem}$. In other words,

$$\mathbf{z}_{sem} = (z_1, z_2, ..., z_n) = \text{Enc}_{sem}((x_1, x_2, .., x_n)).$$

The semantic embedding $\mathbf{z}_{sem}$ is supposed to contain only the semantic information of the source sentence $\mathbf{x}$. To separate the semantic information from the syntactic information, we use a Transformer (Vaswani et al., 2017) *without the positional encoding* as the semantic encoder. We posit that by removing position information from the source sentence $\mathbf{x}$, the semantic embedding $\mathbf{z}_{sem}$ would encode less syntactic information.

We assume that words without ordering capture most of the semantics of one sentence. Indeed, semantics is also related to the order. For example, exchanging the subject and the object of a sentence changes its meaning. However, the decoder trained on a large corpus also captures the *selectional preferences* (Katz and Fodor, 1963; Wilks, 1975) in generation, which enables the decoder to infer the proper order of words. In addition, we observe that when two sentences are paraphrased, they usually share similar words, especially those words related to the semantics. For example, "*What is the best way to improve writing skills?*" and "*How can I improve my writing skills?*" are paraphrased, and the shared words (*improve*, *writing*, and *skills*) are strongly related to the semantics. In Section 3.4, we show that our designed semantic embedding captures enough semantic information to generate paraphrases.

Syntactic encoder. The syntactic encoder embeds the target parse $\mathbf{p} = (p_1, p_2, ..., p_k)$ into a contextualized syntactic embedding $\mathbf{z}_{syn}$. That is,

$$\mathbf{z}_{syn} = (z_1, z_2, ..., z_k) = \text{Enc}_{syn}((p_1, p_2, .., p_k)).$$

Since the target parse $\mathbf{p}$ contains no semantic information but only syntactic information, we use a Transformer *with the positional encoding* as the syntactic encoder.

Decoder. Finally, we design a decoder that takes the semantic embedding $\mathbf{z}_{sem}$ and the syntactic embedding $\mathbf{z}_{syn}$ as the input and generates a paraphrase $\mathbf{y}$. In other

words,

$$\mathbf{y} = (y_1, y_2, ..., y_m) = \text{Dec}(\mathbf{z}_{sem}, \mathbf{z}_{syn}).$$

We choose Transformer as the decoder to generate $\mathbf{y}$ autoregressively. Notice that the semantic embedding $\mathbf{z}_{sem}$ does not encode the position information and the syntactic embedding $\mathbf{z}_{syn}$ does not contain semantics. This forces the decoder to extract the semantics from $\mathbf{z}_{sem}$ and retrieve the syntactic structure from $\mathbf{z}_{syn}$. The attention weights attaching to $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$ make the decoder learn the association between the semantics and the syntax as well as the relation between the word order and the parse structures. Therefore, SynPG is able to reorganize the source sentence and use the given syntactic structure to rephrase the source sentence.

3.2.2 Unsupervised Training

Our design of the disentanglement makes it possible to train SynPG without using annotated pairs. We train SynPG with the objective to reconstruct the source sentences. More specifically, when training on a sentence $\mathbf{x}$, we first separate $\mathbf{x}$ into two parts: 1) an unordered word list $\bar{\mathbf{x}}$ and 2) its linearized parse $\mathbf{p}_x$ (can be obtained by a pretrained parser). Then, SynPG is trained to reconstruct $\mathbf{x}$ from $(\bar{\mathbf{x}}, \mathbf{p}_x)$ with the reconstruction loss

$$\mathcal{L} = -\sum_{i=1}^{n} \log P(y_i = x_i | \bar{\mathbf{x}}, \mathbf{p}_x, \mathbf{y}_1, ..., \mathbf{y}_{i-1}).$$

Notice that if we do not disentangle the semantics and the syntax, and directly use a seq2seq model to reconstruct $\mathbf{x}$ from $(\mathbf{x}, \mathbf{p}_x)$, it is likely that the seq2seq model only learns to copy $\mathbf{x}$ and ignores $\mathbf{p}_x$ since $\mathbf{x}$ contains all the necessary information for the reconstruction. Consequently, at inference time, no matter what target parse $\mathbf{p}$ is given, the seq2seq model always copies the whole source sentence $\mathbf{x}$ as the output (more discussion in Section 3.4).

On the contrary, SynPG learns the disentangled embeddings $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$. This makes SynPG capture the relation between the semantics and the syntax to reconstruct the source sentence $\mathbf{x}$. Therefore, at test time, given the source sentence $\mathbf{x}$ and a new target parse $\mathbf{p}$, SynPG is able to apply the learned relation to rephrase the

source sentence $\mathbf{x}$ according to the target parse $\mathbf{p}$.

Word dropout. We observe that the ground truth paraphrase may contain some words not appearing in the source sentence; however, the paraphrases generated by the vanilla SynPG tend to include only words appearing in the source sentence due to the reconstruction training objective. To encourage SynPG to improve the diversity of the word choices in the generated paraphrases, we randomly discard some words from the source sentence during training. More precisely, each word has a probability to be dropped out in each training iteration. Accordingly, SynPG has to predict the missing words during the reconstruction, and this enables SynPG to select different words from the source sentence to generate paraphrases. More details are discussed in Section 3.4.5.

3.2.3 Templates and Parse Generator

In the previous discussion, we assume that a full target constituency parse tree is provided as the input to SynPG. However, the full parse tree of the target paraphrase sentence is unlikely available at inference time. Therefore, following the setting in Iyyer et al. (2018), we consider generating the paraphrase based on the *template*, which is defined as the top two levels of the full constituency parse tree. For example, the template of `(S(NP(PRP))(VP(VBZ)(NP(NNS)))(.))` is `(S(NP)(VP))(.))`.

Motivated by Iyyer et al. (2018), we train a parse generator to generate full parses from templates. The proposed parse generator has the same architecture as SynPG, but the input and the output are different. The parse generator takes two inputs: a tag sequence $\mathbf{tag}_x$ and a target template $\mathbf{t}$. The tag sequence $\mathbf{tag}_x$ contains all the POS tags of the source sentence $\mathbf{x}$. For example, the tag sequence of the sentence "*He eats apples.*" is "`<PRP> <VBZ> <NNS> <.>`". Similar to the source sentence in SynPG, we do not consider the word order of the tag sequence during encoding. The expected output of the parse generator is a full parse $\tilde{\mathbf{p}}$ whose a syntactic structure follows the target template $\mathbf{t}$.

We train the parse generator without any additional annotations as well. Let $\mathbf{t}_x$ be the the template of $\mathbf{p}_x$ (the parse of $\mathbf{x}$), we end-to-end train the parse generator with the input being $(\mathbf{tag}_x, \mathbf{t}_x)$ and the output being $\mathbf{p}_x$.

Generating paraphrases from templates. The parse generator makes us generate paraphrases by providing target templates instead of target parses. The steps to generate a paraphrase given a source sentence $\mathbf{x}$ and a target template $\mathbf{t}$ are as follows:

1. Get the tag sequence $\mathbf{tag}_x$ of the source sentence $\mathbf{x}$.
2. Use the parse generator to generate a full parse $\tilde{\mathbf{p}}$ with input $(\mathbf{tag}_x, \mathbf{t})$.
3. Use SynPG to generate a paraphrase $\mathbf{y}$ with input $(\mathbf{x}, \tilde{\mathbf{p}})$.

Post-processing. We notice that certain templates are not suitable for some source sentences and therefore the generated paraphrases are nonsensical. We follow Iyyer et al. (2018) and use n-gram overlap and paraphrastic similarity computed by the model[2] from Wieting and Gimpel (2018) to remove nonsensical paraphrases[3].

3.3 Experimental Settings

We conduct extensive experiments to demonstrate that SynPG performs better syntactic control than other unsupervised paraphrase models, while the quality of generated paraphrases by SynPG is comparable to others. In addition, we show that the performance of SynPG is competitive or even better than supervised models when the training data is large enough.

3.3.1 Datasets

For the training data, we consider ParaNMT-50M (Wieting and Gimpel, 2018), a paraphrase dataset containing over 50 million pairs of reference sentences and the corresponding paraphrases as well as the quality scores. We select about 21 million pairs with higher quality scores as our training examples. Notice that we use *only the reference sentences* to train SynPG and unsupervised paraphrase models since we do not require paraphrase pairs.

We sample 6,400 pairs from ParaNMT-50M as the testing data. To evaluate the transferability of SynPG, we also consider the other three datasets: 1) Quora (Iyer

[2] https://github.com/jwieting/para-nmt-50m

[3] We set the minimum n-gram overlap to 0.3 and the minimum paraphrastic similarity to 0.7.

et al., 2017) contains over 400,000 paraphrase pairs and we sample 6,400 pairs from them. 2) PAN (Madnani et al., 2012) contains 5,000 paraphrase pairs. 3) MRPC (Dolan et al., 2004) contains 2,753 paraphrase pairs.

3.3.2 Evaluation

We consider paraphrase pairs to evaluate all the models. For each test paraphrase pair $(\mathbf{x}_1, \mathbf{x}_2)$, we consider $\mathbf{x}_1$ as the source sentence and treat $\mathbf{x}_2$ as the target sentence (ground truth). Let $\mathbf{p}_2$ be the parse of $\mathbf{x}_2$, given $(\mathbf{x}_1, \mathbf{p}_2)$, The model is expected to generate a paraphrase $\mathbf{y}$ that is similar to the target sentence $\mathbf{x}_2$.

We use BLEU score (Papineni et al., 2002) and human evaluation to measure the similarity between $\mathbf{x}_2$ and $\mathbf{y}$. Moreover, to evaluate how well the generated paraphrase $\mathbf{y}$ follows the target parse $\mathbf{p}_2$, we define the *template matching accuracy* (TMA) as follows. For each ground truth sentence $\mathbf{x}_2$ and the corresponding generated paraphrase $\mathbf{y}$, we get their parses ($\mathbf{p}_2$ and $\mathbf{p}_y$) and templates ($\mathbf{t}_2$ and $\mathbf{t}_y$). Then, we calculate the percentage of pairs whose $\mathbf{t}_y$ *exactly matches* $\mathbf{t}_2$ as the *template matching accuracy.*

3.3.3 Models for Comparison

We consider the following unsupervised paraphrase models: 1) **CopyInput**: a naïve baseline which directly copies the source sentence as the output without paraphrasing. 2) **BackTrans**: back-translation is proposed to generate paraphrases (Mallinson et al., 2017; Wieting and Gimpel, 2018; Hu et al., 2019). In our experiment, we use the pretrained EN-DE and DE-EN translation models[4] proposed by Ng et al. (2019) to conduct back-translation. Notice that training translation models requires additional translation pairs. Therefore, BackTrans needs more resources than ours and the translation data may not available for some low-resource languages. 3) **VAE**: we consider a vanilla variational autoencoder (Bowman et al., 2016) as a simple baseline. 4) **SIVAE**: syntax-infused variational autoencoder (Zhang et al., 2019a) utilizes additional syntax information to improve the quality of sentence generation and paraphrase generation. Unlike SynPG, SIVAE does not disentangle the semantics and syntax. 5) **Seq2seq-Syn**: we train a seq2seq model with Transformer architecture to

[4] https://github.com/pytorch/fairseq/tree/master/examples/wmt19

reconstruct $\mathbf{x}$ from $(\mathbf{x}, \mathbf{p}_x)$ without the disentanglement. We use this model to study the influence of the disentanglement. 6) **SynPG**: our proposed model which learns disentangled embeddings.

We also compare SynPG with supervised approaches. We consider the following: 1) **Seq2seq-Sup**: a seq2seq model with Transformer architecture trained on whole ParaNMT-50M pairs. 2) **SCPN**: syntactically controlled paraphrase network (Iyyer et al., 2018) is a supervised paraphrase model with syntactic control trained on ParaNMT-50M pairs. We use their pretrained model[5].

3.3.4 Implementation Details

We consider byte pair encoding (Sennrich et al., 2016) for tokenization and use Stanford CoreNLP parser (Manning et al., 2014) to get constituency parses. We set the max length of sentences to 40 and set the max length of linearized parses to 160 for all the models. For the encoders and the decoder of SynPG, we use the standard Transformer (Vaswani et al., 2017) with default parameters. The word embedding is initialized by GloVe (Pennington et al., 2014). We use Adam optimizer with the learning rate being 10^{-4} and the weight decay being 10^{-5}. We set the word dropout probability to 0.4 (more discussion in Section 3.4.5). The number of epoch for training is set to 5.

Seq2seq-Syn, Seq2seq-Sup are trained with the similar setting. We reimplemnt VAE and SIVAE, and all the parameters are set to the default value in the original papers.

3.4 Results and Discussion

3.4.1 Syntactic Control

We first discuss if the syntactic specification enables SynPG to control the output syntax better. Table 3.1 shows the template matching accuracy and BLEU score for SynPG and the unsupervised baselines. Notice that here we use the full parse trees

[5] https://github.com/miyyer/scpn

	Model	ParaNMT		Quora		PAN		MRPC	
		TMA	BLEU	TMA	BLEU	TMA	BLEU	TMA	BLEU
No Paraphrasing	CopyInput	33.6	16.4	55.0	20.0	37.3	26.8	47.9	30.7
Unsupervised Models	BackTrans	29.0	16.3	53.0	16.4	27.9	16.2	47.2	21.6
	VAE	26.3	9.6	44.0	8.1	19.4	5.2	20.8	1.2
With Syntactic Specifications	SIVAE	30.0	12.8	48.3	13.1	26.6	11.8	21.5	5.1
	Seq2seq-Syn	33.5	16.3	54.9	19.8	37.1	**26.5**	47.7	**30.4**
	SynPG	**71.0**	**32.2**	**82.6**	**33.2**	**66.3**	26.4	**74.0**	26.2

Table 3.1: Paraphrase results on four datasets. TMA denotes the template matching accuracy, which evaluates how often the generated paraphrases follow the target parses. With the syntactic control, SynPG obtains higher BLEU score and the template matching accuracy. This implies the paraphrases generated by SynPG are more similar to the ground truths and follow the target parses more accurately.

as the syntactic specifications. We will discuss the influence of using the template as the syntactic specifications in Section 3.4.3.

Although we train SynPG on the reference sentences of ParaNMT-50M, we observe that SynPG performs well on Quora, PAN, and MRPC as well. This validates that SynPG indeed learns the syntactic rules and can transfer the learned knowledge to other datasets. CopyInput gets high BLEU scores; however, due to the lack of paraphrasing, it obtains low template matching scores. Compared to the unsupervised baselines, SynPG achieves higher template matching accuracy and higher BLEU scores on all datasets. This verifies that the syntactic specification is indeed helpful for syntactic control.

Next, we compare SynPG with Seq2seq-Syn and SIVAE. All models are given syntactic specifications; however, without the disentanglement, Seq2seq-Syn and SIVAE tend to copy the source sentence as the output and therefore get low template matching scores.

Table 3.2 lists some paraphrase examples generated by all models. Again, we observe that without syntactic specifications, the paraphrases generated by unsupervised baselines are similar to the source sentences. Without the disentanglement, Seq2seq-Syn and SIVAE always copy the source sentences. SynPG is the only model can generate paraphrases syntactically similar to the ground truths.

Model	Example 1 (ParaNMT)	Example 2 (Quora)
Source Sent.	these children are gonna die if we don't act now.	what are the best ways to improve writing skills?
Ground Truth	if we don't act quickly, the children will die.	how could i improve my writing skill?
BackTrans	these children will die if we do not act now.	what are the best ways to improve your writing skills?
VAE	these children are gonna die if we don't act now.	what are the best ways to improve writing skills?
SIVAE	these children are gonna die if we don't act now .	what are the best ways to improve writing skills?
Seq2seq-Syn	these children are gonna die if we don't act now.	what are the best ways to improve writing skills?
SynPG	if we don't act now, these children will die.	how can i improve my writing skills?

Table 3.2: Paraphrases generated by each model. SynPG can generate paraphrases with the syntax more similar to the ground truth than other baselines.

Model	**2**	**1**	**0**	**2+1**	Hit Rate
BackTrans	63.6	22.4	14.0	86.0	11.0
SIVAE	57.6	20.3	22.0	78.0	6.5
SynPG	44.3	32.0	23.7	76.3	28.9

Table 3.3: Human evaluation on a three-point scale (**0** = not a paraphrase, **1** = ungrammatical paraphrase, **2** = grammatical paraphrase). SynPG performs better on hit rate (defined as the percentage of generated paraphrase getting **2** and matching the target parse at the same time) than other unsupervised models.

3.4.2 Human Evaluation

We perform human evaluation using Amazon Mechanical Turk to evaluate the quality of generated paraphrases. We follow the setting of previous work (Kok and Brockett, 2010; Iyyer et al., 2018; Goyal and Durrett, 2020). For each model, we randomly select 100 pairs of source sentence **x** and the corresponding generated paraphrase **y** from ParaNMT-50M test set (after being post-processed as mentioned in Section 3.2.3) and have three Turkers annotate each pair. The annotations are on a three-point scale: **0** means **y** is not a paraphrase of **x**; **1** means **x** is paraphrased into **y** but **y** contains some grammatical errors; **2** means **x** is paraphrased into **y**, which is grammatically correct.

Model	Template Matching Accuracy			
	ParaNMT	Quora	PAN	MRPC
Paraphrases generated by target parses	71.0	82.6	66.3	74.0
Paraphrases generated by target templates	54.1	73.4	51.7	62.3
Parses $\tilde{\mathbf{p}}$ generated by parse generator	98.4	99.0	95.7	93.9

Table 3.4: Influence of using templates. Using templates proves more effortless during the generation process, but may compromise the syntactic control ability.

The results of human evaluation are reported in Table 3.3. If paraphrases rated **1** or **2** are considered meaningful, we notice that SynPG generates meaningful paraphrases at a similar frequency to that of SIVAE. However, SynPG tends to generate more ungrammatical paraphrases (those rated **1**). We think the reason is that most of paraphrases generated by SIVAE are very similar to the source sentences, which are usually grammatically correct. On the other hand, SynPG is encouraged to use different syntactic structures from the source sentences to generate paraphrases, which may lead some grammatical errors.

Furthermore, we calculate the *hit rate*, the percentage of generated paraphrases getting **2** and matching the target parse at the same time. The hit rate measures how often the generated paraphrases follow the target parses and preserve the semantics (verified by human evaluation) simultaneously. The results show that SynPG gets higher hit rate than other models.

3.4.3 Target Parses vs. Target Templates

Next, we discuss the influence of generating paraphrase by using templates instead of using full parse trees. For each paraphrase pair $(\mathbf{x}_1, \mathbf{x}_2)$ in test data, we consider two ways to generate the paraphrase. 1) Generating the paraphrase with the target parse. We use SynPG to generate a paraphrase directly from $(\mathbf{x}_1, \mathbf{p}_2)$. 2) Generating the paraphrase with the target template. We first use the parse generator to generate a parse $\tilde{\mathbf{p}}$ from $(\mathbf{tag}_1, \mathbf{t}_2)$, where $\mathbf{tag}_1$ is the tag sequence of $\mathbf{x}_1$ and $\mathbf{t}_2$ is the template of $\mathbf{p}_2$. Then we use SynPG to generate a paraphrase from $(\mathbf{x}_1, \tilde{\mathbf{p}})$. We calculate the template matching accuracy to compare these two ways to generate paraphrases, as

Template	Generated Paraphrase
Original	can you adjust the cameras?
(S(NP)(VP)(.))	you can adjust the cameras.
(SBARQ(ADVP)(,)(S)(,)(SQ)(.))	well, adjust the cameras , can you?
(S(PP)(,)(NP)(VP)(.))	on the cameras, you can adjust them?
Original	she doesn't keep pictures from her childhood.
(SBARQ(WHADVP)(SQ)(.))	why doesn't she keep her pictures from childhood.
(S(‘‘)(NP)(VP)(’’)(NP)(VP)(.))	“ she doesn't keep pictures from her childhood ” she said.
(S(ADVP)(NP)(VP)(.))	perhaps she doesn't keep pictures from her childhood.

