Exploring Language Structured Prediction in Resource-limited Scenarios

Zhisong Zhang

CMU-LTI-23-005

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15123
www.lti.cs.cmu.edu

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies.*

Keywords: Structured Prediction, Transfer Learning, Active Learning

Abstract

In natural language processing (NLP), many tasks involve *structured prediction*: predicting structured outputs consisting of a group of interdependent variables. This allows extracting useful information from unstructured raw texts, which can benefit downstream tasks and analyses for both humans and machines. To obtain automatic models, the main paradigm is in a data-driven supervised-learning fashion. In this paradigm, the main bottleneck is the availability of manually annotated data, which is usually expensive and time-consuming to obtain. Moreover, we usually would like to extend the models to various new scenarios, like in different domains or languages. The model performance could drop dramatically if the training instances are insufficient to cover the target scenarios, while it is costly and inefficient to annotate large amounts of data instances in all these new cases.

To mitigate this problem and ease the reliance of structured prediction models on large amounts of annotations, we need to consider both aspects of the model and the data, which are the main driving forces of data-driven machine learning. Related to these core aspects, we examine three directions. Firstly, we investigate **structured modeling** in *model design*, which involves how the complex structured outputs are modeled and predicted. This is especially important for structured prediction tasks, which usually have large output spaces. Moreover, on the interaction of *model and data*, we examine **transfer learning** where related data is utilized to help low-resource target tasks. In this case, how to design models that are more agnostic to the discrepancies between the source and target data resources is also crucial for the success of the transfer. Finally, we explore **active learning**, with a specific focus on the *data* itself. When resources are limited, it is difficult to obtain a large amount of annotated instances, but annotating a small set can be feasible. With a strategy to select an informative set of instances, much fewer manual annotations may be required to achieve satisfactory performance.

This thesis consists of three parts, corresponding to these three directions. In the first part, we investigate the influence of structured output modeling in deep neural models. We find that structured modeling brings benefits on sentence-level complete matches and with more efficient models. We further extend the analyses to low-resource scenarios and investigate the interactions of structural constraints and training data sizes. In the second part, we investigate a series of related structured tasks and find that supervision from related data, such as those from the same task but in different languages (cross-lingual learning) and those from related tasks (multi-task learning), can be beneficial, especially if utilizing models that care less about the source and target differences. Finally, in the third part, we perform a systematic investigation of active learning for structured prediction in NLP. Especially, we analyze the effectiveness of annotating and learning with partial structures, which can improve data efficiency for active learning. Moreover, we show that combining active learning with self-training with the unlabeled instances from the active learning data pool can bring further improvements.

Acknowledgments

Time flies fast. I still remember the date when I arrived in Pittsburgh, knowing nothing, looking around curiously in the new environment, and wondering what the future would be. Now five years have passed, and I might still know little, but when looking back, I simply realize how fortunate I've been. I'm clumsy at expressing myself, but I would like to express my deep gratitude to the people who helped me reach the end of this wonderful journey.

First, I would like to express my greatest thanks to my advisors, Eduard Hovy and Emma Strubell. I'm really fortunate to be advised by them during my PhD studies and I've learned a lot from them. I'm especially thankful that they allow me to freely explore the topics that I'm interested in and provide valuable feedback. During the PhD journey, there are many times when I slightly felt lost in my research, but talking with them will always make things much clearer. In addition to research, I'm also thankful for their care of my happiness and feelings, which has helped me a lot, especially during difficult times. It is their support and encouragement that enable me to finish this delightful research journey of PhD study. I would also like to thank my committee members, Graham Neubig and Martha Palmer, for providing helpful feedback and suggestions. I enjoy discussions with them on exploring structures with recent models and considering linguistic aspects.

LTI is such a wonderful place for studying and performing research, and there are lots of people to whom I want to express my thankfulness. I would like to thank the members of the Edvisees group, the SLAB group, and the COMEDY group; I'm fortunate to be in such great environments and I'm grateful for all the help and support from them: Xuezhe Ma, Zhengzhong Liu, Xiang Kong, Dongyeop Kang, Varun Gangal, Naoki Otani, Hwijeen Ahn, Sanket Vaibhav Mehta, Jared Fernandez, Jeremiah Milbauer, Clara Na, and Nupoor Gandhi. The discussions in the group meetings and the chatting in the gatherings will always be my precious memories. I would like to thank Lori Levin and David Mortensen, from whom I learned a lot of linguistic knowledge. I would like to thank the members of the OPERA team, especially Yonatan Bisk and Teruko Mitamura, for their supervision of my work in OPERA. I gained valuable experience from this project. I would also like to thank Kate Schaich and Stacey Young for their great administrative assistance.