Table 3.5: Paraphrases generated by SynPG with different templates.

shown in Table 3.4. We also report the template matching accuracy of the generated parse $\tilde{\mathbf{p}}$.

We find that most of generated parses $\tilde{\mathbf{p}}$ indeed follow the target templates, which means that the parse generator usually generates good parses $\tilde{\mathbf{p}}$. Next, we observe that generating paraphrases with target parses usually performs better than with target templates. The results show a trade-off. Using templates proves more effortless during the generation process, but may compromise the syntactic control ability. In comparison, by using the target parses, we have to provide more detailed parses, but our model can control the syntax better.

Another benefit of generating paraphrase with target templates is that we can easily generate a lot of syntactically different paraphrases by feeding the model with different templates. Table 3.5 lists some paraphrases generated by SynPG with different templates. We can perceive that most generated paraphrases are grammatically correct and have similar meanings to the original sentence.

3.4.4 Training SynPG on Larger Dataset

Finally, we demonstrate that the performance of SynPG can be further improved and be even competitive to supervised models on some datasets if we consider more training data. The advantage of unsupervised paraphrase models is that we do not require parallel pairs for training. Therefore, we can easily boost the performance of SynPG by consider more unannotated texts into training.

We consider SynPG-Large, the SynPG model trained on the reference sentences of ParaNMT-50M as well as One Billion Word Benchmark (Chelba et al., 2014), a

	Model	ParaNMT		Quora		PAN		MRPC	
		TMA	BLEU	TMA	BLEU	TMA	BLEU	TMA	BLEU
Ours	SynPG	71.0	32.2	82.6	33.2	66.3	26.4	74.0	26.2
	SynPG-Large	70.3	31.8	83.8	34.7	66.6	27.1	79.3	36.2
	SynPG-FT	–	–	86.3	**44.4**	66.4	34.2	**80.7**	**44.6**
Supervised Models	Seq2seq-Sup	40.2	19.6	54.0	11.3	29.2	13.1	44.3	16.3
	SCPN	**83.9**	**58.3**	**87.1**	41.0	**72.3**	**37.6**	80.1	41.8

Table 3.6: Training on larger dataset improves the performance of SynPG. Since training SynPG does not require annotated paraphrase pairs, it is possible to fine-tune SynPG on the texts in the target domain. With the fine-tuning, SynPG can have competitive or even better performance than supervised approaches.

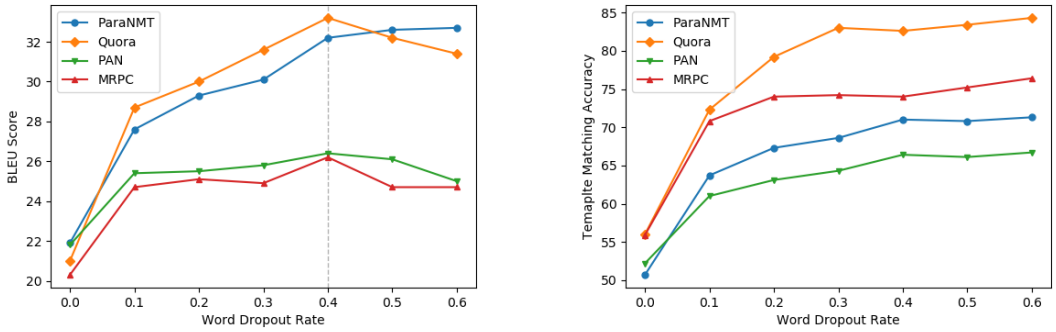
large corpus for training language models. We sample about 24 million sentences from One Billion Word and add them to the training set. In addition, we fine-tune SynPG-Large on only the reference sentences of the testing paraphrase pairs, called SynPG-FT.

From Table 3.6, We observe that enlarging the training data set indeed improves the performance. Also, with the fine-tuning, the performance of SynPG can be much improved and even is better than the performance of supervised models on some datasets. The results demonstrate the potential of unsupervised paraphrase generation with syntactic control.

3.4.5 Word Dropout Rate

The word dropout rate plays an important role for SynPG since it controls the ability of SynPG to generate new words in paraphrases. We test different word dropout rates and report the BLEU scores and the template matching accuracy in Figure 3.3.

From Figure 3.3a, we can observe that setting the word dropout rate to 0.4 can achieve the best BLEU score in most of datasets. The only exception is ParaNMT, which is the dataset used for training. On the other hand, Figure 3.3b shows that higher word dropout rate leads to better template matching accuracy. The reason is that higher word dropout rate gives SynPG more flexibility to generate paraphrases. Therefore, the generated paraphrases can match the target syntactic specifications better. However, higher word dropout rate also make SynPG have less ability to



(a) BLEU score **(b)** Template matching accuracy

Figure 3.3: Influence of word drop out rate. Setting the word dropout rate to 0.4 can achieve the best BLEU score. However, higher word dropout rate leads to better template matching accuracy.

preserve the meaning of source sentences. Considering all the factors above, we recommend to set the word dropout rate to 0.4 for SynPG.

3.5 Improving Robustness of Models

Recently, a lot of work show that NLP models can be fooled by different types of adversarial attacks (Alzantot et al., 2018; Ebrahimi et al., 2018; Iyyer et al., 2018; Tan et al., 2020; Jin et al., 2020). Those attacks generate *adversarial examples* by slightly modifying the original sentences without changing the meanings, while the NLP models change the predictions on those examples. However, a robust model is expected to output the same labels. Therefore, how to make NLP models not affected by the adversarial examples becomes an important task.

Since SynPG is able to generate syntactically different paraphrases, we can improve the robustness of NLP models by data augmentation. The models trained with data augmentation are thus more robust to the syntactically adversarial examples (Iyyer et al., 2018), which are the adversarial sentences that are paraphrases to the original sentences but with syntactic difference.

We conduct experiments on three classification tasks covered by GLUE benchmark (Wang et al., 2019a): SST-2, MRPC, and RTE. For each training example, we use SynPG to generate four syntactically different paraphrases and add them to the

Model	SST-2		MRPC		RTE	
	Acc.	Brok.	Acc.	Brok.	Acc.	Brok.
Base	91.9	46.7	84.1	52.8	63.2	58.3
SynPG	88.9	**39.6**	80.1	**35.5**	60.7	**33.9**

Table 3.7: Data augmentation improves the robustness of models. SynPG denotes the base classifier trained on augmented data generated by SynPG. Acc denotes the accuracy in the original dataset (the higher is the better). Brok denotes the percentage of examples changing predictions after attacking (the lower is the better).

training set. We consider the setting to generate syntactically adversarial examples by SCPN (Iyyer et al., 2018). For each testing example, we generate five candidates of adversarial examples. If the classifier gives at least one wrong prediction on the candidates, we treat the attack to be successful.

We compare the model without data augmentation (Base) and with data augmentation (SynPG) in Table 3.7. We observe that with the data augmentation, the accuracy before attacking is slightly worse than Base. However, after attacking, the percentage of examples changing predictions is much less than Base, which implies that data augmentation indeed improves the robustness of models.

3.6 Related Work

Paraphrase generation. Traditional approaches usually require hand-crafted rules, such as rule-based methods (McKeown, 1983), thesaurus-based methods (Bolshakov and Gelbukh, 2004; Kauchak and Barzilay, 2006), and lattice matching methods (Barzilay and Lee, 2003). However, the diversity of their generated paraphrases is usually limited.

Recently, neural models make success on paraphrase generation (Prakash et al., 2016; Mallinson et al., 2017; Cao et al., 2017; Egonmwan and Chali, 2019; Li et al., 2019; Gupta et al., 2018). These approaches treat paraphrase generation as a translation task and design seq2seq models based on a large amount of parallel data. To reduce the effort to collect parallel data, unsupervised paraphrase generation has attracted attention in recent years. Wieting et al. (2017); Wieting and Gimpel (2018) use translation models to generate paraphrases via back-translation. Zhang et al.

(2019a); Roy and Grangier (2019) generate paraphrases based on variational autoencoders. Reinforcement learning techniques are also considered for paraphrase generation (Li et al., 2018).

Controlled generation. Recent work on controlled generation can be grouped into two families. The first family is doing end-to-end training with an additional trigger to control the attributes, such as sentiment (Shen et al., 2017; Hu et al., 2017; Fu et al., 2018; Dai et al., 2019), tense (Logeswaran et al., 2018), plots (Ammanabrolu et al., 2020; Fan et al., 2019; Tambwekar et al., 2019; Yao et al., 2019; Goldfarb-Tarrant et al., 2019, 2020), societal bias (Wallace et al., 2019; Sheng et al., 2020, 2021), and syntax (Iyyer et al., 2018; Goyal and Durrett, 2020; Kumar et al., 2020). The second family controls the attributes by learning disentangled representations. For example, Romanov et al. (2019) disentangle the meaning and the form of a sentence. Chen et al. (2019a,b); Bao et al. (2019) disentangle the semantics and the syntax of a sentence.

3.7 Summary

We present syntactically controlled paraphrase generator (SynPG), an paraphrase model that can control the syntax of generated paraphrases based on the given syntactic specifications. SynPG is designed to disentangle the semantics and the syntax of sentences. The disentanglement enables SynPG to be trained without the need for annotated paraphrase pairs. Extensive experiments show that SynPG performs better syntactic control than unsupervised baselines, while the quality of the generated paraphrases is competitive to supervised approaches. Finally, we demonstrate that SynPG can improve the robustness of NLP models by generating additional training examples. SynPG is especially helpful for the domain where annotated paraphrases are hard to obtain but a large amount of unannotated text is available. One limitation of SynPG is the need for mannually providing target syntactic templates at inference time. We leave the automatic template generation as our future work.

CHAPTER 4

Improving Syntax-Level Robustness with Abstract Meaning Representations

4.1 Introduction

Syntactically controlled paraphrase generation approaches aim to control the format of generated paraphrases by taking into account additional parse specifications as the inputs, as illustrated by Figure 4.1. It has attracted increasing attention in recent years since it can diversify the generated paraphrases and benefit a wide range of NLP applications (Iyyer et al., 2018; Huang and Chang, 2021; Sun et al., 2021), including task-oriented dialog generation (Gao et al., 2020), creative generation (Tian et al., 2021), and model robustness (Huang and Chang, 2021).

Recent works have shown success in training syntactically controlled paraphrase generators (Iyyer et al., 2018; Chen et al., 2019b; Kumar et al., 2020; Sun et al., 2021). Although their models can generate high-quality paraphrases and achieve good syntactic control ability, the training process needs a large amount of supervised data, e.g., parallel paraphrase pairs. Annotating paraphrase pairs is usually expensive because it requires intensive domain knowledge and high-level semantic understanding. Due to the difficulty in collecting parallel data, the ability of supervised approaches are limited, especially when adapting to new domains.

To reduce the annotation demand, unsupervised approaches can train syntactically controlled paraphrase generators without the need for parallel pairs (Zhang et al., 2019a; Bao et al., 2019; Huang and Chang, 2021). Most of them achieve syntactic control by learning disentangled embeddings for semantics and syntax separately (Bao et al., 2019; Huang and Chang, 2021). However, without parallel data, it is

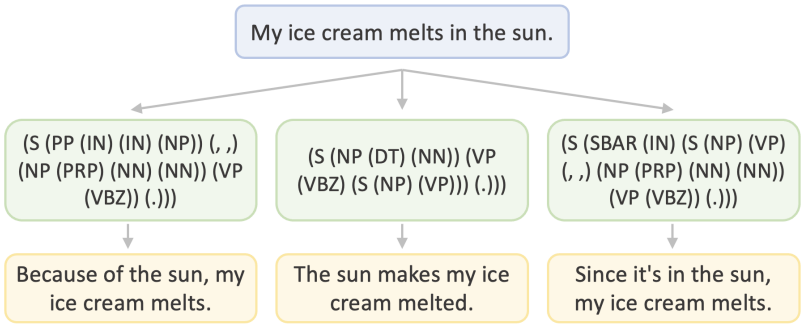


Figure 4.1: An illustration of syntactically controlled paraphrase generation. Given a source sentence and different parse specifications, the model generates different paraphrases following the parse specifications.

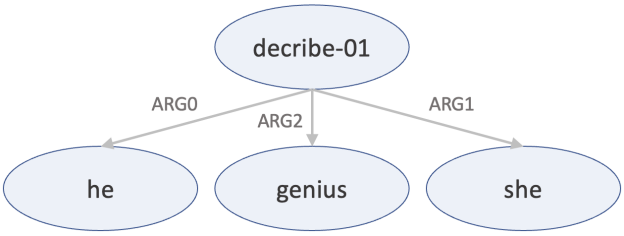


Figure 4.2: The same AMR graph for a pair of paraphrased sentences "He described her as a genius." and "She was a genius, according to his description."

challenging to learn a good disentanglement and capture semantics well. As we will show later (Section 4.4.1), unsupervised approaches can generate bad paraphrases by mistakenly swapping object and subject of a sentence.

In this work, we propose to use Abstract Meaning Representations (AMR) (Banarescu et al., 2013) to learn better disentangled semantic embeddings for unsupervised syntactically controlled paraphrase generation. AMR is a semantic graph structure that covers the abstract meaning of a sentence. As shown in Figure 4.2, two sentences would have the same (or similar) AMR graph as long as they carry the same abstract meaning, even they are expressed with different syntactic structures. This property makes AMRs a good resource to capture sentence semantics.

Based on this, we design an **AMR**-enhanced **P**araphrase **G**enerator (AMRPG), which separately learns (1) *semantic embeddings* with the AMR garphs extracted from the input sentence and (2) *syntactic embeddings* from the constituency parse of the input sentence. Then, AMRPG trains a decoder to reconstruct the input sentence from the semantic and syntactic embeddings. The reconstruction objective and the design

of the disentanglement of semantics and the syntax makes AMRPG learn to generate syntactically controlled paraphrases without using parallel pairs. Our experiments show that AMRPG performs better syntactic control than existing unsupervised approaches. Additionally, we demonstrate that the generated paraphrases of AMRPG can be used for data augmentation to improve the robustness of NLP models.

4.2 Related Work

Paraphrase generation. Traditional paraphrase generators are usually based on hand-crafted rules (Barzilay and Lee, 2003) or seq2seq models (Cao et al., 2017; Gupta et al., 2018; Fu et al., 2019). To generate diverse paraphrases, different techniques are proposed, including random pattern embeddings (Kumar et al., 2019), latent space perturbation (Roy and Grangier, 2019; Zhang et al., 2019a; Cao and Wan, 2020), multi-round generation (Lin and Wan, 2021), reinforcement learning (Liu et al., 2020a), prompt-tuning (Chowdhury et al., 2022), order control (Goyal and Durrett, 2020), and syntactic control (Iyyer et al., 2018; Kumar et al., 2020; Huang and Chang, 2021; Sun et al., 2021).

Abstract meaning representation (AMR). Since AMR (Banarescu et al., 2013) captures high-level semantics, it has been applied for various NLP tasks, including summarization (Sachan and Xing, 2016), dialogue modeling (Bai et al., 2021), information extraction (Zhang et al., 2021). Some works also focus on training high-quality AMR parsers with graph encoders (Cai and Lam, 2020), seq2seq models (Konstas et al., 2017; Zhou et al., 2020), and decoder-only models (Bevilacqua et al., 2021).

4.3 Unsupervised Syntactically Controlled Paraphrase Generation

4.3.1 Problem Formulation

We follow previous works (Iyyer et al., 2018; Huang and Chang, 2021) and consider constituency parses (without terminals) as the control signals. Given a source

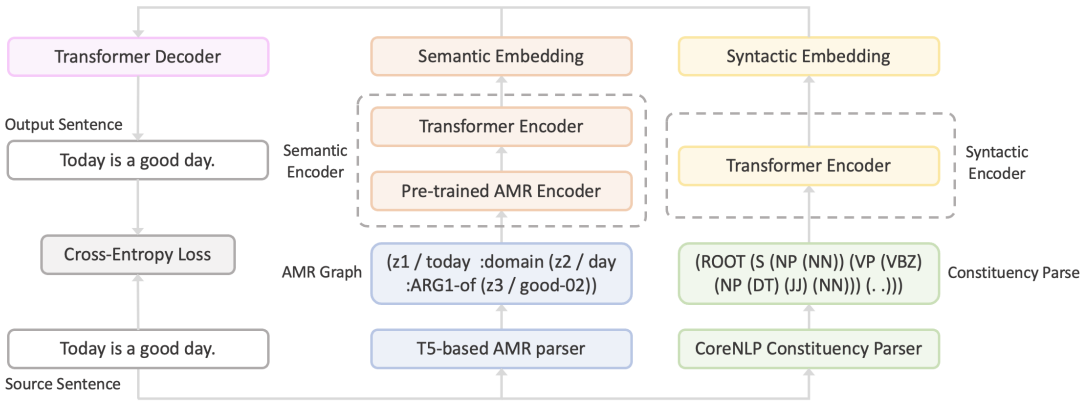


Figure 4.3: AMRPG's framwork. It separately encodes the AMR graph and the constituency parse of the input sentence into two disentangled semantic and syntactic embeddings. A decoder is then learned to reconstruct the input sentence from the semantic and syntactic embeddings.

sentence s and a target parse p, the goal of the syntactically controlled paraphrase generator is to generate a target sentence t which has similar semantics to the source sentence s and has syntax following the parse p. In the unsupervised setting, the paraphrase generator cannot access any target sentences and target parses but only the source sentences and source parses during training.

4.3.2 Proposed Method: AMRPG

Motivated by previous approaches (Bao et al., 2019; Huang and Chang, 2021), we design AMRPG to learn separate embeddings for semantics and syntax, as illustrated by Figure 4.3. Then, AMRPG learns a decoder with the objective to reconstruct the source sentence. The challenge here is how to learn embeddings such that the semantic embedding contains only semantic information while the syntactic embedding contains only syntactic information. We introduce the details as follows.

Semantic embedding. Given a source sentence, we first use a pre-trained AMR parser[1] to get its AMR graph. Next, we use a semantic encoder to encode the AMR graph into the semantic embedding e_{sem}. Specifically, the semantic encoder consists of two parts: a fixed pre-trained AMR encoder (Ribeiro et al., 2021) followed by a

[1] https://github.com/bjascob/amrlib-models

learnable Transformer encoder. We additionally perform *node masking* when training the semantic encoder. Specifically, every node in the AMR graph has a probability to be masked out during training. This can improve the robustness of AMRPG.

As mentioned above, two semantically similar sentences would have similar AMR graphs regardless of their syntax. This property encourages AMRPG to capture only semantic information in semantic embeddings. Compared with previous work (Huang and Chang, 2021), which uses bag-of-words to learn the semantic embeddings, using AMR can capture semantics better and lead to better performance, as shown in Section 4.4.

Syntactic embedding. Given a source sentence, we use the Stanford CoreNLP toolkit (Manning et al., 2014) to get its constituency parse. Then, we remove all the terminals in the parse and learns a Transformer encoder to encode the parse into the syntactic embedding e_{syn}. Since we remove the terminals, the syntactic embedding contains only the syntactic information of the source sentence.

Decoder. We train a Transformer decoder that takes the semantic embedding e_{sem} and the syntactic embedding e_{syn} as the input, and reconstructs the source sentence with a cross-entropy loss. The reconstruction objective makes AMRPG not require parallel paraphrase pairs for training.

Inference. Given a source sentence s and a target parse p, we use the semantic encoder to encode the AMR graph of s into the semantic embedding, use the syntactic encoder to encode p into the syntactic embedding, and use the decoder to generate the target sentence t.