I would also like to thank my collaborators: Xuezhe Ma, Wasi Ahmad, Kai-Wei Chang, Nanyun Peng, Junxian He, Junpei Zhou, Zecong Hu, Xiang Kong, Zhengzhong Liu, Yizhe Zhang, and Bill Dolan. Collaboration is an important lesson that I learned during my graduate study and it is my collaborators' help that enables me to finish my work during this PhD journey.

Finally, my deepest gratitude to my parents, Y. Hu and X. Zhang, for their unconditional love and support, to whom this thesis is dedicated.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In natural language processing (NLP), we usually need to extract structured information from unstructured texts. This process is crucial for better accessing and representing the information available in text and can be helpful in further downstream applications and analyses for both humans and machines. Moreover, proper analysis of language structures is usually a requirement for basic natural language understanding. For example, to realize the meaning of the sentence "The student wrote a paper at Carnegie Mellon University", we need to figure out several layers of structures, some of which are shown in Figure 1.1 (automatically processed by CoreNLP (Manning et al., 2014) and AllenNLP (Gardner et al., 2018)):

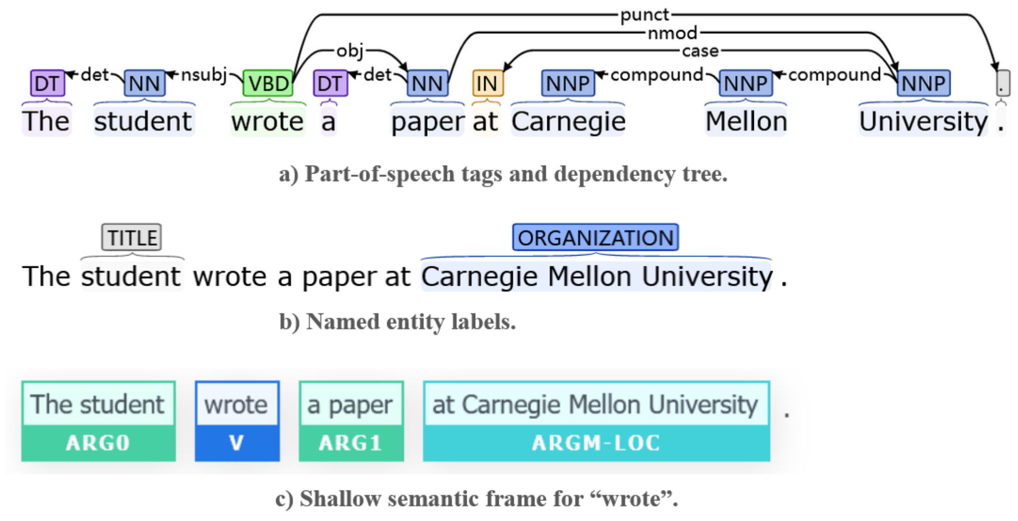

DT
det
NN
nsubj
VBD
obj
DT
det
NN
punct
nmod
case
IN
NNP
compound
NNP
compound
NNP
.
The student wrote a paper at Carnegie Mellon University .
a) Part-of-speech tags and dependency tree.
TITLE
ORGANIZATION
The student wrote a paper at Carnegie Mellon University .
b) Named entity labels.
The student
ARG0
wrote
V
a paper
ARG1
at Carnegie Mellon University
ARGM-LOC
.
c) Shallow semantic frame for "wrote".

Figure 1.1: Example structures for an English sentence.

- Each word in the sentence carries certain *syntactic* functionalities, such as lexical categories which can be reflected by part-of-speech (POS) tags. For example, "wrote" is the main verb, and "student" and "paper" are nouns.
- The words and phrases also carry *semantic* meanings and can refer to entities. For example, "student" is a person, and "paper" is an object. Moreover, the multi-word expression

"Carnegie Mellon University" is a named entity of an organization.

- Going beyond individual words and phrases, each sentence has its *syntactic* structure, connecting the sub-units of words and phrases with syntactic relations. For example, "student" is the subject (nsubj) of the main verb "wrote" while "paper" is its object (obj).
- As the underlying driving force, semantics also involves relational structures. Considering shallow semantics, the "student" is the agent (ARG0) of the writing action, and the "paper" is the theme (ARG1) that is written.

All these tasks above involve *structured prediction* (BakIr et al., 2007), where the system needs to output a structured object consisting of a group of inter-dependent variables, such as a sequence of POS or entity labels, a syntactic tree or a semantic graph.