4.4 Experiments

4.4.1 Syntactically Controlled Paraphrase Generation

Datasets. We consider ParaNMT (Wieting and Gimpel, 2018) for training and testing. We use *only the source sentences* in ParaNMT to train AMRPG and other unsupervised baselines, and use both the source sentences and target sentences to

train supervised baselines. To further test the model's ability to generalize to new domains, we directly use the models trained with ParaNMT to test on Quora (Iyer et al., 2017), MRPC (Dolan et al., 2004), and PAN (Madnani et al., 2012). Following previous work (Huang and Chang, 2021), our test data is: (1) 6,400 examples of ParaNMT, (2) 6,400 examples of Quora, (3) 2,048 examples of PAN, and (4) 1,920 examples of MRPC.

Evaluation metrics. Following previous work Huang and Chang (2021), we consider paraphrase pairs to evaluate the performance. Given a paraphrase pairs (s_1, s_2), we use the Standford CoreNLP constituency parser Manning et al. (2014) to get their parses (p_1, p_2). The input of all baselines would be (s_1, p_2) and the ground truth would be s_2.

Assuming the generated paraphrase is g, We use BLEU score to measure the similarity between the generated paraphrase g and the ground truth s_2. We also calculate the template matching accuracy (TMA) by computing the exact matching accuracy of the top-2 levels of p_g and p_2 (p_g is the constituency parse of g).

Baselines. We consider the following unsupervised models: SIVAE (Zhang et al., 2019a), SynPG (Huang and Chang, 2021), AMRPG, and T5-Baseline, which replaces the AMR encoder with a T5-encoder. We also consider SCPN (Iyyer et al., 2018) as the supervised baseline.

Implementation details. We use around 20 millions of examples[2] in ParaNMT (Wieting and Gimpel, 2018) to train AMRPG and all baselines. The semantic encoder and the syntactic decoder are trained from scratch, with the default architecture and the default parameters of `torch.nn.Transformer`. The max length for input sentences, the linearized constituency parses, and the linearized AMR graph are set to 40, 160, and 250, respectively. The word dropout rate is 0.4 while the node masking rate is 0.6. We consider Adam optimizer with the learning rate being 10^{-4} and the weight decay being 10^{-5}. The total number of epochs is set to 10. When generating the outputs, we use random sampling with temperature being 0.5. The model is

[2] https://github.com/uclanlp/synpg

Model	ParaNMT		Quora		PAN		MRPC	
	TMA	BLEU	TMA	BLEU	TMA	BLEU	TMA	BLEU
Unsupervised Approaches (without using parallel pairs)								
SIVAE (Zhang et al., 2019a)	30.0	12.8	48.3	13.1	26.6	11.8	21.5	5.1
SynPG (Huang and Chang, 2021)	71.0	32.2	82.6	33.2	**66.3**	26.4	**74.0**	26.2
T5-Baseline	57.1	22.8	66.1	22.2	55.3	21.0	66.2	18.8
AMRPG	**74.3**	**39.1**	**84.8**	**33.9**	65.6	**31.0**	71.9	**34.8**
Unsupervised Approaches (using target domain source sentences)								
SynPG (Huang and Chang, 2021)	-	-	86.3	44.4	66.4	34.2	**80.7**	44.6
AMRPG	-	-	**86.5**	**45.4**	**67.5**	**37.6**	76.8	**45.9**
Supervised Approaches (using additional parallel pairs in ParaNMT; not compariable to ours)								
SCPN (Iyyer et al., 2018)	83.9	58.3	87.1	41.0	72.3	37.6	80.1	41.8

Table 4.1: Results of syntactically controlled paraphrase generation. AMRPG performs the best among all unsupervised approaches and can outperform supervised approaches when considering the target domain source sentences.

trained with 4 NVIDIA V100 GPUs with 16 GB memory each. It takes around 7 days to finish the training process.

Results. Table 4.1 shows the results of syntactically controlled paraphrase generation. AMRPG performs the best among the unsupervised approaches. Specifically, AMRPG outperforms SynPG, the state-of-the-art unsupervised model, with a large gap in terms of BLEU score. This justifies that using AMR can learn better disentangled embeddings and capture semantics better.

We observe that there is indeed a performance gap between AMRPG and SCPN (supervised baseline). However, since AMRPG is an unsupervised model, it is possible to use the source sentences from the target domains to further fine-tune AMRPG without additional annotation cost. As shown in the table, AMRPG with further fine-tuning can achieve even better performance than SCPN when considering domain adaptation (Quora, MRPC, and PAN). This demonstrates the flexibility and the potential of unsupervised paraphrase models.

Qualitative examples. Table 4.2 lists some paraphrases generated by SynPG and AMRPG. As we mentioned in Section 4.3, SynPG uses bag-of-words to learn semantic embeddings and therefore SynPG is easy to get confused about the relations between entities or mistake the subject for the object. In contrast, AMRPG can preserve more

Input	The dog chased the cat on the street.
Parse template	`(S(NP(DT)(NN))(VP(VBN)(PP))(.))`
Target	The cat was chased by the dog on the street.
SynPG	The dog was chased by the cat on the street.
AMRPG	The cat was chased by a dog in the street.
Input	John will send a gift to Tom when Christmas comes.
Parse template	`(S(SBAR (WHADVP)(S))(,)(NP(NNP))(VP(MD)(VP))(.))`
Target	When Christmas comes, John will send a gift to Tom.
SynPG	When Tom comes, John will send a gift to Christmas.
AMRPG	When Christmas comes, John will send a gift to Tom.

Table 4.2: Paraphrase examples generated by SynPG and AMRPG. AMRPG captures semantics better and generates higher quality of paraphrases than SynPG.

semantics.

4.4.2 Improving Robustness of NLP Models

We demonstrate that the paraphrases generated by AMRPG can improve the robustness of NLP models by data augmentation. Following the setting of previous work (Huang and Chang, 2021), we consider three classification tasks in GLUE (Wang et al., 2019a): MRPC, RTE, and SST-2. We compare three baselines: (1) the classifier trained with original training data, (2) the classifier trained with original training data and augmented data generated by SynPG, and (3) the classifier trained with original training data and augmented data generated by AMRPG. Specifically, for every instance in the original training data, we generate four paraphrases as the augmented examples by considering four common syntactic templates.

Training Details. We use the pre-trained SynPG parse generator to generate the full parse for each instance with the following parse templates: "`(S(NP)(VP)(.))`", "`(S(VP)(.))`", "`(NP(NP)(.))`", and "`(FRAG(SBAR)(.))`". Then, we use the generated full parses as the parse specifications to generate paraphrases for data augmentation. When training classifiers with data augmentation, the original instances have four times of weights as the augmented instances when computing the loss. We use the scripts from Huggingface[3] with default values to train the classifiers.

[3] https://github.com/huggingface/transformers/blob/main/examples/pytorch/text-classification/run_glue.py

Model	MRPC		RTE		SST-2	
	Acc.	Brok.	Acc.	Brok.	Acc.	Brok.
Base	**83.3**	52.9	**62.1**	58.1	**92.2**	38.8
+ SynPG	80.6	42.2	61.7	40.3	91.5	38.5
+ AMRPG	80.6	**38.3**	58.8	**39.3**	91.6	**36.7**

Table 4.3: Augmenting paraphrases generated by AMRPG improves the robustness of NLP models. Acc denotes the clean accuracy (the higher is the better). Brok denotes the percentage of examples being successfully attacked (the lower is the better).

Generating Adversarial Examples. We use the official script[4] of SCPN (Iyyer et al., 2018) to generate syntactically adversarial examples. Specifically, we consider the first five parse templates for RTE and SST-2 and first three parse templates for MRPC to generate the adversarial examples. As long as one of the adversarial examples makes the classifier change the prediction, we count it as a successful attack on this instance.

Results. Table 4.3 shows the clean accuracy and the broken rate (the percentage of examples being attacked) after attacked by the syntactically adversarial examples generated with SCPN (Iyyer et al., 2018). Although the classifiers trained with data augmentation have slightly worse clean accuracy, they have significantly lower broken rates, which implies that data augmentation improves the model robustness. Also, data augmentation with AMRPG performs better than data augmentation with SynPG in terms of the broken rate. We attribute this to the better quality of paraphrase generation of AMRPG.

4.5 Summary

We propose AMRPG that utilizes AMR to learn a better disentanglement of semantics and syntax without using any parallel data. This enables AMRPG to captures semantics better and generate more accurate syntactically controlled paraphrases than existing unsupervised approaches. We also demonstrate that how to apply AMRPG to improve the robustness of NLP models.

[4] https://github.com/miyyer/scpn

Part II

Language-Level Robustness

In this part, we focus on language-level robustness. A robust NLP model should give us consistent behaviors regardless of the language of the input text, as long as the underlying meaning of the input text remains unchanged. For example, a robust NLP model should make the same prediction for the English text *"This restaurant is very good."* and its translation in French *"Ce resto est très bon."*

In the field of NLP, there is a setting known as *zero-shot cross-lingual transfer* (Hu et al., 2020; Liang et al., 2020), which studies the behaviors of models across various languages. In this setting, we train NLP models with instances in *source languages* and test the models with instances in *target languages*. The challenge is to handle the discrepancy between languages and transfer knowledge from source languages to target languages. Improving the zero-shot cross-lingual transfer performance of models usually implies improving the language-level robustness of NLP models. Therefore, in the following chapters, we put our attention on zero-shot cross-lingual transfer. In Chapter 5 and 6, we introduce two designs to improve zero-shot cross-lingual transfer performance: robust training and generation-based models.

CHAPTER 5

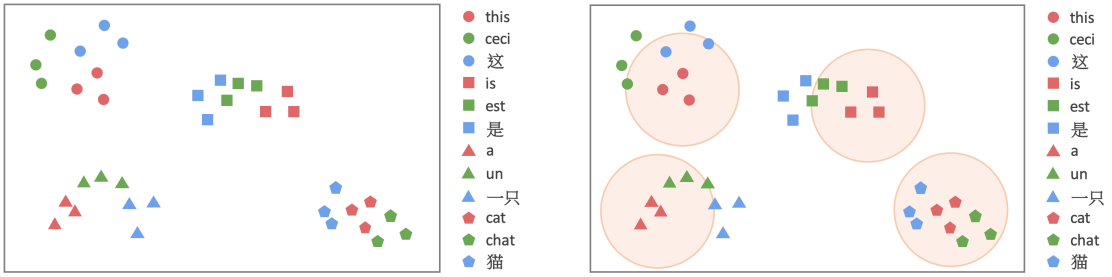
Improving Language-Level Robustness with Robust Training

5.1 Introduction

Zero-shot cross-lingual transfer learning aims to learn models with data available in one or more source languages and use them in other target languages for which there is no data (zero-resource) available. The zero-shot cross-lingual transfer has a great practical value for low-resource languages since it reduces the requirement of labeled data to learn models for downstream tasks, e.g., text classification (Conneau et al., 2018b; Yang et al., 2019) and question answering (Lewis et al., 2020b).

Recently, pre-trained multilingual language encoders, such as multilingual BERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020a), demonstrate promising performance on zero-shot cross-lingual transfer learning for a wide range of downstream tasks (Hu et al., 2020; Liang et al., 2020). These language encoders learn a shared multilingual contextual embedding space; they are able to represent word pairs in parallel sentences with similar contextual representations. However, the multilingual encoders fail to capture this similarity when the source and target languages are less similar at levels of morphology, syntax, and semantics (Ahmad et al., 2019a,b).

Prior 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 for zero-shot cross-lingual transfer learning. As illustrated in Figure 5.1a, words with similar meanings (e.g. *this*, *ceci*, and 这) have similar representations in the contextual multilingual embedding space, even though these words are in different languages. This alignment helps models transfer



(a) Contextual representations of different words.

(b) Robust regions try to cover neighbor embeddings.

Figure 5.1: An illustration of different words in the multilingual contextual embedding space. (a) Words with similar meanings in different languages have similar representations but they are not exactly aligned. (b) We aim to learn a robust classifier whose robust regions (orange circles) that cover as many neighbor words as possible.

the learned knowledge from source languages to target languages. Therefore, several works focus on improving the quality of alignments in the multilingual embedding space (Cao et al., 2020; Chi et al., 2021; Pan et al., 2021; Dou and Neubig, 2021). Nevertheless, learning such alignments usually requires sentence-level or word-level parallel corpora, which are expensive to be obtained for low-resource languages. In addition, because the meanings of words in different languages are usually not exactly matched, learn a perfect alignment could be impossible.

In this work, we start from another point of view to improve zero-shot cross-lingual transfer performance. We aim to make the multilingual encoders robust such that they can tolerate a certain amount of noise in the input embeddings. More specifically, as shown in Figure 5.1b, we target to construct *robust regions* (orange circles) for embeddings in the multilingual embedding space. During training, the robust model is expected to output similar predictions for embeddings in the same robust region. Therefore, as long as similar words in different languages fall into the same robust region, even if they are not perfectly aligned, the model can still have similar predictions for them.

To learn the robust model, we first draw connections between adversarial examples (Li et al., 2020b; Garg and Ramakrishnan, 2020; Jin et al., 2020) and the failure cases of zero-shot cross-lingual transfer, and then study two widely used robust train-

ing methods to learn the robust model: (1) adversarial training (Goodfellow et al., 2015; Madry et al., 2018) and (2) randomized smoothing (Cohen et al., 2019; Ye et al., 2020). Both of them can make the model robust against perturbations in the input embeddings by modifying the training objective when fine-tuning model for the downstream task. For randomized smoothing, we also adopt the data augmentation approach (Ye et al., 2020) to learn the robust model.

We perform experiments on two cross-lingual text classification tasks, paraphrase identification and natural language inference[1]. The experimental results demonstrate that robust training indeed improves the performance of zero-shot cross-lingual transfer on the classification benchmarks: PAWS-X (Yang et al., 2019) and XNLI (Conneau et al., 2018b). On average the cross-lingual transfer performance improves by 2.1 and 1.6 points on PAWS-X and XNLI, respectively. In addition, we show that robust training remarkably improves *generalized* cross-lingual transfer (Lewis et al., 2020b). In this setting, the pair of input sentences in the text classification tasks belong to two different languages, e.g., paraphrase prediction for a pair of sentences in English and Korean.

5.2 Related Work

Zero-shot cross-lingual transfer learning. In recent years, several pre-trained multilingual language models are proposed for zero-shot cross-lingual transfer, including multilingual BERT (Devlin et al., 2019), XLM (Conneau and Lample, 2019), and XLM-R (Conneau et al., 2020a; Goyal et al., 2021). Many studies put attentions on the rationales that make zero-shot cross-lingual transfer work (K et al., 2020; Lauscher et al., 2020; Conneau et al., 2020b; Artetxe et al., 2020; Dufter and Schütze, 2020). Various tasks and datasests are presented to facilitate zero-shot cross-lingual transfer learning (Conneau et al., 2018b; Yang et al., 2019; Clark et al., 2020; Artetxe et al., 2020; Lewis et al., 2020b). XTREME (Hu et al., 2020) and XGLUE (Liang et al., 2020) further provide benchmarks for zero-shot cross-lingual transfer learning.

[1]Our code is available at https://github.com/uclanlp/Robust-XLT

Embedding space alignments. Learning to align embedding spaces have always been an important research topic to improve multilinguality. Early works focus on word embedding spaces (Mikolov et al., 2013; Smith et al., 2017; Artetxe et al., 2017). Recently, many approaches are proposed to align contextual word embedding spaces, such as learning rotation projections (Schuster et al., 2019; Aldarmaki and Diab, 2019; Conneau et al., 2020b) and fine-tuning pre-trained multilingual language models (Chi et al., 2021; Feng et al., 2022; Cao et al., 2020; Qin et al., 2020; Liu et al., 2020c; Dou and Neubig, 2021; Wei et al., 2021). However, most of them require additional supervision signals, such as parallel sentence pairs (Chi et al., 2021; Feng et al., 2022; Wei et al., 2021), bilingual dictionary (Cao et al., 2020; Qin et al., 2020; Liu et al., 2020c), or both (Pan et al., 2021). These additional supervised corpora are usually expensive for low-resource languages.

Embedding misalignment handling. Instead of directly aligning the representations, there is a line of research making the model be aware of the embedding misalignment issues by considering additional syntactic features, such as part-of-speech (Kozhevnikov and Titov, 2013) and dependency parse trees (Ahmad et al., 2019b; Subburathinam et al., 2019; Zhang et al., 2019b; Liu et al., 2019a; Ahmad et al., 2021a,b), and other syntactic features (Meng et al., 2019). However, those syntactic features require large human efforts to obtained.

Robust training. Recently, adversarial attacks are presented to check the robustness of NLP models, such as character manipulation (Ebrahimi et al., 2018; Gil et al., 2019), word replacements (Alzantot et al., 2018; Li et al., 2020b; Garg and Ramakrishnan, 2020; Jin et al., 2020), and syntactic rearrangements (Iyyer et al., 2018). To against those attacks, various robust training methods are proposed. For example, Alzantot et al. (2018) trains a robust model by data augmentation with generated adversarial examples. Other works (Ebrahimi et al., 2018; Dong et al., 2021; Zhou et al., 2021) consider adversarial training, which includes the adversarial accuracy to the training objective. A few studies propose transformations on inputs before feeding them to models (Edizel et al., 2019; Jones et al., 2020). Randomized smoothing (Cohen et al., 2019; Ye et al., 2020) is presented to make models robust against noise

in input representations. Another line of research aims at providing theoretical guarantee of robustness, including interval bound propagation methods (Jia et al., 2019; Huang et al., 2019a) and verification methods (Shi et al., 2020). Most of those robust training methods focus on defending adversarial attacks, while we propose to apply robust training methods to improve the zero-shot cross-lingual transfer performance.

5.3 Zero-Shot Cross-Lingual Transfer with Robust Training

In this work, we focus on zero-shot cross-lingual transfer for text classification tasks. Our goal is to learn a classifier f from a set of training examples in source languages $X_{src} = \{(x_i, y_i)\}_{i=1}^{N}$. At test time, we directly use the classifier f to conduct inference on a set of test examples in target languages $X_{tgt} = \{x_i\}_{i=1}^{M}$. We expect that the classifier f can transfer the learned knowledge from the source languages to the target languages.

5.3.1 Connection with Adversarial Examples

The aligned representations of different languages have been shown as a crucial factor (Cao et al., 2020; Chi et al., 2021; Pan et al., 2021) for multilingual embeddings to be effective for zero-shot cross-lingual transfer. For example, assuming the source language and the target language are English and French, respectively, and considering a pair of parallel sentences "*this is a cat*" (in English) and "*Ceci est un chat*" (in French), we can get the contextual representations of the source English sentence $\mathbf{E}_{src} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4)$ and the target French sentence $\mathbf{E}_{tgt} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4)$. Let $\boldsymbol{\delta}$ denote the difference between the source and the target contextual representations as

follows.[2]

$$\begin{aligned}\boldsymbol{\delta} &= \mathbf{E}_{src} - \mathbf{E}_{tgt} \\ &= (\mathbf{v}_1 - \mathbf{u}_1, \mathbf{v}_2 - \mathbf{u}_2, \mathbf{v}_3 - \mathbf{u}_3, \mathbf{v}_4 - \mathbf{u}_4) \\ &= (\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \boldsymbol{\delta}_3, \boldsymbol{\delta}_4).\end{aligned}$$

Since words with similar meanings have similar representations, the norm of their differences $\|\boldsymbol{\delta}_i\|$ is supposed to be small. Therefore, if $f(\mathbf{E}_{src}) = c$, we have a high probability for $f(\mathbf{E}_{tgt}) = c$ as well, which means that the classifier f is able to transfer the learned knowledge from the source language to the target language. If unfortunately, the transfer fails, we have

$$\begin{aligned} &f(\mathbf{E}_{tgt}) = f(\mathbf{E}_{src} + \boldsymbol{\delta}) \neq f(\mathbf{E}_{src}), \\ &\text{where } \|\boldsymbol{\delta}_i\| \text{ is small.} \end{aligned} \tag{5.1}$$

We observe that Eq. (5.1) is very similar to the definition of *adversarial examples* (Alzantot et al., 2018; Li et al., 2020b; Garg and Ramakrishnan, 2020; Jin et al., 2020). The goal of adversarial examples is to find a small perturbation $\boldsymbol{\Delta}$ for an instance $\mathbf{x}$ such that a classifier h changes the prediction on $\mathbf{x}$, as illustrated by the following equation.

$$\begin{aligned} &h(\tilde{\mathbf{x}}) = h(\mathbf{x} + \boldsymbol{\Delta}) \neq h(\mathbf{x}), \\ &\text{where } \|\boldsymbol{\Delta}\| \text{ is small.} \end{aligned} \tag{5.2}$$

For the case that cross-lingual transfer fails, the difference between the source and target representations $\boldsymbol{\delta}$ behaves like an adversarial perturbation. This inspires us to consider robust training methods, which are designed for defending adversarial examples, to improve the zero-shot cross-lingual transfer performance. More specifically, our goal is to train a robust classifier that can tolerate small perturbations on input embeddings. As shown in Figure 5.1b, we aim to train a robust classifier f that has robust regions (orange circles) such that the robust classifier f outputs similar values for input embeddings are in the same robust region.

[2]For the ease of describing our idea, we assume the word orders in different languages are the same. Later in experiments, we relax this condition and present a preliminary study on the influence of word orders in Section 5.4.4.

We study two widely used robust training methods in literature: (1) adversarial training and (2) randomized smoothing, as they have been successfully used for defending adversarial attacks (Ebrahimi et al., 2018; Jia et al., 2019; Huang et al., 2019a; Cohen et al., 2019).

5.3.2 Adversarial Training

The main idea of adversarial training is considering the most effective adversarial perturbation in each optimization iteration. More precisely, in normal training, we learn a classifier f by solving the following optimization problem

$$\min_{f} \sum_{(x,y)\in X_{src}} \mathcal{L}(f(\text{Enc}(x)), y),$$

where $\text{Enc}(\cdot)$ is the multilingual encoder and $\mathcal{L}$ is the cross-entropy loss. When considering adversarial training, we solve the following min-max optimization problem instead

$$\min_{f} \sum_{(x,y)\in X_{src}} \max_{\|\boldsymbol{\delta}_i\|\leq\varepsilon} \mathcal{L}(f(\text{Enc}(x)+\boldsymbol{\delta}), y),$$

where ε is a hyper-parameter to control the size of robust regions which are described by several norm balls $\|\boldsymbol{\delta}_i\|$. The inner maximization finds the most effective perturbation to change the prediction, while the outer minimization tries to ensure the correct prediction against the perturbation. With this min-max optimization, the classifier f is aware of perturbations within the robust regions $\|\boldsymbol{\delta}_i\|$ and becomes more robust.

5.3.3 Randomized Smoothing

Unlike adversarial training, which always considers the most effective perturbation, randomized smoothing focuses on the expectation case and aims to guarantee the local smoothness of the classifier at the same time. Following previous work (Cohen et al., 2019; Ye et al., 2020), we let f be the classifier learned by solving the normal optimization problem and learn a smoothed classifier g such that

$$g(\text{Enc}(x)) = \arg\max_{c\in\mathcal{Y}} \mathbb{P}_{\boldsymbol{\delta}}(f(\text{Enc}(x)+\boldsymbol{\delta}) = c),$$

where $\mathbb{P}_{\boldsymbol{\delta}}$ is a prior distribution of the perturbation $\boldsymbol{\delta}$ and $\mathcal{Y}$ is the label space. In other words, we want that $g(\text{Enc}(x))$ has a similar output value (label predictions) to $f(\text{Enc}(x))$. The random perturbation $\boldsymbol{\delta}$ is introduced to ensure the local smoothness of g. That is, $g(\text{Enc}(x) + \boldsymbol{\delta})$, the output for the perturbed input, is similar to the output value of $g(\text{Enc}(x))$. Compared to the original classifier f, the smoothed classifier g is more robust against local perturbations.

We consider two different ways to learn the smoothed classifier g: (1) random perturbation and (2) data augmentation.

Random perturbation (RP). Specifically, we focus on the following objective

$$\min_{g} \sum_{(x,y)\in X_{src}} \mathbb{P}_{\boldsymbol{\delta}}(\mathcal{L}(g(\text{Enc}(x) + \boldsymbol{\delta}), y)).$$

In each optimization step, we randomly sample a perturbation $\boldsymbol{\delta}$ from $\mathbb{P}_{\boldsymbol{\delta}}$ and add it to $\text{Enc}(x)$. Then, we use the perturbed representation as the input to calculate the loss and update the classifier g.

Data augmentation (DA). Another common way to approximate the smoothed classifier g is data augmentation (Ye et al., 2020). Instead of randomly sampling the perturbation $\boldsymbol{\delta}$, we consider a predefined synonym set (Alzantot et al., 2018). For every example $x = (w_1, w_2, ..., w_n)$ in X_{src}, we generate m augmented examples by replacing each word w_i in x with one of its synonym words (including w_i itself). We allow multiple replacements in one example. Then, we use the augmented data to train a smoothed classifier g.

It is worth noting that the predefined synonym set is required for *only* source languages. Unlike previous work (Qin et al., 2020; Liu et al., 2020c), which uses bilingual dictionary of *both* source languages and target languages, the proposed method does not need any additional annotations of target languages.

5.4 Experiments

We conduct experiments to verify that robust training indeed improves the performance of zero-shot cross-lingual transfer.

5.4.1 Setup

We consider two cross-lingual text classification datasets: Cross-lingual Paraphrase Adversaries from Word Scrambling (PAWS-X) (Yang et al., 2019) and Cross-lingual Natural Language Inference (XNLI) (Conneau et al., 2018b). The goal of PAWS-X is to determine whether two sentences are paraphrases to each other or not. XNLI is designed for natural language inference; given a premise and a hypothesis, the classifier predicts the relation of the two sentences from {*entailment*, *neutral*, *contradiction*}.

For both datasets, we consider English as the source language and treat other languages as the target languages. We use the train, validation, and test splits provided by XTREME framework (Hu et al., 2020). Specifically, we conduct 10 runs of experiments with 10 different random seeds. In each run, we train the classifier on the English training set, use the English validation set to search the best parameters, and record the results of the test sets. Finally, the averaged results of 10-run experiments are reported.

Compared models. We consider the following four different models:

- **mBERT**: the standard multilingual BERT (Devlin et al., 2019).
- **mBERT-ADV**: multilingual BERT with adversarial training.
- **mBERT-RS-RP**: multilingual BERT with randomized smoothing via random perturbation.
- **mBERT-RS-DA**: multilingual BERT with randomized smoothing via data augmentation.

Implementation details. For adversarial training, we consider L_∞-norm as the norm of perturbation $\|\boldsymbol{\delta}_i\|$. The size of robust regions is searched from $\{0.001, 0.01, 0.1, 1.0\}$. For the randomized smoothing via random perturbation, we consider uniform distribution over a L_∞-norm ball. The size of ball is searched from $\{0.001, 0.01, 0.1, 1.0\}$.

Model	en	de	es	fr	ja	ko	zh	avg.
mBERT*	94.0	85.7	87.4	87.0	73.0	69.6	77.0	82.0
mBERT (reproduce)	93.7	85.4	88.2	87.8	75.3	74.2	79.1	83.4
mBERT-ADV	93.7	86.5	88.5	87.8	76.1	75.3	80.4	84.0
mBERT-RS-RP	**94.5**	87.4	**90.0**	**89.5**	77.9	77.5	**82.0**	**85.5**
mBERT-RS-DA	93.5	**87.8**	88.8	88.8	**79.3**	**78.3**	81.5	85.4

Table 5.1: Averaged results of zero-shot cross-lingual transfer on PAWS-X with 10 different random seeds. Highest scores are in bold. Underlines denote that the improvement is significant with $p \leq 0.05$ for the bootstrapped paired t-test. *We report the numbers in the previous paper (Hu et al., 2020).

For the randomized smoothing via data augmentation, we consider the synonym set provide by previous work (Alzantot et al., 2018), which is constructed by searching nearest neighbors of words in the GloVe embedding space (Pennington et al., 2014) post-processed by the counter-fitting method (Mrksic et al., 2016). The number of augmented examples m is set to 10 and 3 for PAWS-X and XNLI, respectively, while more discussion on m is shown in Section 5.4.2. For other parameters, such as the learning rate and the batch size, we follow the training scripts provided by XTREME framework (Hu et al., 2020).

5.4.2 Zero-Shot Cross-Lingual Transfer

Table 5.1 shows the averaged results of PAWS-X with 10 different random seeds. We first notice that all mBERT-ADV, mBERT-RS-RP, and mBERT-RS-DA perform better than the standard mBERT on average. Especially, robust training leads to up to 4.0% improvement on Japanese, up to 4.1% improvement on Korean, and up to 2.9% improvement on Chinese. The results suggest that robust training helps in improving the performance of zero-shot cross-lingual transfer learning.

We observe that randomized smoothing is usually better than adversarial training. The reason is that adversarial training always considers the most effective adversarial perturbation during the optimization process. Adversarial perturbations are suitable for defending adversarial examples as they are specifically designed for attacking the classifier. However, in the zero-shot cross-lingual transfer case, the perturbations are not explicitly designed but reflect the natural difference between languages. There-

Model	en	ar	bg	de	el	es	fr	hi
mBERT*	80.8	64.3	68.0	70.0	65.3	73.5	73.4	58.9
mBERT (reproduce)	82.3	64.8	68.2	70.8	66.4	74.3	73.7	59.7
mBERT-ADV	81.9	64.9	68.3	71.7	66.5	74.4	74.5	59.6
mBERT-RS-RP	**82.6**	65.4	68.7	70.5	67.2	**75.0**	74.1	59.8
mBERT-RS-DA	81.0	**66.4**	**69.9**	**71.8**	**68.0**	74.7	**74.2**	**62.7**
Model	ru	sw	th	tr	ur	vi	zh	avg.
mBERT*	67.8	49.7	54.1	60.9	57.2	69.3	67.8	65.4
mBERT (reproduce)	68.7	50.0	53.0	60.9	57.7	70.3	69.2	66.0
mBERT-ADV	68.8	48.8	50.6	61.7	59.2	70.0	69.4	66.0
mBERT-RS-RP	69.5	48.4	50.5	59.7	57.9	70.5	69.7	66.0
mBERT-RS-DA	**70.6**	**51.1**	**55.7**	**62.9**	**60.9**	**71.8**	**71.4**	**67.6**

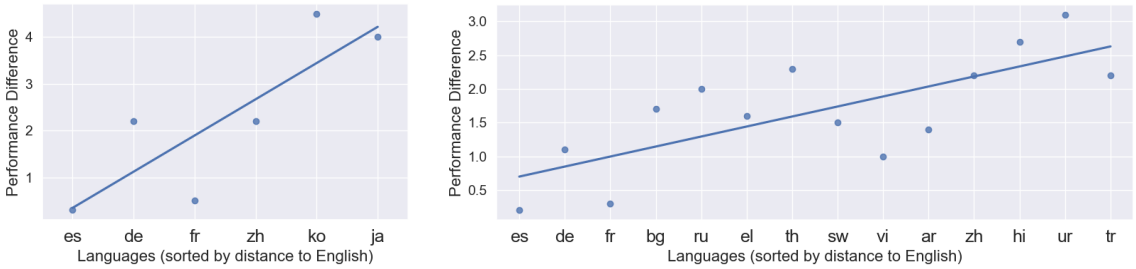
Table 5.2: Averaged results of zero-shot cross-lingual transfer on XNLI with 10 different random seeds. Highest scores are in bold. Underlines denote that the improvement is significant with $p \leq 0.05$ for the bootstrapped paired t-test. *We report the numbers in the previous paper (Hu et al., 2020).

fore, randomized smoothing, which considers the average case, becomes the better choice.

We have a similar conclusion for the XNLI dataset. As shown in Table 5.2, robust training indeed leads to improvements on zero-shot cross-lingual transfer. Again, randomized smoothing performs better than the adversarial training approach.

Finally, we compare the two different ways (random perturbation and data augmentation) to learn the smoothed classifier. They have competitive performance on PAWS-X; however, data augmentation performs better than random perturbation on XNLI. We hypothesize that the ideal robust regions in practice may not be perfect norm balls. In fact, they are more like convex hulls composed by the neighbor words (Dong et al., 2021). By considering a predefined synonym set, mBERT-RS-DA can better capture the shapes of robust regions, leading to a more stable performance.

What languages are benefited most from robust training? We notice that cross-lingual transfer to some languages is significantly improved by robust training, especially those languages that are quite different from the source language (English). To verify this conjecture, we consider lang2vec (Littell et al., 2017), a tool that ex-



(a) PAWS-X **(b)** XNLI

Figure 5.2: Performance difference between mBERT-RS-DA and mBERT over different languages. We sort the languages according to their distances to English from left (small) to right (large). Performance on languages with larger distances to English is improved more with the robust training.

tracts features of different languages by querying the URIEL typological database[3], to calculate the distance between English and other languages. Then, we show the performance gaps between mBERT-RS-DA and mBERT over all languages as well as the least square regression line in Figure 5.2. Note that the languages are sorted according to their distances to English from left to right.

From Figure 5.2a, we observe an obvious trend for PAWS-X that languages with larger distances to English have more performance gain with robust training. We posit that it is because languages with larger distances have more different representations from English in the multilingual embedding space. The norm of the perturbation $\boldsymbol{\delta}$ defined in Section 5.3 will be larger and thus the failure cases occur more often. By performing robust training, we reduce failure cases that lead to a larger improvement. Similar trend can be observed for XNLI (Figure 5.2b). Performance on languages with larger distances to English is improved more with the robust training.

How many augmented data needed for randomized smoothing? Since mBERT-RS-DA seems to be the most effective model for both PAWS-X and XNLI, we do further ablation on the number of augmented data for each example m. Figure 5.3 shows the average performance of mBERT-RS-DA on PAWS-X over different choices of m. We can observe that larger m leads to better performance in general because more augmented examples help the model better approximate the local smoothness,

[3] http://www.cs.cmu.edu/~dmortens/projects/7_project

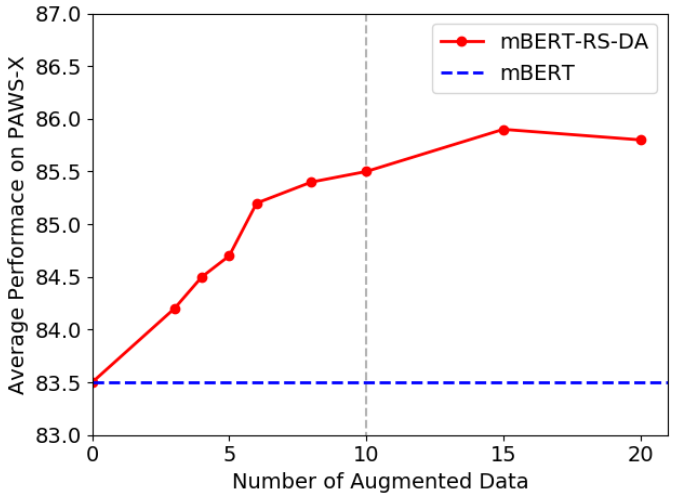
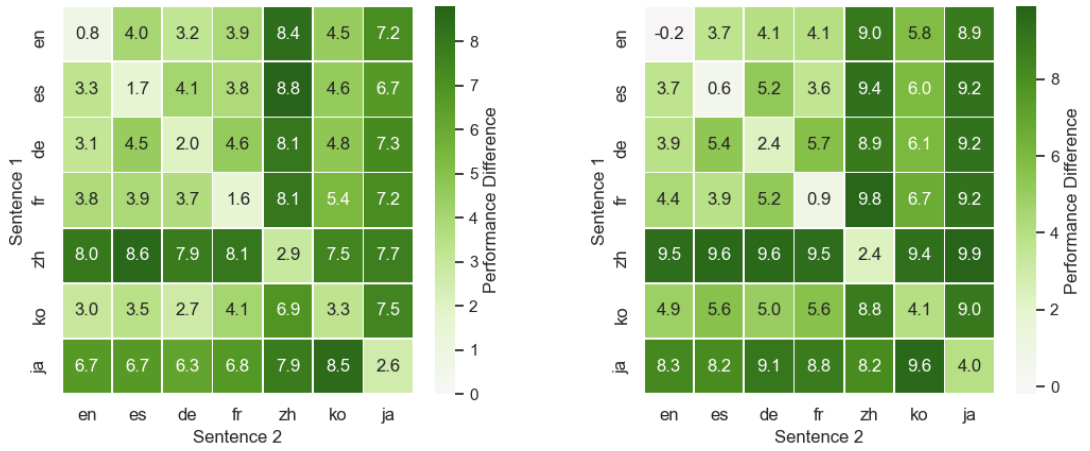


Figure 5.3: Performance of mBERT-RS-DA on PAWS-X over different m (the number of augmented instances generated by synonym replacements).

	en	es	de	fr	zh	ko	ja	avg.
en	93.7	85.4	85.0	85.0	66.5	66.4	63.4	77.9
es	86.1	88.2	80.5	83.9	63.7	64.0	60.9	75.3
de	85.6	79.7	85.4	79.8	63.9	64.7	61.5	74.4
fr	84.8	83.0	80.3	87.8	63.7	63.9	61.1	74.9
zh	66.7	63.7	63.9	64.3	79.1	62.4	64.3	66.3
ko	67.1	64.9	65.3	65.0	62.7	74.2	65.1	66.3
ja	63.0	61.1	61.1	61.1	64.6	63.6	75.3	64.3
avg.	78.1	75.1	74.5	75.3	66.3	65.6	64.5	71.3

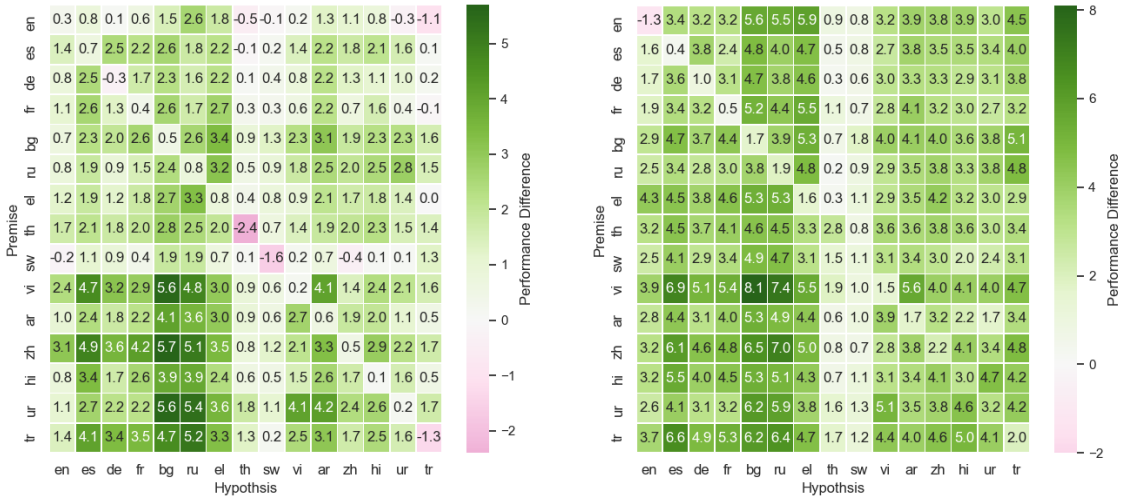
Table 5.3: Results for mBERT on PAWS-X.

resulting in more accurate robust regions. Interestingly, when $m \leq 10$, increasing m can significantly improve the performance. When $m > 10$, increasing m only slightly improves the performance. This result suggests that setting m to 10 for PAWS-X. Interestingly, we observe that setting m to 3 is good enough for XNLI. This ablation study indicates that randomized smoothing with data augmentation can use just a few augmented instances per example to learn good robust regions.