Structured prediction has been an important and fundamental topic for natural language processing (NLP) (Smith, 2011) and many NLP tasks can be cast as or already involve the prediction of structured outputs. For example, sequence labeling, the most fundamental structured prediction formalism, can be applied to a wide range of NLP tasks, including linguistic tasks like POS tagging (Ratnaparkhi, 1996; Lafferty et al., 2001) and chunking (Tjong Kim Sang and Buchholz, 2000), information extraction tasks such as named entity recognition (NER) (McCallum and Li, 2003) and entity and event mention detection (Lin et al., 2020) as well as slot filling in spoken language understanding (Raymond and Riccardi, 2007; Yao et al., 2014). Beyond tagging, predicting more complex structures like trees and graphs is also required for many tasks, like syntactic parsing (Kübler et al., 2009), semantic role labeling (Gildea and Jurafsky, 2002), and meaning representation graph parsing (Koller et al., 2019). Moreover, natural language generation tasks like machine translation and summarization can be also modeled as structured prediction problems (Edunov et al., 2018). Therefore, more effective structured prediction approaches could benefit a variety of NLP applications.

1.1 Motivation

Recently, the field of NLP has witnessed a wave of successful applications of neural networks (Goodfellow et al., 2016), which bring improvements to a variety of NLP tasks (Bengio et al., 2003; Collobert et al., 2011; Sutskever et al., 2014; Chen and Manning, 2014; Ma and Hovy, 2016; He et al., 2017; Vaswani et al., 2017). Especially, adopting large-scale unlabeled corpora to pre-train neural models, such as those for static word embeddings (Mikolov et al., 2013a,b; Pennington et al., 2014; Bojanowski et al., 2017) and contextualized word representations (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019), has been a crucial step for the success. Structured prediction tasks themselves also benefit much from the utilization of deep learning. For example, the task of dependency parsing has been continuously benefiting from better neural contextualized encoders (Kiperwasser and Goldberg, 2016), structured decoding modules (Dozat and Manning, 2017) as well as pre-trained contextualized representations (Kulmizev et al., 2019).

More recently, the developments of large language models (LLMs) have further revolutionized the field of NLP. Especially, models such as ChatGPT[1] and GPT-4 (OpenAI, 2023) have

[1]https://chat.openai.com/

shown great capabilities of natural language understanding and text generation. With a simple prompting-based approach (Liu et al., 2023), many NLP tasks can be tackled by letting the model respond to the corresponding prompts (Jiao et al., 2023; Qin et al., 2023; Bang et al., 2023; Wei et al., 2023; Bubeck et al., 2023). It will be especially interesting to think about the roles of structured prediction in the era of LLMs.

Why Studying Language Structures? Recent neural models and especially LLMs have shown great capabilities of solving a wide range of NLP tasks, and more interestingly, this can be done by directly taking raw text sequences as input without relying on explicit language structures. In this case, we will need to face the question of why we still study and care about language structures. Here are some of the reasons why language structures can still be relevant to the field of NLP:

- **For linguistic importance.** Natural languages are organized in a structured way. This will not change regardless of what underlying models we are using to process natural languages. Studying language structures is still linguistically and theoretically important to understand the underlying working mechanisms of natural languages, especially the connections between surface form and underlying meaning. For example, word order, which indicates how words are linearized in sentences, is a typical example of how structures encode meaning (Dryer, 2007). The two sentences "The dog chases the cat" and "The cat chases the dog" contain the same tokens but denote different meanings with different word orders. Therefore, understanding language structures is a prerequisite to understanding the semantic meaning. Although nowadays typical LLMs do not explicitly encode language structures, they must have some mechanisms to implicitly encode structural knowledge.
- **For precise representations.** Although LLMs have surprisingly great capabilities at text generation and completion according to the given prompts, these open-ended generation models still suffer from the problem of hallucination (Ji et al., 2023), producing unrelated, unintended and even degenerated outputs. Moreover, extra efforts are still required to more precisely access the information contained in raw texts. Representing using explicit structures is such a way to obtain preciseness, which would make it easier for computers to directly store and retrieve relevant information. For example, information extraction systems extract nodes and relations from raw texts, which can further be utilized to construct knowledge graphs (Ji et al., 2021). Text-to-SQL systems convert natural language queries to structured query languages, which can be directly executed by SQL systems (Qin et al., 2022). In these applications, understanding the correspondences between the target structured representations and the source language structures can be a promising way to enhance more precise transformations.
- **For direct applications.** Although LLMs have shown great capabilities of solving many NLP tasks in an end-to-end fashion, there are still many cases where explicit structures can be helpful. An educational application where the system aims to help a student to learn a foreign language is a typical example. For an input sequence from the student, our goal is not only doing corrections but also highlighting and tagging the mistakes that the student makes. For the latter job, it will be more straightforward to model it as a structured prediction task of extracting and labeling input spans. Moreover, when we aim

to perform natural-language queries over a large number of instances, such as millions of database entries, it will be more efficient to use structured ways to represent and query the candidates rather than transforming all the items to natural language forms and querying with LLMs.