(a) mBERT-RS-RP **(b)** mBERT-RS-DA

Figure 5.4: Results for generalized zero-shot cross-lingual transfer on PAWS-X. We report the performance difference between the compared model and mBERT over different combinations of languages.



(a) mBERT-RS-RP **(b)** mBERT-RS-DA

Figure 5.5: Results for generalized zero-shot cross-lingual transfer on XNLI. We report the performance difference between the compared model and mBERT over different combinations of languages.

5.4.3 Zero-Shot Generalized Cross-Lingual Transfer Results

Next, we study the zero-shot cross-lingual transfer in a *generalized* setting. Lewis et al. (2020b) proposed the generalized setting for the question answering task where

	en	es	de	fr	zh	ko	ja	avg.
en	94.5	89.3	88.2	88.9	74.9	70.8	70.6	82.5
es	89.3	90.0	84.6	87.7	72.5	68.6	67.5	80.0
de	88.7	84.2	87.4	84.4	72.0	69.5	68.8	79.3
fr	88.6	86.9	83.9	89.5	71.8	69.3	68.3	79.7
zh	74.6	72.3	71.9	72.4	82.0	69.9	72.0	73.6
ko	70.1	68.4	68.0	69.0	69.6	77.5	72.5	70.7
ja	69.7	67.7	67.5	67.9	72.5	72.1	77.9	70.7
avg.	82.2	79.8	78.8	80.0	73.6	71.1	71.1	76.7

Table 5.4: Results for mBERT-RS-RP on PAWS-X.

	en	es	de	fr	zh	ko	ja	avg.
en	93.5	89.1	89.1	89.1	75.6	72.2	72.3	83.0
es	89.7	88.8	85.7	87.5	73.0	70.0	70.1	80.7
de	89.5	85.1	87.9	85.5	72.8	70.7	70.7	80.3
fr	89.2	86.9	85.5	88.8	73.4	70.5	70.3	80.7
zh	76.1	73.3	73.6	73.9	81.5	71.8	74.2	74.9
ko	72.0	70.4	70.3	70.5	71.5	78.3	74.1	72.4
ja	71.3	69.2	70.2	69.8	72.8	73.2	79.3	72.3
avg.	83.0	80.4	80.3	80.7	74.4	72.4	73.0	77.7

Table 5.5: Results for mBERT-RS-DA on PAWS-X.

the question and the context may belong to two different languages.[4] We consider the generalized setting for cross-lingual text classification since the input of PAWS-X and XNLI tasks are pairs of sentences. For example, consider XNLI on English-Arabic sentence pairs; the premises are in English, and the hypotheses are in Arabic. Note that due to the parallel nature of PAWS-X and XNLI dataset[5], we can pair up sentences from two different languages. Notice that we directly use the trained models in Section 5.4.2 to conduct inference in the generalized setting. In other words, all the classifiers are trained on English-English sentence pairs, without the consideration of target languages.

The results of mBERT-RS-RP and mBERT-RS-DA on PAWS-X and XNLI over all combinations of languages are shown in Figure 5.4 and Figure 5.5, respectively.

[4]QA systems should be able to answer questions written in French by reading an English context.
[5]PAWS-X and XNLI datasets consist of 7-way and 15-way parallel sentence pairs.

	en	es	de	fr	bg	ru	el	th	sw	vi	ar	zh	hi	ur	tr	avg.
en	82.3	70.3	65.8	69.7	60.5	63.1	55.3	44.6	41.1	63.9	57.7	64.6	52.0	49.5	52.3	59.5
es	73.5	74.3	62.9	69.0	60.5	63.7	57.3	44.6	40.6	61.4	57.9	60.8	50.4	47.1	51.6	58.4
de	71.8	65.5	70.8	65.6	59.5	63.3	55.8	44.3	41.0	60.2	56.5	60.1	52.5	49.4	52.0	57.9
fr	73.6	69.0	64.0	73.8	59.5	63.1	55.7	44.1	40.5	62.2	57.3	61.6	51.1	48.5	51.8	58.4
bg	67.8	63.7	60.8	62.5	68.2	64.2	56.0	44.2	39.9	57.4	56.3	57.8	51.2	47.2	50.3	56.5
ru	69.1	65.2	62.6	64.4	62.7	68.7	55.0	44.2	39.9	59.0	56.7	58.6	50.6	46.8	50.0	56.9
el	62.7	61.4	58.0	60.2	57.1	57.7	66.4	44.4	40.5	56.4	55.6	54.0	49.6	46.8	50.7	54.8
th	54.8	52.0	49.9	51.3	49.1	50.4	49.0	53.0	39.4	51.1	49.9	49.3	45.9	44.8	45.4	49.0
sw	54.2	51.2	48.7	50.5	47.2	47.9	47.9	41.8	50.0	48.5	49.1	48.5	45.4	44.4	45.8	48.1
vi	67.4	60.3	57.4	61.2	52.9	57.1	52.9	44.2	39.8	70.3	53.3	62.0	49.2	45.9	47.5	54.8
ar	63.9	60.4	57.0	59.5	54.5	57.1	53.3	43.9	40.4	55.4	64.8	55.2	50.3	48.4	49.9	54.3
zh	67.9	59.9	57.2	59.9	53.4	56.5	50.4	42.7	39.6	60.8	53.5	69.2	48.0	45.7	48.0	54.2
hi	61.4	55.5	55.0	55.3	52.6	54.4	51.9	43.8	40.3	53.8	53.1	53.7	59.7	52.7	49.9	52.9
ur	60.1	54.0	53.9	55.1	48.8	51.5	49.6	41.9	39.7	50.0	52.1	52.3	54.4	57.7	48.2	51.3
tr	61.0	55.1	53.6	55.1	52.0	52.6	50.9	42.4	40.7	52.3	52.0	53.2	49.7	47.3	60.9	51.9
avg.	66.1	61.2	58.5	60.9	55.9	58.1	53.8	44.3	40.9	57.5	55.1	57.4	50.7	48.1	50.3	54.6

Table 5.6: Results for mBERT on XNLI.

While the diagonal numbers indicate the transfer results in the cross-lingual transfer settings, the non-diagonal entries present the generalized transfer performances. Note that we report the performance difference between the compared model and mBERT (exact numbers can be found in Table 5.3, 5.4, 5.5, 5.6, 5.7, and 5.8) and the languages are sorted according to their distances to English. We observe that the non-diagonal numbers are much larger than the diagonal numbers, which suggests that robust training results in larger performance improvements in the generalized cross-lingual transfer setting. Given that the input sentences in training examples are in the same language (English), during inference, mBERT makes more mistakes in the classification tasks as the contextual representations for the input sentences may not be aligned accurately. However, mBERT-RS-RP and mBERT-RS-DA can tolerate a certain amount of noise in input embeddings. Therefore, they are more stable when the input sentences come from different languages, leading to a significant improvement.

5.4.4 Study on Syntactic Perturbations

As mentioned in Section 5.3, our primary focus is on the perturbations in the multilingual embedding space and does not consider the influence of language syntax in

	en	es	de	fr	bg	ru	el	th	sw	vi	ar	zh	hi	ur	tr	avg.
en	82.6	71.2	65.9	70.3	62.0	65.7	57.0	44.1	40.9	64.1	58.9	65.7	52.8	49.2	51.2	60.1
es	74.9	75.0	65.4	71.2	63.0	65.6	59.5	44.5	40.8	62.9	60.1	62.6	52.5	48.7	51.7	59.9
de	72.6	68.0	70.5	67.4	61.7	64.9	58.0	44.4	41.4	61.0	58.8	61.4	53.6	50.4	52.2	59.1
fr	74.7	71.6	65.2	74.1	62.1	64.8	58.4	44.4	40.8	62.7	59.5	62.4	52.7	48.9	51.7	59.6
bg	68.5	66.0	62.9	65.1	68.7	66.8	59.4	45.1	41.1	59.7	59.4	59.7	53.5	49.6	51.8	58.5
ru	69.9	67.1	63.5	65.9	65.0	69.5	58.2	44.7	40.9	60.8	59.2	60.6	53.1	49.5	51.5	58.6
el	63.9	63.3	59.3	62.0	59.8	61.0	67.2	44.7	41.3	57.3	57.7	55.7	51.4	48.2	50.7	56.2
th	56.4	54.1	51.7	53.3	51.9	52.9	51.0	50.5	40.1	52.5	51.8	51.3	48.2	46.3	46.8	50.6
sw	54.1	52.3	49.6	50.9	49.1	49.7	48.6	41.8	48.4	48.7	49.8	48.1	45.4	44.5	47.1	48.6
vi	69.9	65.0	60.5	64.1	58.6	61.8	56.0	45.1	40.4	70.5	57.4	63.4	51.5	48.0	49.1	57.4
ar	64.9	62.8	58.7	61.7	58.7	60.7	56.3	44.7	41.1	58.1	65.4	57.1	52.2	49.6	50.3	56.1
zh	71.1	64.8	60.8	64.1	59.0	61.6	53.9	43.5	40.8	63.0	56.8	69.7	50.9	47.8	49.7	57.2
hi	62.2	58.9	56.7	57.9	56.5	58.3	54.3	44.5	40.8	55.3	55.8	55.3	59.8	54.2	50.4	54.7
ur	61.2	56.7	56.1	57.3	54.4	57.0	53.2	43.7	40.8	54.1	56.3	54.6	56.9	57.9	49.9	54.0
tr	62.4	59.2	57.0	58.6	56.7	57.9	54.2	43.7	40.9	54.8	55.1	54.9	52.2	48.8	59.7	54.4
avg.	67.3	63.7	60.3	62.9	59.1	61.2	56.4	44.6	41.4	59.0	57.5	58.8	52.4	49.4	50.9	56.3

Table 5.7: Results for mBERT-RS-RP on XNLI.

cross-lingual transfer. Different languages have linguistic differences, such as word order. Differences in word order across languages affect the contextual embedding space that impacts cross-lingual transfer (Ahmad et al., 2019b). Therefore, we conduct a preliminary experiment to study the influence of syntax in robust training.

mBERT-RS-DA uses a predefined synonym set to generate perturbed examples for data augmentation. Following a similar strategy, we construct *syntactically perturbed examples* for data augmentation. More specifically, for every example $x = (w_1, w_2, ..., w_n)$ in X_{src}, we generate m syntactically perturbed examples by randomly swapping adjacent words with a probability $p = 0.1$. This random swapping may result in some examples with different word orders, which simulates the syntactic perturbations. Then, we use those syntactically perturbed examples to train the smoothed classifier g, called mBERT-RS-syntax.

Table 5.9 presents the preliminary results. The average performance of mBERT-RS-syntax is similar to the performance of standard mBERT. Interestingly, the zero-shot cross-lingual transfer performance drops when the target languages are more similar to the source language English (German, Spanish, and French), while the transfer performance increases when the target languages are more different from English (Japanese, Korean, and Chinese). This preliminary result suggests that it is possible to improve the zero-shot cross-lingual transfer by considering syntactic

	en	es	de	fr	bg	ru	el	th	sw	vi	ar	zh	hi	ur	tr	avg.
en	81.0	73.7	69.0	72.8	66.2	68.6	61.2	45.5	41.9	67.1	61.6	68.4	55.9	52.4	56.8	62.8
es	75.2	74.7	66.7	71.4	65.3	67.7	62.0	45.1	41.4	64.1	61.7	64.3	53.9	50.5	55.5	61.3
de	73.4	69.2	71.9	68.7	64.1	67.1	60.4	44.6	41.6	63.2	59.9	63.4	55.4	52.4	55.8	60.7
fr	75.4	72.4	67.2	74.2	64.7	67.5	61.2	45.2	41.2	65.0	61.4	64.8	54.2	51.2	55.0	61.4
bg	70.7	68.4	64.6	66.9	69.9	68.0	61.3	44.9	41.6	61.4	60.4	61.7	54.8	51.0	55.4	60.1
ru	71.6	68.6	65.4	67.4	66.4	70.6	59.7	44.4	40.9	61.9	60.2	62.4	53.9	50.5	54.8	59.9
el	67.0	65.9	61.9	64.8	62.3	63.0	68.0	44.6	41.6	59.3	59.1	58.2	52.7	49.7	53.7	58.1
th	58.0	56.4	53.6	55.4	53.7	54.9	52.4	55.8	40.3	54.7	53.5	53.0	49.5	47.8	48.8	52.5
sw	56.7	55.3	51.7	53.9	52.2	52.6	51.0	43.2	51.1	51.6	52.5	51.5	47.4	46.8	48.9	51.1
vi	71.3	67.2	62.4	66.5	61.0	64.5	58.5	46.1	40.8	71.8	58.9	66.0	53.3	49.9	52.2	59.4
ar	66.7	64.9	60.0	63.5	59.8	61.9	57.7	44.4	41.5	59.3	66.4	58.4	52.5	50.1	53.3	57.4
zh	71.2	66.0	61.9	64.7	59.9	63.5	55.4	43.5	40.3	63.6	57.3	71.5	52.1	49.1	52.8	58.2
hi	64.6	60.9	59.0	59.9	57.9	59.5	56.2	44.5	41.4	56.9	56.6	57.8	62.8	57.4	54.1	56.6
ur	62.8	58.1	57.0	58.3	55.0	57.5	53.5	43.6	41.0	55.1	55.6	56.1	58.9	60.9	52.4	55.0
tr	64.7	61.7	58.5	60.4	58.2	59.0	55.7	44.1	41.9	56.6	56.0	57.7	54.7	51.4	62.9	56.2
avg.	68.7	65.6	62.0	64.6	61.1	63.1	58.3	45.3	41.9	60.8	58.7	61.0	54.1	51.4	54.2	58.0

Table 5.8: Results for mBERT-RS-DA on XNLI.

Model	en	de	es	fr	ja	ko	zh	avg.
mBERT*	**94.0**	85.7	87.4	87.0	73.0	69.6	77.0	82.0
mBERT (reproduce)	93.7	85.4	88.2	87.8	75.3	74.2	79.1	83.4
mBERT-RS-DA	93.5	**87.8**	**88.8**	88.8	**79.3**	**78.3**	81.5	85.4
mBERT-RS-syntax	93.0	85.5	87.7	88.0	76.5	76.7	80.7	83.5

Table 5.9: Results of syntactic perturbations on PAWS-X. Highest scores are in bold. Underlines denote that the improvement is significant with $p \leq 0.05$ for the bootstrapped paired t-test. *We report the numbers in the previous paper (Hu et al. (2020)).

perturbations. One potential extension is adopting paraphrase generation models (Iyyer et al., 2018; Huang and Chang, 2021) to construct more sophisticated syntactic perturbations and we leave this direction for future work.

5.5 Summary

In this work, we propose a robust model by drawing connections between adversarial examples and the failure cases of zero-shot cross-lingual transfer. We adopt two robust training methods, adversarial training and randomized smoothing, to train the desired robust model. The experimental results demonstrate that robust training

improves zero-shot cross-lingual transfer on text classification tasks. In addition, the improvement is more significant in the generalized cross-lingual transfer setting.

CHAPTER 6

Improving Language-Level Robustness with Generation-Based Models

6.1 Introduction

Event argument extraction (EAE) aims to recognize the entities serving as event arguments and identify their corresponding roles. As illustrated by the English example in Figure 6.1, given a trigger word *"destroyed"* for a *Conflict:Attack* event, an event argument extractor is expected to identify *"commando"*, *"Iraq"*, and *"post"* as the event arguments and predict their roles as *"Attacker"*, *"Place"*, and *"Target"*, respectively.

Zero-shot cross-lingual EAE has attracted considerable attention since it eliminates the requirement of labeled data for constructing EAE models in low-resource languages (Subburathinam et al., 2019; Ahmad et al., 2021b; Nguyen and Nguyen, 2021). In this setting, the model is trained on the examples in the *source* languages and directly tested on the instances in the *target* languages.

Recently, generation-based models[1] have shown strong performances on monolingual structured prediction tasks (Yan et al., 2021; Huang et al., 2021b; Paolini et al., 2021), including EAE (Li et al., 2021; Hsu et al., 2022). These works fine-tune pre-trained generative language models to generate outputs following designed templates such that the final predictions can be easily decoded from the outputs. Compared to the traditional classification-based models (Wang et al., 2019b; Wadden et al., 2019; Lin et al., 2020), they better capture the structures and dependencies between

[1]We use pre-trained *generative* language models to refer to pre-trained models with encoder-decoder structure, such as BART (Lewis et al., 2020a), T5 (Raffel et al., 2020), and mBART (Liu et al., 2020b). For models adapting these pre-trained generative models to generate texts for downstream applications, we denote them as *generation-based* models.

Figure 6.1: An illustration of cross-lingual event argument extraction. Given sentences in arbitrary languages and their event triggers (*destroyed* and 起义), the model needs to identify arguments (*commando*, *Iraq* and *post* v.s. 军队, and 反对派) and their corresponding roles (Attacker, Target, and Place).

entities, as the templates provide additional declarative information.

Despite the successes, the designs of templates in prior works are language-dependent, which makes it hard to be extended to the zero-shot cross-lingual transfer setting (Subburathinam et al., 2019; Ahmad et al., 2021b). Naively applying such models trained on the source languages to the target languages usually generates *code-switching* outputs, yielding poor performance for zero-shot cross-lingual transfer,[2] as we will empirically show in Section 6.5.4. How to design *language-agnostic* generation-based models for zero-shot cross-lingual structured prediction problems is still an open question.

In this work, we present a study that leverage multilingual pre-trained generative models for zero-shot cross-lingual event argument extraction and propose X-GEAR (**Cross**-lingual **G**enerative **E**vent **A**rgument extracto**R**). Given an input passage and a carefully designed prompt that contains an event trigger and the corresponding language-agnostic template, X-GEAR is trained to generate a sentence that fills in a language-agnostic template with arguments. X-GEAR inherits the strength of generation-based models that captures event structures and the dependencies between entities better than classification-based models. Moreover, the pre-trained decoder inherently identifies named entities as candidates for event arguments and does not

[2]For example, TANL (Paolini et al., 2021) is trained to generate "`[Two soldiers|target] were attacked`" to represent *Two soldiers* being a *target* argument. When directly applying it to Chinese, the ground truth for TANL becomes "`[两位士兵|target]`被攻击", which is a sentence alternating between Chinese and English.

need an additional named entity recognition module. The *language-agnostic templates* prevents the model from overfitting to the source language's vocabulary and facilitates cross-lingual transfer.

We conduct experiments on two multilingual EAE datasets: ACE-2005 (Doddington et al., 2004) and ERE (Song et al., 2015). The results demonstrate that X-Gear outperforms the state-of-the-art zero-shot cross-lingual EAE models. We further perform ablation studies to justify our design and present comprehensive error analyses to understand the limitations of using multilingual generation-based models for zero-shot cross-lingual transfer. Our code is available at https://github.com/PlusLabNLP/X-Gear.

6.2 Related Work

Zero-shot cross-lingual structured prediction. Zero-shot cross-lingual learning is an emerging research topic as it eliminates the requirement of labeled data for training models in low-resource languages (Ruder et al., 2021; Huang et al., 2021a). Various structured prediction tasks have been studied, including named entity recognition (Pan et al., 2017; Huang et al., 2019b; Hu et al., 2020), dependency parsing (Ahmad et al., 2019b,a; Meng et al., 2019), relation extraction (Zou et al., 2018; Ni and Florian, 2019), and event argument extraction (Subburathinam et al., 2019; Nguyen and Nguyen, 2021; Fincke et al., 2022). Most of them are *classification-based models* that build classifiers on top of a multilingual pre-trained *masked* language models. To further deal with the discrepancy between languages, some of them require additional information, such as bilingual dictionaries (Liu et al., 2019a; Ni and Florian, 2019), translation pairs (Zou et al., 2018), and dependency parse trees (Subburathinam et al., 2019; Ahmad et al., 2021b; Nguyen and Nguyen, 2021). However, as pointed out by previous literature (Li et al., 2021; Hsu et al., 2022), classification-based models are less powerful to model dependencies between entities compared to *generation-based models*.

Generation-based structured prediction. Several works have demonstrated the great success of generation-based models on monolingual structured prediction tasks,

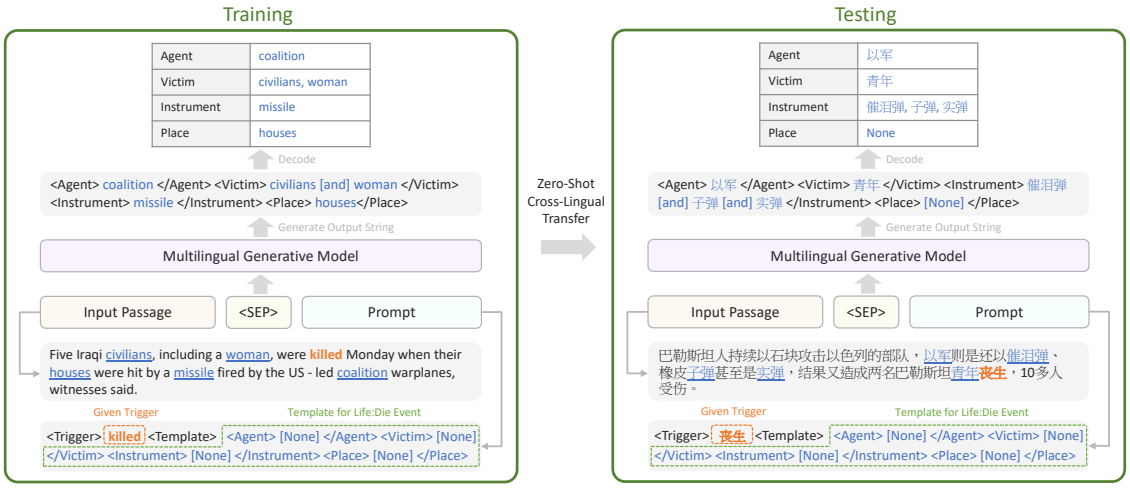


Figure 6.2: The overview of X-GEAR. Given an input passage and a carefully designed prompt containing an event trigger and a language-agnostic template, X-GEAR fills in the language-agnostic template with event arguments.

including named entity recognition (Yan et al., 2021), relation extraction (Huang et al., 2021b; Paolini et al., 2021), and event extraction (Du et al., 2021; Li et al., 2021; Hsu et al., 2022; Lu et al., 2021). Yet, as mentioned in Section 6.1, their designed generating targets are language-dependent. Accordingly, directly applying their methods to the zero-shot cross-lingual setting would result in less-preferred performance.

Prompting methods. There are growing interests recently to incorporate prompts on pre-trained language models in order to guide the models' behavior or elicit knowledge (Peng et al., 2019; Sheng et al., 2020; Shin et al., 2020; Schick and Schütze, 2021; Qin and Eisner, 2021; Scao and Rush, 2021). Following the taxonomy in (Liu et al., 2023), these methods can be classified depending on whether the language models' parameters are tuned and on whether trainable prompts are introduced. Our method belongs to the category that fixes the prompts and tunes the language models' parameters. Despite the flourish of the research in prompting methods, there is only limited attention being put on multilingual tasks (Winata et al., 2021).

6.3 Zero-Shot Cross-Lingual Event Argument Extraction

We focus on zero-shot cross-lingual EAE. Given an input passage and an event trigger, an EAE model identifies arguments and their corresponding roles. More specifically, as illustrated by the training examples in Figure 6.2, given an input passage $\mathbf{x}$ and an event trigger $\mathbf{t}$ (*killed*) belonging to an event type $\mathbf{e}$ (*Life:Die*), an EAE model predicts a list of arguments $\mathbf{a} = [a_1, a_2, ..., a_l]$ (*coalition*, *civilians*, *woman*, *missile*, *houses*) and their corresponding roles $\mathbf{r} = [r_1, r_2, .., r_l]$ (*Agent*, *Victim*, *Victim*, *Instrument*, *Place*). In the zero-shot cross-lingual setting, the training set $X_{train} = \{(\mathbf{x}_i, \mathbf{t}_i, \mathbf{e}_i, \mathbf{a}_i, \mathbf{r}_i)\}_{i=1}^{N}$ belongs to the source languages while the testing set $X_{test} = \{(\mathbf{x}_i, \mathbf{t}_i, \mathbf{e}_i, \mathbf{a}_i, \mathbf{r}_i)\}_{i=1}^{M}$ are in the target languages.

Similar to monolingual EAE, zero-shot cross-lingual EAE models are expected to capture the dependencies between arguments and make structured predictions. However, unlike monolingual EAE, zero-shot cross-lingual EAE models need to handle the differences (e.g., grammar, word order) between languages and learn to transfer the knowledge from the source languages to the target languages.

6.4 Proposed Method: X-Gear

We formulate zero-shot cross-lingual EAE as a language generation task and propose X-Gear, a **Cross**-lingual **G**enerative **E**vent **A**rgument extracto**R** that is illustrated in Figure 6.2. There are two challenges raised by this formulation: (1) The input language may vary during training and testing; (2) The generated output strings need to be easily parsed into final predictions. Therefore, the output strings have to reflect the change of the input language accordingly while remaining well-structured.

We address these challenges by designing *language-agnostic templates*. Specifically, given an input passage $\mathbf{x}$ and a designed prompt that contains the given trigger $\mathbf{t}$, its event type $\mathbf{e}$, and a *language-agnostic template*, X-Gear learns to generate an output string that fills in the language-agnostic template with information extracted from input passage. The language-agnostic template is designed in a structured way

such that parsing the final argument predictions $\mathbf{a}$ and role predictions $\mathbf{r}$ from the generated output is trivial. Moreover, since the template is language-agnostic, it facilitates cross-lingual transfer.

X-GEAR fine-tunes multilingual pre-trained generative models, such as mBART-50 (Tang et al., 2020) or mT5 (Xue et al., 2021), and augments them with a copy mechanism to better adapt to input language changes. We present its details as follows, including the language-agnostic templates, the target output string, the input format, and the training details.

6.4.1 Language-Agnostic Template

We create one language-agnostic template $T_{\mathbf{e}}$ for each event type $\mathbf{e}$, in which we list all possible associated roles[3] and form a unique HTML-tag-style template for that event type $\mathbf{e}$. For example, in Figure 6.2, the *Life:Die* event is associated with four roles: *Agent*, *Victim*, *Instrument*, and *Place*. Thus, the template for *Life:Die* events is designed as:

<Agent>`[None]`</Agent><Victim>`[None]`</Victim>
<Instrument>`[None]`</Instrument><Place>`[None]`</Place>.

For ease of understanding, we use English words to present the template. However, these tokens (`[None]`, <Agent>, </Agent>, <Victim>, etc.) are encoded as special tokens[4] that the pre-trained models have never seen and thus their representations need to be learned from scratch. Since these special tokens are not associated with any language and are not pre-trained, they are considered as *language-agnostic*.

6.4.2 Target Output String

X-GEAR learns to generate target output strings that follow the form of language-agnostic templates. To compose the target output string for training, given an instance $(\mathbf{x}, \mathbf{t}, \mathbf{e}, \mathbf{a}, \mathbf{r})$, we first pick out the language-agnostic template $T_{\mathbf{e}}$ for the event

[3]The associated roles can be obtained by skimming training data or directly from the annotation guideline if provided.

[4]In fact , the special tokens can be replaced by any other format, such as <–token1–> or </–token1–>. Here, we use <Agent> and </Agent> to highlight that arguments between these two special tokens are corresponding to the *Agent* role.

type **e** and then replace all "`[None]`" in $T_{\mathbf{e}}$ with the corresponding arguments in **a** according to their roles **r**. If there are multiple arguments for one role, we concatenate them with a special token "`[and]`". For instance, the training example in Figure 6.2 has two arguments (*civilians* and *woman*) for the *Victim* role, and the corresponding part of the output string would be

<Victim> `civilians [and] woman` </Victim>.

If there are no corresponding arguments for one role, we keep "`[None]`" in $T_{\mathbf{e}}$. By applying this rule, the full output string for the training example in Figure 6.2 becomes

<Agent> `coalition` </Agent><Victim> `civilians[and] woman` </Victim>
<Instrument> `missile` </Instrument><Place> `houses` </Place>.

Since the output string is in the HTML-tag style, we can easily decode the argument and role predictions from the generated output string via a simple rule-based algorithm.

6.4.3 Input Format

As we mentioned previously, the key for the generative formulation for zero-shot cross-lingual EAE is to guide the model to generate output strings in the desired format. To facilitate this behavior, we feed the input passage **x** as well as a *prompt* to X-Gear, as shown by Figure 6.2. The *prompt* contains all valuable information for the model to make predictions, including a trigger **t** and a language-agnostic template $T_{\mathbf{e}}$. Notice that we do not *explicitly* include the event type **e** in the prompt because the template $T_{\mathbf{e}}$ *implicitly* contains this information. In Section 6.6.1, we will show the experiments on explicitly adding event type **e** to the prompt and discuss its influence on the cross-lingual transfer.

6.4.4 Training

To enable X-Gear to generate sentences in different languages, we resort multilingual pre-trained generative model to be our base model, which models the conditional probability of generating a new token given the previous generated tokens and the

input context to the encoder c, i.e,

$$P(x|c) = \prod_i P_{gen}(x_i|x_{<i}, c),$$

where x_i is the output of the decoder at step i.

Copy mechanism. Although the multilingual pre-trained generative models can generate sequences in many languages, solely relying on them may result in generating hallucinating arguments (Li et al., 2021). Since most of the tokens in the target output string appear in the input sequence,[5] we augment the multilingual pre-trained generative models with a copy mechanism to help X-GEAR better adapt to the cross-lingual scenario. Specifically, we follow See et al. (2017) to decide the conditional probability of generating a token t as a weighted sum of the vocabulary distribution computed by multilingual pre-trained generative model P_{gen} and copy distribution P_{copy}

$$P_{\text{X-GEAR}}(x_i = t|x_{<i}, c) = \\ w_{copy} \cdot P_{copy}(t) + (1 - w_{copy}) \cdot P_{gen}(x_i = t|x_{<i}, c)$$

where $w_{copy} \in [0, 1]$ is the copy probability computed by passing the decoder hidden state at time step i to a linear layer. As for P_{copy}, it refers to the probability over input tokens weighted by the cross-attention that the last decoder layer computed (at time step i). Our model is then trained end-to-end with the following loss:

$$\mathcal{L} = -\log \sum_i P_{\text{X-GEAR}}(x_i|x_{<i}, c).$$

6.5 Experiments

6.5.1 Datasets

We consider two commonly used event extraction datasets: ACE-2005 and ERE. We consider English, Arabic, and Chinese annotations for **ACE-2005** (Doddington et al., 2004) and follow the preprocessing in Wadden et al. (2019) to keep 33 event types

[5] Except for the special tokens `[and]` and `[None]`.

Dataset	Lang.	Train			Dev			Test		
		#Sent.	#Event	#Arg.	#Sent.	#Event	#Arg.	#Sent.	#Event	#Arg.
ACE-2005	en	17172	4202	4859	923	450	605	832	403	576
	ar	2722	1743	2506	289	117	174	272	198	287
	zh	6305	2926	5581	486	217	404	482	190	336
ERE	en	14734	6208	8924	1209	525	730	1161	551	882
	es	4582	3131	4415	311	204	279	323	255	354

Table 6.1: Dataset statistics of ACE-2005 and ERE.

and 22 argument roles. **ERE** (Song et al., 2015) is created by the Deep Exploration and Filtering of Test program. We consider its English and Spanish annotations and follow the preprocessing in Lin et al. (2020) to keep 38 event types and 21 argument roles.

Table 6.1 presents the detailed statistics for the ACE-2005 dataset and ERE dataset. For the English and Chinese splits in ACE-2005, we use the setting provided by Wadden et al. (2019) and Lin et al. (2020), respectively. As for Arabic part, we adopt the setup proposed by Xu et al. (2021). Observing that part of the sentence breaks made from Xu et al. (2021) being extremely long for pretrained models to encode, we perform additional preprocessing and postprocessing procedures for Arabic data. Specifically, we split Arabic sentences into several portions that any of the portion is shorter than 80 tokens. Then, we map the models' predictions of the split sentences back to the original sentence during postprocessing.

Notice that prior works working on the zero-shot cross-lingual transfer of event arguments mostly focus on event argument role labeling (Subburathinam et al., 2019; Ahmad et al., 2021b), where they assume ground truth entities are provided during both training and testing. In their experimental data splits, events in a sentence can be scattered in all training, development, and test split since they treat each event-entity pair as a different instance. In this work, we consider event argument extraction (Wang et al., 2019b; Wadden et al., 2019; Lin et al., 2020), which is a more realistic setting.

6.5.2 Evaluation Metric

We follow previous work (Lin et al., 2020; Ahmad et al., 2021b) and consider the *argument classification F1 score* to measure the performance of models. An argument-role pair is counted as correct if both the argument offsets and the role type match the ground truth. Given the ground truth arguments $\mathbf{a}$, ground truth roles $\mathbf{r}$, predicted arguments $\tilde{\mathbf{a}}$, and predicted roles $\tilde{\mathbf{r}}$, the argument classification F1 score is defined as the F1 score between the set $\{(\mathbf{a}_i, \mathbf{r}_i)\}$ and the set $\{(\tilde{\mathbf{a}}_j, \tilde{\mathbf{r}}_j)\}$. For every model, we experiment with three different random seeds and report the average results.

6.5.3 Compared Models

We compare the following models:

- **OneIE** (Lin et al., 2020), the state-of-the-art for monolingual event extraction, is a classification-based model trained with multitasking, including entity extraction, relation extraction, event extraction, and *event argument extraction*. We simply replace its pre-trained embedding with XLM-RoBERTa-large (Conneau et al., 2020a) to fit the zero-shot cross-lingual setting. Note that the multi-task learning makes OneIE require *additional annotations*, such as named entity annotations and relation annotations. We use their provided code[6] to train the model with the provided default settings. It is worth mention that for the Arabic split in the ACE-2005 dataset, OneIE is trained with only entity extraction, event extraction, and event argument extraction since there is no relation labels in Xu et al. (2021)'s preprocessing script. All other parameters are set to the default values.

- **CL-GCN** (Subburathinam et al., 2019) is a classification-based model for cross-lingual event argument role labeling (EARL). It considers *dependency parsing annotations* to bridge different languages and use GCN layers (Kipf and Welling, 2017) to encode the parsing information. We follow the implementation of previous work (Ahmad et al., 2021b) and add two GCN layers on top of XLM-RoBERTa-large. Since CL-GCN focuses on EARL tasks, which assume the ground truth entities are available during testing, we add one name entity recognition module jointly

[6] http://blender.cs.illinois.edu/software/oneie/

trained with CL-GCN. We refer the released code from Ahmad et al. (2021b)[7] to re-implement the CL-GCN method. Specifically, we adapt the baseline framework that described and implemented in OneIE's code (Lin et al., 2020), but we remove its relation extraction module and add two layers of GCN on top of XLM-RoBERTa-large. The pos-tag and dependency parsing annotations are obtained by applying Stanza (Qi et al., 2020). All other parameters are set to the be the same as the training of OneIE.

- **GATE** (Ahmad et al., 2021b), the state-of-the-art model for zero-shot cross-lingual EARL, is a classification-based model which considers *dependency parsing annotations* as well. Unlike CL-GCN, it uses a Transformer layer (Vaswani et al., 2017) with modified attention to encode the parsing information. We follow the original implementation and add two GATE layers on top of pre-trained multilingual language models.[8] Similar to CL-GCN, we add one name entity recognition module jointly trained with GATE. We refer the official released code from Ahmad et al. (2021b) to re-implement GATE. Similar to CL-GCN, we adapt the baseline framework that described and implemented in OneIE's code, but we remove its relation extraction module and add two layers of GATE on top of XLM-RoBERTa-large, mT5, or mBART-50-large. The pos-tag and dependency parsing annotations are also obtained by applying Stanza (Qi et al., 2020). The hyper-parameter of δ in GATE is set to be [2, 2, 4, 4, ∞, ∞, ∞, ∞]. All other parameters are set to the be the same as the training of OneIE.

- **TANL** (Paolini et al., 2021) is a generation-based model for monolingual EAE. Their predicted target is a sentence that embeds labels into the input passage, such as `[Two soldiers|target] were attacked`, which indicates that "*Two soldiers*" is a "*target*" argument. To adapt TANL to zero-shot cross-lingual EAE, we change its pre-trained generative model from T5 (Raffel et al., 2020) to mT5-base (Xue et al., 2021). To adapt TANL to zero-shot cross-lingual EAE, we adapt the public

[7] https://github.com/wasiahmad/GATE

[8] To better compare our method with this strong baseline, we consider three different pre-trained multilingual language models for GATE – (1) XLM-RoBERTa-large (2) mBART-50-large (3) mT5-base. For mBART-50-large and mT-base, we follow BART's recipe (Lewis et al., 2020a) to extract features for EAE predictions. Specifically, the input passage is fed into both encoder and decoder, and the final token representations are elicited from the decoder output.

Model	# of parameters	en ⇓ en	en ⇓ zh	en ⇓ ar	ar ⇓ ar	ar ⇓ en	ar ⇓ zh	zh ⇓ zh	zh ⇓ en	zh ⇓ ar	avg
OneIE (XLM-R-large)	∼570M	63.6	42.5	37.5	57.8	27.5	31.2	69.6	51.5	31.1	45.8
CL-GCN (XLM-R-large)	∼570M	59.8	29.4	25.0	47.5	25.4	19.4	62.2	40.8	23.3	37.0
GATE (XLM-R-large)	∼590M	67.0	49.2	44.5	59.6	27.6	26.3	**70.6**	46.7	**37.3**	47.6
GATE (mBART-50-large)	∼630M	65.5	43.0	38.9	58.5	27.5	26.1	65.9	45.3	30.2	44.5
GATE (mT5-base)	∼590M	59.8	47.7	32.6	45.4	20.7	21.0	64.0	35.3	22.8	38.8
TANL (mT5-base)	∼580M	59.1	38.6	29.7	50.1	18.3	16.9	65.2	33.3	18.3	36.6
X-GEAR (mBART-50-large)	∼610M	68.3	48.9	37.8	59.8	30.5	29.2	63.6	45.9	32.3	46.2
X-GEAR (mT5-base)	∼580M	67.9	53.1	42.0	66.2	27.6	30.5	69.4	52.8	32.0	49.1
X-GEAR (mT5-large)	∼1230M	**71.2**	**54.0**	**44.8**	**68.9**	**32.1**	**33.3**	68.9	**55.8**	33.1	**51.3**

Table 6.2: Average results in argument classification F1(%) of ACE-2005 with three different seeds. The best is in bold and the second best is underlined. "en ⇒ zh" denotes models transferring from en to zh. Compared with models using similar numbers of parameters, X-GEAR (mT5-base) outperforms baselines. To test the influence of using larger pre-trained generative models, we add X-GEAR (mT5-large), which achieves even better results.

code[9] and replace its pre-trained based model T5 (Raffel et al., 2020) with mT5-base (Xue et al., 2021). All other parameters are set to their default values.