- **For better understanding.** Neural models have been questioned for their opaque nature (Lipton, 2018) and a better understanding of their working mechanism is an important research problem. A series of papers have been devoted to their analysis and interpretation (Belinkov and Glass, 2019; Belinkov et al., 2020). For example, probing has been utilized to query structured linguistic information (Conneau et al., 2018; Tenney et al., 2019; Hewitt and Manning, 2019; Chi et al., 2020; Conia and Navigli, 2022; Müller-Eberstein et al., 2022) over the neural representations. In a way, these can be viewed as looking for explicit structures inside neural networks, and exploring the connections between implicit neural encoding and explicit language structures is a promising way to achieve better interpretations of neural networks.
- **For better controlling.** Although LLMs seem to excel at language generation, it still remains a question of how to control them in the way that the users want. Prompting engineering is still a crucial procedure to properly steer the LMs (Liu et al., 2023). Towards more controllable text generation (Zhang et al., 2022a), utilizing language structures can be an interesting direction. For example, syntax has been utilized as the controlling guidance for paraphrasing (Iyyer et al., 2018; Huang and Chang, 2021) and graph-to-text generation (Liu et al., 2022b; Lee et al., 2022). Frame semantics have also been used to control dialog generation (Gupta et al., 2021b). In addition to traditional linguistic structures, the structures of the prompts have also been shown important for the effectiveness of using LLMs (Zhao et al., 2021b; Min et al., 2022b). How to better formalize the prompts is also crucial to better control the LLMs.
- **For better analyzing.** Any model is imperfect and will make mistakes. It is important to analyze the model's errors to make further improvements. In the aspects of model analysis and evaluation, language structures can also be helpful. For example, McCoy et al. (2019) show that constituency structures can be useful to construct controlled evaluation examples for natural language inference, Vamvas and Sennrich (2022) extract source constituents to detect over- and under-translation, Cardon et al. (2022) annotate linguistic corpus to evaluate text simplification and Xu et al. (2022) leverage syntax to diagnose spurious correlation problems for entity typing. The directions of error analysis and performance evaluation can be especially crucial in the era of LLMs because of their opaque nature and applications in open-ended text generation. Language structures can be helpful tools for these directions.

Challenge of Limited Data. Although neural models and large-scale pre-training have brought clear benefits to structured prediction tasks in NLP, the main problem-solving paradigm is still in a data-driven supervised-learning fashion. This paradigm relies on the availability of manually annotated data, which is usually expensive and time-consuming to obtain. The neural models can perform well with abundant annotated instances, but learning remains difficult in scenarios where such data resources are limited (Hedderich et al., 2021). Moreover, we usually hope that the models can be able to generalize to various new scenarios, like in different domains or languages. For example, we may want to apply a parser trained on newswire data to biomedical texts for further processing. There are thousands of different languages in this world and we would like to have a text analyzer that could handle multiple languages in a unified way. In these scenarios, the model performance could drop dramatically if the training instances are insufficient to cover the target scenarios, and it is costly and inefficient to annotate large amounts of data instances in all these new cases. For structured prediction, this resource-limitation problem can be more challenging, since the annotation of high-quality structured data requires proper training and demanding work from the annotators, limiting the scale of available corpora. For example, in the Penn Treebank (Marcus et al., 1993), one of the largest and most utilized syntactic corpora, most of the annotations come from news sources and it is unclear how well a parser trained on this single source can adapt to other text genres. While Universal Dependencies (Nivre et al., 2020), one of the largest syntactic annotation efforts, cover over one hundred languages, the treebank sizes are still limited for many low-resource languages, leading to sub-optimal performance. Therefore, it remains an interesting and important question of how to obtain more effective structured predictors in resource-limited scenarios.

1.2 Research Objectives

In this thesis, we perform a comprehensive investigation on NLP structured prediction problems in resource-limited scenarios, which bring great challenges due to the unavailability of large amounts of annotated data for model training. To mitigate this problem, we need to consider both the aspects of model and data, since they are the main driving forces of data-driven machine learning. Specifically, we investigate the following three directions.