- **X-GEAR** is our proposed model. We consider three different pre-trained generative language models: mBART-50-large (Tang et al., 2020), mT5-base, and mT5-large (Xue et al., 2021). We consider three different pre-trained generative language models: mBART-50-large (Tang et al., 2020), mT5-base, and mT5-large (Xue et al., 2021). When fine-tune the pre-trained models, we set the learning rate to 10^{-4} for mT5, and 10^{-5} for mBART-50-large. The batch size is set to 8. The number of training epochs is 60.

6.5.4 Results

Table 6.2 and Table 6.3 list the results on ACE-2005 and ERE, respectively, with all combinations of source languages and target languages. Note that all the models have similar numbers of parameters except for X-GEAR with mT5-large.

[9] https://github.com/amazon-research/tanl

Model	en ⇓ en	en ⇓ es	es ⇓ es	es ⇓ en	avg
OneIE (XLM-R-large)	64.4	56.8	64.8	56.9	60.7
CL-GCN (XLM-R-large)	61.9	51.9	62.9	48.5	55.9
GATE (XLM-R-large)	66.4	**61.5**	63.0	56.5	61.9
TANL (mT5-base)	65.9	40.3	58.6	47.4	53.1
X-GEAR (mBART-50-large)	69.5	57.3	63.9	58.9	62.4
X-GEAR (mT5-base)	<u>69.8</u>	57.9	<u>66.1</u>	<u>59.0</u>	<u>63.2</u>
X-GEAR (mT5-large)	**72.9**	<u>59.7</u>	**67.4**	**64.1**	**66.0**

Table 6.3: Average results in argument classification F1(%) of ERE with three different seeds. The best is in bold and the second best is underlined. "en ⇒ es" denotes that models transfer from en to es.

Comparison to prior generative models. We first observe that TANL has poor performance when transferring to different languages. The reason is that its language-dependent template makes TANL easily generate code-switching outputs,[10] which is a case that pre-trained generative model rarely seen, leading to poor performance. In contrast, X-GEAR considers the language-agnostic templates and achieves better performance for zero-shot cross-lingual transfer.

Comparison to classification models. X-GEAR with mT5-base outperforms OneIE, CL-GCN, and GATE on almost all the combinations of the source language and the target language. This suggests that our proposed method is indeed a promising approach for zero-shot cross-lingual EAE.

It is worth noting that OneIE, CL-GCN, and GATE require an additional pipeline named entity recognition module to make predictions. Moreover, CL-GCN and GATE need additional dependency parsing annotations to align the representations of different languages. On the contrary, X-GEAR is able to leverage the learned knowledge from the pre-trained generative models, and therefore no additional modules or annotations are needed.

[10]Such as the example shown in footnote 2.

Comparison to different pre-trained generative language models. Interestingly, using mT5-base is more effective than using mBART-50-large for X-Gear, although they have a similar amount of parameters. We conjecture that the use of special tokens leads to this difference. mBART-50 has different begin-of-sequence (BOS) tokens for different languages. During generation, we have to specify which BOS token we would like to use as the start token. We guess that this language-specific BOS token makes mBART-50 harder to transfer the knowledge from the source language to the target language. Unlike mBART-50, mT5 does not have such language-specific BOS tokens. During generation, mT5 uses the padding token as the start token to generate a sequence. This design is more general and benefit zero-shot cross-lingual transfer.

Larger pre-trained models are better. Finally, we demonstrate that the performance of X-Gear can be further boosted with a larger pre-trained generative language model. As shown by Table 6.2 and Table 6.3, X-Gear with mT5-large achieves the best scores on most of the cases.

6.6 Analysis

6.6.1 Ablation Studies

Copy mechanism. We first study the effect of the copy mechanism. Table 6.4 lists the performance of X-Gear with and without copy mechanism. It shows improvements in adding a copy mechanism when using mT5-large and mT-base. However, interestingly, adding a copy mechanism is not effective for mBART-50. We conjecture that this is because the pre-trained objective of mBART-50 is denoising autoencoding (Liu et al., 2020b), and it has already learned to copy tokens from the input. Therefore, adding a copy mechanism is less useful. In contrast, the pre-trained objective of mT5 is to only generate tokens been masked out, resulting in lacking the ability to copy input. Thus, the copy mechanism becomes beneficial for mT5.

Including event type in prompts. In Section 6.4, we mentioned that the designed prompt for X-Gear consists of only the input sentence and the language-agnostic

Model	en ⇓ xx	ar ⇓ xx	zh ⇓ xx	xx ⇓ en	xx ⇓ ar	xx ⇓ zh	avg
mBART-50-large	**51.6**	39.8	47.2	48.2	43.2	47.2	46.2
- w/o copy	50.9	**42.2**	**49.6**	**50.6**	**43.5**	**48.7**	**47.6**
mT5-base	**54.3**	**41.4**	**51.4**	**49.4**	**46.7**	**51.0**	**49.1**
- w/o copy	52.1	39.5	47.6	48.1	42.7	48.5	46.4
mT5-large	**56.7**	44.8	**52.6**	**53.0**	**48.9**	52.1	**51.3**
- w/o copy	55.1	**45.0**	51.5	52.0	46.3	**53.2**	50.5

Table 6.4: Ablation study on copy mechanism for ACE-2005. "en ⇒ xx" indicates the average of "en ⇒ en", "en ⇒ zh", and "en ⇒ ar".

template. In this section, we discuss whether *explicitly* including the event type information in the prompt is helpful. We consider three ways to include the event type information:

- **English tokens**. We put the English version of the event type in the prompt even if we are training or testing on non-English languages, for example, using *Attack* for the event type *Attack*.
- **Translated tokens**. For each event type, we prepare the translated version of that event type token. For example, both *Attack* and 攻击 represents the *Attack* event type. During training or testing, we decide the used token(s) according to the language of the input passage. Since all the event types are written in English in ACE-2005 and ERE, we use an off-the-self machine translation tool to perform the translation.
- **Special tokens**. We create a special token for every event type and let the model learn the representations of the special tokens from scratch. For instance, we use `<-attack->` to represent the *Attack* event type.

Table 6.5 shows the results. In most cases, including event type information in the prompt decreases the performance. One reason is that one word in a language can be mapped to several words in another language. For example, the *Life* event type is related to *Marry*, *Divorce*, *Born*, and *Die* four sub-event types. In English, we can use just one word *Life* to cover all four sub-event types. However, In Chinese, when talking about *Marry* and *Divorce*, *Life* should be translated to "生活"; when

Model	en ⇓ xx	ar ⇓ xx	zh ⇓ xx	xx ⇓ en	xx ⇓ ar	xx ⇓ zh	avg
X-GEAR (mT5-base)	**54.3**	**41.4**	51.4	49.4	**46.7**	**51.0**	**49.1**
w/ English Tokens	53.3	39.3	**52.3**	49.2	46.5	49.2	48.3
w/ Translated Tokens	51.7	40.4	52.2	**49.8**	45.6	48.8	48.1
w/ Special Tokens	52.3	39.7	51.8	49.0	45.4	49.3	47.9

Table 6.5: Ablation study on including event type information in prompts for ACE-2005. "en ⇒ xx" indicates the average of "en ⇒ en", "en ⇒ zh", and "en ⇒ ar".

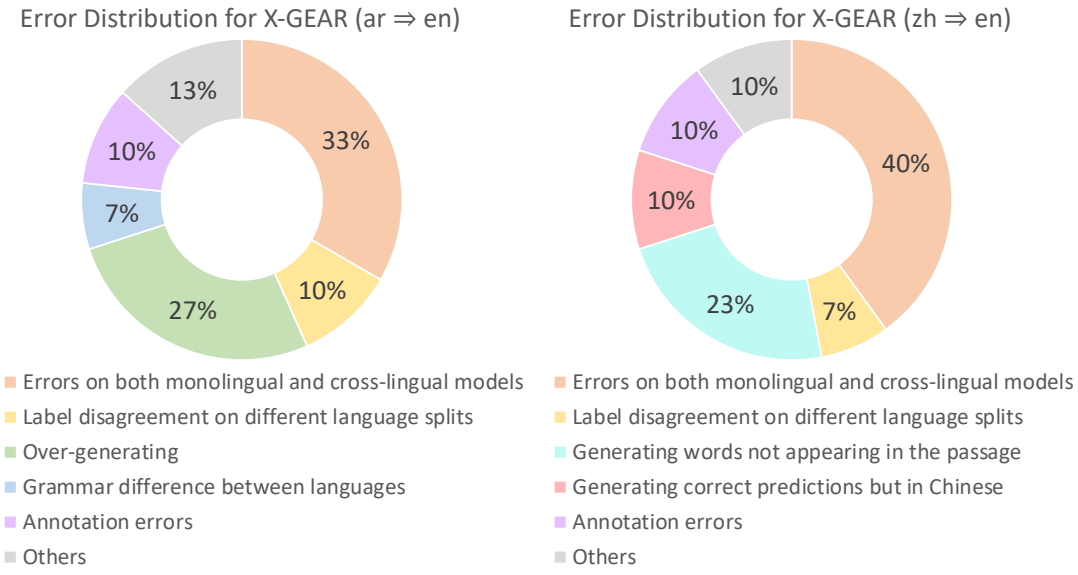


Figure 6.3: Distribution of errors that made by X-GEAR (mT5-base). ***Left:*** The distribution for our model that transfers from Arabic to English; ***Right:*** The distribution for our model trained on Chinese and tested on English.

talking about *Born* and *Die*, *Life* should be translated to "生命". This mismatch may cause the performance drop when considering event types in prompts. We leave how to efficiently use event type information in the cross-lingual setting as future work.

Influence of role order in templates. The order of roles in the designed language-agnostic templates can potentially influence performance. When designing the templates, we intentionally make the order of roles close to the order in natural sen-

Model	en ⇓ xx	ar ⇓ xx	zh ⇓ xx	xx ⇓ en	xx ⇓ ar	xx ⇓ zh	avg
X-GEAR (mT5-base)	54.3	**41.4**	**51.4**	49.4	**46.7**	**51.0**	**49.1**
w/ random order 1	**54.4**	38.9	50.8	48.7	45.1	50.1	48.0
w/ random order 2	52.1	40.4	**51.4**	48.3	45.9	49.7	48.0
w/ random order 3	53.7	40.8	50.7	**50.8**	45.8	48.6	48.4

Table 6.6: Ablation study on different orders of roles in templates for ACE-2005. "en ⇒ xx" indicates the average of "en ⇒ en", "en ⇒ zh", and "en ⇒ ar".

tences.[11] To study the effect of different orders, we train X-GEAR with templates with different random orders and report the results in Table 6.6. X-GEAR with random orders still achieve good performance but slightly worse than the original order. It suggests that X-GEAR is not very sensitive to different templates while providing appropriate order of roles can lead to a small improvement.

Using English tokens instead of special tokens for roles in templates. In Section 6.4, we mentioned that we use language-agnostic templates to facilitate the cross-lingual transfer. To further validate the effectiveness of the language-agnostic template. We conduct experiments using English tokens as the templates. Specifically, we set format

Agent: `[None]` <SEP> Victim: `[None]` <SEP> Instrument: `[None]` <SEP> Place: `[None]`

to be the template for *Life:Die* events. Hence, for non-English instances, the targeted output string is a code-switching sequence. Table 6.7 lists the results. We can observe that applying language-agnostic templates bring X-GEAR 2.3 F1 scores improvements in average.

6.6.2 Error Analysis

We perform error analysis on X-GEAR (mT5-base) when transferring from Arabic to English and transferring from Chinese to English. For each case, we sample 30 failed

[11] For example, types related to subject and object are listed first and types related to methods and places are listed last.

Model	en ⇓ xx	ar ⇓ xx	zh ⇓ xx	xx ⇓ en	xx ⇓ ar	xx ⇓ zh	avg
X-GEAR (mT5-base)	**54.3**	**41.4**	**51.4**	**49.4**	**46.7**	**51.0**	**49.1**
w/ English Tokens	51.4	39.3	49.7	46.6	44.7	49.0	46.8

Table 6.7: Comparison of using English tokens and special tokens for roles in templates. "en ⇒ xx" indicates the average of "en ⇒ en", "en ⇒ zh", and "en ⇒ ar".

examples and present the distribution of various error types in Figure 6.3.

Errors on both monolingual and cross-lingual models. We compare the predicted results from X-GEAR(ar ⇒ en) with X-GEAR(en ⇒ en), or from X-GEAR(zh ⇒ en) with X-GEAR(en ⇒ en). If their predictions are similar and both of them are wrong when compared to the gold output, we classify the error into this category. To overcome the errors in this category, the potential solution is to improve monolingual models for EAE tasks.

Over-generating. Errors in this category happen more often in X-GEAR(ar ⇒ en). It is likely because the entities in Arabic are usually much longer than that in English when measuring by the number of sub-words. Based on our statistics, the average entity span length is 2.85 for Arabic and is 2.00 for English (length of sub-words). This leads to the natural for our X-GEAR(ar ⇒ en) to overly generate some tokens even though they have captured the correct concept. An example is that the model predicts *"The EU foreign ministers"*, while the ground truth is *"ministers"*.

Label disagreement on different language splits. The annotations for the ACE dataset in different language split contain some ambiguity. For example, given sentence *"He now also advocates letting in U.S. troops for a war against Iraq even though it is a fellow Muslim state."* and the queried trigger *"war"*, the annotations in English tends to label *Iraq* as the *Place* where the event happen, while similar situations in other languages will mark *Iraq* as the *Target* for the war.

Grammar difference between languages. An example for this category is *"... Blackstone Group would buy Vivendi's theme park division, including Universal Stu-*

dios Hollywood ...” and the queried trigger *“buy”*. We observe that X-GEAR(ar ⇒ en) predicts *Videndi* as the *Artifact* been sold and *division* is the *Seller*, while X-GEAR(en ⇒ en) can correctly understand that *Videndi* are the *Seller* and *division* is the *Artifact*. We hypothesize the reason being the differences between the grammar in Arabic and English. The word order of the sentence *“Vivendi’s theme park division”* in Arabic is reversed with its English counterpart, that is, *“theme park division”* will be written before *“Vivendi”* in Arabic. Such difference leads to errors in this category.

Generating words not appearing in the passage. In X-GEAR(zh ⇒ en), we observe several cases that generate words not appearing in the passage. There are two typical situations. The first case is that X-GEAR(zh ⇒ en) mixes up singular and plural nouns. For example, the model generates *“studios”* as prediction while only *“studio”* appears in the passage. This may be because Chinese does not have morphological inflection for plural nouns. The second case is that X-GEAR(zh ⇒ en) will generate random predictions in Chinese.

Generating correct predictions but in Chinese. This is a special case of *“Generating words not appearing in the passage”*. In this category, we observe that although the prediction is in Chinese (hence, a wrong prediction), it is correct if we translate the prediction into English.

6.6.3 Constrained Decoding

Among all the errors, we highlight two specific categories — *“Generating words not appearing in the passage”* and *“Generating correct predictions but in Chinese”*. These errors can be resolved by applying constrained decoding (Cao et al., 2022) to force all the generated tokens to appear input.

Table 6.8 presents the result of X-GEAR with constrained decoding. We observe that adapting such constraints indeed helps the cross-lingual transferability, yet it also hurts the performance in some monolingual cases. We conduct a qualitative inspection of the predictions. The observation is that constrained decoding algorithm although guarantees all generated tokens appearing in the input, the coercive method breaks the overall sequence distribution that learned. Hence, in many monolingual

Model	en ⇓ en	en ⇓ zh	en ⇓ ar	ar ⇓ ar	ar ⇓ en	ar ⇓ zh	zh ⇓ zh	zh ⇓ en	zh ⇓ ar	avg (mono.)	avg (cross.)	avg (all)
X-Gear (mBART-50-large)	**68.3**	48.9	37.7	**59.8**	30.5	29.2	**63.6**	45.9	**32.3**	**63.9**	37.4	**46.2**
w/ constrained decoding	68.0	**49.1**	**37.8**	59.5	**30.6**	29.2	59.7	**47.7**	31.3	62.4	**37.6**	45.9
X-Gear (mT5-base)	67.9	53.1	42.0	66.2	27.6	**30.5**	**69.4**	52.8	32.0	**67.8**	39.7	**49.1**
w/ constrained decoding	67.9	53.1	42.0	66.2	**27.8**	30.4	66.7	**53.1**	**33.1**	67.0	**39.9**	48.9
X-Gear (mT5-large)	71.2	54.0	44.8	68.9	**32.1**	33.3	**68.9**	55.8	33.1	**69.7**	42.2	51.3
w/ constrained decoding	71.2	**54.8**	**45.6**	68.9	32.0	33.3	66.2	**57.7**	**35.0**	68.8	**43.1**	**51.6**

Table 6.8: Results of applying constrained decoding. The avg(mono.) column represents the results that average over values in *en* ⇒ *en*, *zh* ⇒ *zh*, and *ar* ⇒ *ar*. The avg(cross.) column represents the results that average over values in *en* ⇒ *zh*, *en* ⇒ *ar*, *zh* ⇒ *en*, *zh* ⇒ *ar*, *ar* ⇒ *en*, and *ar* ⇒ *zh*.

examples, once one of the tokens is corrected by constrained decoding, its following generated sequence changes a lot, while the original predicted suffixed sequence using beam decoding are actually correct. This leads to a performance decrease.[12]

6.7 Summary

We present the first generation-based models for zero-shot cross-lingual event argument extraction. To overcome the discrepancy between languages, we design language-agnostic templates and propose X-Gear, which well capture output dependencies and can be used without additional named entity extraction modules. Our experimental results show that X-Gear outperforms the current state-of-the-art, which demonstrates the potential of using a language generation framework to solve zero-shot cross-lingual structured prediction tasks.

[12]Indeed, a similar situation happens to cross-lingual cases; however, since the original performance for cross-lingual transfer is not high enough, the benefits of correcting tokens are more significant than this drawback.

Part III

Conclusion

CHAPTER 7

Conclusion and Future Directions

7.1 Summary of Contributions

In this dissertation, we improve different levels of the robustness of NLP models by enhancing their understanding of semantically equivalent texts. We conclude the contributions as follows.

In Chapter 2, we propose to improve the syntax-level robustness of NLP models by encoding a text into two separate embeddings: semantic embedding and syntactic embedding. We show that the two disentangled embeddings can be learned with the help of annotated paraphrase pairs and a carefully designed adversarial loss. Our experimental results demonstrate that the separated semantic embedding is less sensitive to syntax and results in better performance on the semantic textual similarity task.

In Chapter 3, we consider unannotated texts, instead of annotated paraphrase pairs, to learn the disentanglement of semantics and syntax. Our proposed Reconstruction objective encourages the model to extract semantics from bag-of-words and capture the syntactic features from constituency parse trees. The proposed model is less affected by the syntactic perturbations to the input text and therefore achieves better performance on the syntactically controlled paraphrase generation task. In addition, the generated paraphrases can be used for data augmentation to improve the syntax-level robustness of NLP models.

Chapter 4 further improves the quality of disentanglement by introducing abstract meaning representation (AMR). Compared to the bag-of-words method in Chapter 3, AMR graphs persevere more order information, which is helpful for models to capture the semantics of texts. By leveraging AMR graphs, we achieve better performance on

the syntactically controlled paraphrase generation task and the syntax-level robustness test.

In Chapter 5, we focus on improving the language-level robustness of NLP models by studying the zero-shot cross-lingual transfer. We point out the relation between the failure cases of zero-shot cross-lingual transfer and adversarial attacks, and introduce two robust training techniques, randomized smoothing and adversarial training, to improve the performance of zero-shot cross-lingual transfer. The robust training can handle the imperfect alignment between different languages in the embedding space, leading to promising results on two zero-shot cross-lingual text classification tasks.

In Chapter 6, we study zero-shot cross-lingual event argument extraction. We propose generation-based models that create a language-agnostic output space and map knowledge from different languages to the same space. Compared to the traditional classification-based models, generation-based models are less sensitive to the input languages, resulting in better performance on zero-shot cross-lingual event argument extraction.

7.2 Future Directions

Structure-free pre-trained text representations. Our research work reveals that the existing text representations are too sensitive to syntax. In Chapter 2, 3, and 4, we have shown that separating semantics and syntax is one of the solutions to this robustness issue. However, to fundamentally resolve this issue, we will need a new structure to represent texts rather than the current contextualized text representations.

Figure 7.1 illustrates a potential new structure to represent texts. Instead of encoding a text into a list of vectors corresponding to each token, I suggest encoding a text into a graph, where each node in the graph denotes a concept in the text while each edge describes the relations between concept nodes. I believe such graph representations are more similar to how humans understand a text, where we first extract concepts from a text and figure out their relations. In addition, since the graph representations are based on extracted concepts, no matter what syntax the input text is, we can always similar graphs for semantically equivalent texts. The

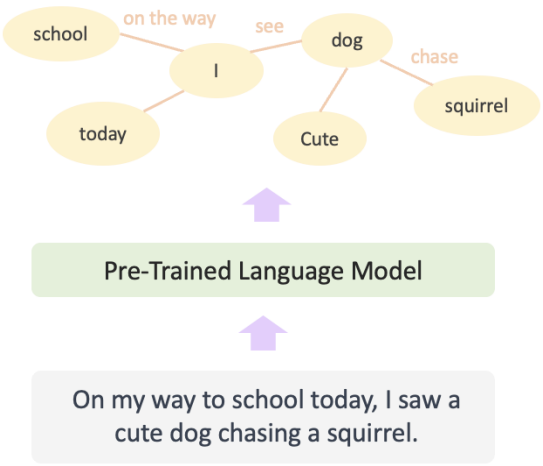


Figure 7.1: An example of structure-free text representations.

NLP models trained on such graph representations can therefore be more robust and reliable. For this research direction, we may study the following questions: (1) How to extract concepts and construct graphs? (2) What resources do we need? (3) How to scale up the training process for pre-training?

Code-switching texts. Existing work focusing on language-level robustness usually considers texts written in only one language, even for zero-shot cross-lingual transfer. In Chapter 5, we observe that NLP models have very unstable performance when the input texts contain two or more languages. However, texts written in mixed languages become more and more common in recent applications. For instance, the texts that appear in tweets are usually code-switching. Another example is the commands used for smart assistants. There will be a lot of special terms and names that cannot be expressed in a single language. How to leverage the knowledge of existing pre-trained multilingual language models to handle code-switching texts has attracted more and more attention in recent years. Our robust training techniques can be a potential solution to address this challenge.

Bibliography

Eneko Agirre, Daniel M. Cer, Mona T. Diab, and Aitor Gonzalez-Agirre. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, 2012.

Eneko Agirre, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. *sem 2013 shared task: Semantic textual similarity. In Mona T. Diab, Timothy Baldwin, and Marco Baroni, editors, *Proceedings of the Second Joint Conference on Lexical and Computational Semantics, *SEM 2013*, 2013.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING*, 2014.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, 2015.

Eneko Agirre, Carmen Banea, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, 2016.

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Kai-Wei Chang, and Nanyun Peng. Cross-lingual dependency parsing with unlabeled auxiliary languages. In *The 2019 SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, 2019a.

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard H. Hovy, Kai-Wei Chang, and Nanyun Peng. 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 (NAACL-HLT)*, 2019b.

Wasi Uddin Ahmad, Haoran Li, Kai-Wei Chang, and Yashar Mehdad. Syntax-augmented multilingual BERT for cross-lingual transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021a.

Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. GATE: graph attention transformer encoder for cross-lingual relation and event extraction. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, 2021b.

Hanan Aldarmaki and Mona T. Diab. Context-aware cross-lingual mapping. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J. Martin, and Mark O. Riedl. Story realization: Expanding plot events into sentences. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Mikel Artetxe and Holger Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Trans. Assoc. Comput. Linguistics*, 7: 597–610, 2019.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Xuefeng Bai, Yulong Chen, Linfeng Song, and Yue Zhang. Semantic representation for dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, 2021.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse (LAW-ID@ACL)*, 2013.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xin-Yu Dai, and Jiajun Chen. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

Regina Barzilay and Lillian Lee. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2003.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Igor A. Bolshakov and Alexander F. Gelbukh. Synonymous paraphrasing using wordnet and internet. In *Proceedings of the 9th International Conference on Applications of Natural Languages to Information Systems (NLDB)*, 2004.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, 2016.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*, 2023.

Deng Cai and Wai Lam. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Nicola De Cao, Ledell Wu, Kashyap Popat, Mikel Artetxe, Naman Goyal, Mikhail Plekhanov, Luke Zettlemoyer, Nicola Cancedda, Sebastian Riedel, and Fabio Petroni. Multilingual autoregressive entity linking. *Trans. Assoc. Comput. Linguistics*, 10:274–290, 2022.

Steven Cao, Nikita Kitaev, and Dan Klein. Multilingual alignment of contextual word representations. In *8th International Conference on Learning Representations (ICLR)*, 2020.

Yue Cao and Xiaojun Wan. Divgan: Towards diverse paraphrase generation via diversified generative adversarial network. In *Findings of the Association for Computational Linguistics: EMNLP*, 2020.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. Joint copying and restricted generation for paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. Universal sentence encoder for english. In *Proceedings of the*

2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL*, 2017.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. In *Proceedings of 15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2014.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. A multi-task approach for disentangling syntax and semantics in sentence representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019a.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019b.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

Jishnu Ray Chowdhury, Yong Zhuang, and Shuyi Wang. Novelty controlled paraphrase generation with retrieval augmented conditional prompt tuning. In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

Jonathan H. Clark, Jennimaria Palomaki, Vitaly Nikolaev, Eunsol Choi, Dan Garrette, Michael Collins, and Tom Kwiatkowski. Tydi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Trans. Assoc. Comput. Linguistics*, 8:454–470, 2020.

Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, 2018.

Alexis Conneau and Guillaume Lample. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019 (NeurIPS)*, 2019.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single \$&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018a.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. XNLI: evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018b.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020a.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020b.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. Style transformer: Unpaired text style transfer without disentangled latent representation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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 (NAACL-HLT)*, 2019.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. Weischedel. The automatic content extraction (ACE) program - tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*, 2004.

Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, 2004.

Xinshuai Dong, Anh Tuan Luu, Rongrong Ji, and Hong Liu. Towards robustness against natural language word substitutions. In *9th International Conference on Learning Representations (ICLR)*, 2021.

Zi-Yi Dou and Graham Neubig. 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 (EACL)*, 2021.

Xinya Du, Alexander M. Rush, and Claire Cardie. GRIT: generative role-filler transformers for document-level event entity extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Philipp Dufter and Hinrich Schütze. Identifying necessary elements for bert's multilinguality. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

Bora Edizel, Aleksandra Piktus, Piotr Bojanowski, Rui Ferreira, Edouard Grave, and Fabrizio Silvestri. Misspelling oblivious word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

Elozino Egonmwan and Yllias Chali. Transformer and seq2seq model for paraphrase generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation@EMNLP-IJCNLP*, 2019.

Angela Fan, Mike Lewis, and Yann N. Dauphin. Strategies for structuring story generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.

Steven Fincke, Shantanu Agarwal, Scott Miller, and Elizabeth Boschee. Language model priming for cross-lingual event extraction. In *Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

Yao Fu, Yansong Feng, and John P. Cunningham. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019 (NeurIPS)*, 2019.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. Style transfer in text: Exploration and evaluation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

Silin Gao, Yichi Zhang, Zhijian Ou, and Zhou Yu. Paraphrase augmented task-oriented dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Siddhant Garg and Goutham Ramakrishnan. BAE: bert-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Yotam Gil, Yoav Chai, Or Gorodissky, and Jonathan Berant. White-to-black: Efficient distillation of black-box adversarial attacks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

Yoav Goldberg. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*, 2019.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph M. Weischedel, and Nanyun Peng. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations (ICLR)*, 2015.

Naman Goyal, Jingfei Du, Myle Ott, Giri Anantharaman, and Alexis Conneau. Larger-scale transformers for multilingual masked language modeling. In *Proceedings of the 6th Workshop on Representation Learning for NLP, RepL4NLP@ACL-IJCNLP*, 2021.

Tanya Goyal and Greg Durrett. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. A deep generative framework for paraphrase generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. DEGREE: A data-efficient generation-based event extraction model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2022.

I-Hung Hsu, Kuan-Hao Huang, Shuning Zhang, Wenxin Cheng, Premkumar Natarajan, Kai-Wei Chang, and Nanyun Peng. TAGPRIME: A unified framework for relational structure extraction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023a.

I-Hung Hsu, Zhiyu Xie, Kuan-Hao Huang, Premkumar Natarajan, and Nanyun Peng. AMPERE: Amr-aware prefix for generation-based event argument extraction model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023b.

J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. PARABANK: monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Kuan-Hao Huang and Kai-Wei Chang. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Kuan-Hao Huang, Chen Li, and Kai-Wei Chang. Generating sports news from live commentary: A chinese dataset for sports game summarization. In *Proceedings of the 2020 Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (AACL)*, 2020.

Kuan-Hao Huang, Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. Improving zero-shot cross-lingual transfer learning via robust training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021a.

Kuan-Hao Huang, Varun Iyer, I-Hung Hsu, Anoop Kumar, Kai-Wei Chang, and Aram Galstyan. ParaAMR: A large-scale syntactically diverse paraphrase dataset by amr back-translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Kung-Hsiang Huang, Sam Tang, and Nanyun Peng. Document-level entity-based extraction as template generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021b.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. Achieving verified robustness to symbol substitutions via interval bound propagation. 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)*, 2019a.

Xiaolei Huang, Jonathan May, and Nanyun Peng. What matters for neural cross-lingual named entity recognition: An empirical analysis. In *2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), short*, 2019b.

Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. First quora dataset release: Question pairs. *data.quora.com*, 2017.

Mohit Iyyer, Varun Manjunatha, Jordan L. Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2018.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. Certified robustness to adversarial word substitutions. 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)*, 2019.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. Robust encodings: A framework for combating adversarial typos. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. Cross-lingual ability of multilingual BERT: an empirical study. In *8th International Conference on Learning Representations (ICLR)*, 2020.

Jerrold J Katz and Jerry A Fodor. The structure of a semantic theory. *language*, 39 (2):170–210, 1963.

David Kauchak and Regina Barzilay. Paraphrasing for automatic evaluation. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2006.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations (ICLR)*, 2017.

Stanley Kok and Chris Brockett. Hitting the right paraphrases in good time. In *Proceedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2010.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. Neural AMR: sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

Mikhail Kozhevnikov and Ivan Titov. Cross-lingual transfer of semantic role labeling models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2013.

Ashutosh Kumar, Satwik Bhattamishra, Manik Bhandari, and Partha P. Talukdar. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha P. Talukdar. Syntax-guided controlled generation of paraphrases. *Trans. Assoc. Comput. Linguistics*, 8:330–345, 2020.

Anne Lauscher, Vinit Ravishankar, Ivan Vulic, and Goran Glavas. From zero to hero: On the limitations of zero-shot language transfer with multilingual transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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, ACL 2020, Online, July 5-10, 2020*, 2020a.

Patrick S. H. Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. MLQA: evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020b.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020a.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. BERT-ATTACK: adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020b.

Sha Li, Heng Ji, and Jiawei Han. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. Decomposable neural paraphrase generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang Liu, Fan Yang, Daniel Campos, Rangan Majumder, and Ming Zhou. 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)*, 2020.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Zhe Lin and Xiaojun Wan. Pushing paraphrase away from original sentence: A multi-round paraphrase generation approach. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*, 2021.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori S. Levin. 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 (EACL)*, 2017.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 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)*, 2019a.

Mingtong Liu, Erguang Yang, Deyi Xiong, Yujie Zhang, Yao Meng, Changjian Hu, Jinan Xu, and Yufeng Chen. A learning-exploring method to generate diverse paraphrases with multi-objective deep reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, 2020a.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9):195:1–195:35, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXic:1907.11692*, 2019b.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Trans. Assoc. Comput. Linguistics*, 8:726–742, 2020b.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, 2020c.

Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, 2018.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, 2021.

Nitin Madnani, Joel R. Tetreault, and Martin Chodorow. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2012.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations(ICLR)*, 2018.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing

toolkit. In *Proceedings of the 52th Conference of the Association for Computational Linguistics (ACL)*, 2014.

Kathleen R. McKeown. Paraphrasing questions using given and new information. *Am. J. Comput. Linguistics*, 9(1):1–10, 1983.

Tao Meng, Nanyun Peng, and Kai-Wei Chang. 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)*, 2019.

Tomás Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina Maria Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. Counter-fitting word vectors to linguistic constraints. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2016.

Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. Semeval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL*, 2017.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook fair's WMT19 news translation task submission. In *WMT*, 2019.

Minh Van Nguyen and Thien Huu Nguyen. Improving cross-lingual transfer for event argument extraction with language-universal sentence structures. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, 2021.

Jian Ni and Radu Florian. Neural cross-lingual relation extraction based on bilingual word embedding mapping. 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)*, 2019.

Lin Pan, Chung-Wei Hang, Haode Qi, Abhishek Shah, Mo Yu, and Saloni Potdar. 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 (NAACL-HLT*, 2021.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations (ICLR)*, 2021.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002.

Tanmay Parekh, I-Hung Hsu, Kuan-Hao Huang, Kai-Wei Chang, and Nanyun Peng. GENEVA: Benchmarking generalizability for event argument extraction with hundreds of event types and argument roles. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Hao Peng, Ankur P. Parikh, Manaal Faruqui, Bhuwan Dhingra, and Dipanjan Das. Text generation with exemplar-based adaptive decoding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

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

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2018.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek V. Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING)*, 2016.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A python natural language processing toolkit for many human languages. In Asli Celikyilmaz and Tsung-Hsien Wen, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL)*, 2020.

Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. In *Proceedings of the*

Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI), 2020.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 2020.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-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)*, 2019.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, 2021.

Alexey Romanov, Anna Rumshisky, Anna Rogers, and David Donahue. Adversarial decomposition of text representation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

Aurko Roy and David Grangier. Unsupervised paraphrasing without translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, and Melvin Johnson. XTREME-R: towards more challenging and nuanced multilingual evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Mrinmaya Sachan and Eric P. Xing. Machine comprehension using rich semantic representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

Teven Le Scao and Alexander M. Rush. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

Timo Schick and Hinrich Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter*

of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), 2019.

Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, 2017.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Conference of the Association for Computational Linguistics (ACL)*, 2016.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, 2017.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. Towards controllable biases in language generation. In *Findings of the Association for Computational Linguistics: EMNLP*, 2020.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. "nice try, kiddo": Investigating ad hominems in dialogue responses. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. Robustness verification for transformers. In *8th International Conference on Learning Representations (ICLR)*, 2020.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *5th International Conference on Learning Representations (ICLR)*, 2017.

Zhiyi Song, Ann Bies, Stephanie M. Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. From light to rich ERE: annotation of entities, relations, and events. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation, (EVENTS@HLP-NAACL)*, 2015.

Ananya Subburathinam, Di Lu, Heng Ji, Jonathan May, Shih-Fu Chang, Avirup Sil, and Clare R. Voss. 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)*, 2019.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J. Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. In *6th International Conference on Learning Representations (ICLR)*, 2018.

Jiao Sun, Xuezhe Ma, and Nanyun Peng. AESOP: paraphrase generation with adaptive syntactic control. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. Controllable neural story plot generation via reward shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

Samson Tan, Shafiq R. Joty, Min-Yen Kan, and Richard Socher. It's morphin' time! combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Conference of the Association for Computational Linguistics (ACL)*, 2020.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*, 2020.

Yufei Tian, Arvind Krishna Sridhar, and Nanyun Peng. Hypogen: Hyperbole generation with commonsense and counterfactual knowledge. In *Findings of the Association for Computational Linguistics: EMNLP*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, 2017.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. Entity, relation, and event extraction with contextualized span 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)*, 2019.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing NLP. 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), 2019.

Yixin Wan, Kuan-Hao Huang, and Kai-Wei Chang. PIP: Parse-instructed prefix for syntactically controlled paraphrase generation. In *Findings of the Association for Computational Linguistics: ACL 2023 (ACL-Findings)*, 2023.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019a.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. HMEAE: hierarchical modular event argument 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)*, 2019b.

Xiangpeng Wei, Yue Hu, Rongxiang Weng, Luxi Xing, Heng Yu, and Weihua Luo. On learning universal representations across languages. In *9th International Conference on Learning Representations (ICLR)*, 2021.

John Wieting and Kevin Gimpel. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Conference of the Association for Computational Linguistics (ACL)*, 2018.

John Wieting, Jonathan Mallinson, and Kevin Gimpel. Learning paraphrastic sentence embeddings from back-translated bitext. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

John Wieting, Kevin Gimpel, Graham Neubig, and Taylor Berg-Kirkpatrick. Simple and effective paraphrastic similarity from parallel translations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019.

John Wieting, Graham Neubig, and Taylor Berg-Kirkpatrick. A bilingual generative transformer for semantic sentence embedding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Yorick Wilks. An intelligent analyzer and understander of english. *Commun. ACM*, 18(5):264–274, 1975.

Genta Indra Winata, Andrea Madotto, Zhaojiang Lin, Rosanne Liu, Jason Yosinski, and Pascale Fung. Language models are few-shot multilingual learners. *arXiv preprint arXiv:2109.07684*, 2021.

Haoran Xu, Seth Ebner, Mahsa Yarmohammadi, Aaron Steven White, Benjamin Van Durme, and Kenton W. Murray. Gradual fine-tuning for low-resource domain adaptation. *arXiv preprint arXiv:2103.02205*, 2021.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. A unified generative framework for various NER subtasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, 2021.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. Docchat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, (EMNLP-IJCNLP)*, 2019.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

Mao Ye, Chengyue Gong, and Qiang Liu. SAFER: A structure-free approach for certified robustness to adversarial word substitutions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension. In *6th International Conference on Learning Representations (ICLR)*, 2018.

Xinyuan Zhang, Yi Yang, Siyang Yuan, Dinghan Shen, and Lawrence Carin. Syntax-infused variational autoencoder for text generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, 2019a.

Yue Zhang, Rui Wang, and Luo Si. 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)*, 2019b.

Zixuan Zhang, Nikolaus Nova Parulian, Heng Ji, Ahmed Elsayed, Skatje Myers, and Martha Palmer. Fine-grained information extraction from biomedical literature based on knowledge-enriched abstract meaning representation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th*

International Joint Conference on Natural Language Processing (ACL/IJCNLP), 2021.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Pei Zhou, Weijia Shi, Jieyu Zhao, Kuan-Hao Huang, Muhao Chen, Ryan Cotterell, and Kai-Wei Chang. Examining gender bias in languages with grammatical gender. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Qiji Zhou, Yue Zhang, Donghong Ji, and Hao Tang. AMR parsing with latent structural information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Yi Zhou, Xiaoqing Zheng, Cho-Jui Hsieh, Kai-Wei Chang, and Xuanjing Huang. Defense against synonym substitution-based adversarial attacks via dirichlet neighborhood ensemble. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Bowei Zou, Zengzhuang Xu, Yu Hong, and Guodong Zhou. Adversarial feature adaptation for cross-lingual relation classification. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, 2018.