Structured modeling. Traditionally, how to model the structured output space is one of the central topics for structured prediction in NLP. Since we need to predict complex structures like sequences and graphs, the size of the output space is usually exponential to the input length. For example, in the simplest sequence labeling case where we want to label a sentence of L tokens with T tags, there can be L^T possible outputs, making it impossible to explicitly enumerate each of the possibilities. Fortunately, the output space is highly structured, and with proper independence assumptions, we could usually factorize the outputs and find polynomial solutions. For example, the factorization of a sequence of labels into a linear chain with bi-gram interactions enables square-time decoding (Viterbi) and learning (forward-backward) algorithms (Viterbi, 1967; Baum et al., 1970); and the factorization of a dependency parse tree into individual dependency edges enables cubic-time processing methods (Eisner, 1996; McDonald et al., 2005a). Most traditional NLP work has been devoted to developing better structured models (Lafferty et al., 2001; McDonald and Pereira, 2006; Koo and Collins, 2010; Punyakanok et al., 2008). Nevertheless, as deep learning methods come into the field of NLP and bring performance improvements with better neural representations, the potential costs and benefits of structured modeling with these neural models have not yet been comprehensively explored. Furthermore, structured information, especially structural constraints such as tree constraints in parsing, can provide valuable inductive biases to the models, especially in low-resource scenarios where there is a lack of abundant data to implicitly learn such constraints. It will be worthwhile to explore the incorporation of such information into the model.

Transfer Learning. Although abundant direct annotations may be lacking for the target task, there can be helpful signals from other corpus and annotations. This is the main motivation of transfer learning (Pan and Yang, 2009; Ruder et al., 2019), which allows us to utilize the knowledge from related data to boost the performance on the target task. In NLP, the extra signals may come from various types of resources, such as those in the same task but in different languages (cross-lingual learning) (Yarowsky and Ngai, 2001; Padó and Lapata, 2009) and those in different but related tasks (multi-task learning) (Caruana, 1997; Ruder, 2017). The recent influential pre-training/fine-tuning framework (Devlin et al., 2019) can also be viewed as a form of sequential transfer learning, where the general language knowledge from large-scale unlabeled data is learned by pre-training and transferred to the target task. For structured prediction, there could be lots of chances for transfer learning, considering the shared underlying structures of human languages. For example, for syntactic parsing, there can be shared syntactic structures across different languages. This motivates the project of Universal Dependencies (UD) (Nivre et al., 2016b, 2020), which provides consistent cross-lingual syntactic annotations and has hitherto covered over 100 languages, enabling great chances for transfer learning from high-resource languages to low-resource ones. Moreover, considering the close relationship between the tasks of syntactic parsing and shallow semantic parsing, syntactic information could provide valuable auxiliary signals to help to learn semantic relations (Marcheggiani and Titov, 2017; Swayamdipta et al., 2018; Strubell et al., 2018). In this way, transfer learning leverages and learns from auxiliary signals and provides promising ways to combat the dilemma of lacking direct annotations. Notice that in transfer learning, there are discrepancies between source and target data. Therefore, designing models that are agnostic to these differences can facilitate more effective transfer.

Active Learning. In resource-limited scenarios, although annotating large amounts of labeled data is infeasible, there may still be budgets to obtain a small number of annotations. In this case, the choice of the data instances to label could be crucial for the effectiveness of the model training. This is the main idea behind active learning (Settles, 2009), which incorporates the model in the loop of data annotation. Considering the most common pool-based active learning setting (Lewis and Gale, 1994), a large collection of unlabeled data can be gathered and we iteratively select to-be-annotated data instances from the unlabeled pool with the knowledge from the current model. Each time we obtain new annotations, we further utilize them to learn better models. In this way, the procedure could lead to a model achieving satisfactory performance with fewer labeled data. There are further two aspects that may be interesting to explore. Firstly, for structured prediction tasks, it is usually laborious to annotate the full structures while the real desirable annotations lie in certain ambiguous sub-parts. For example, a sentence may be ambiguous only in the attachment of some difficult sub-structures like prepositional phrases, and in this case, the most effective way is to only annotate the related sub-structures. Allowing annotating and learning with partial structures could potentially make active learning more data-efficient for structured prediction tasks. Moreover, since we already have an unlabeled corpus, active learning could be combined with other semi-supervised learning methods (Zhu, 2005), which may lead to better utilization of the unlabeled data. Especially, self-training (Yarowsky,

1995) is a promising way to better utilize unlabeled target datasets to improve model performance (McClosky et al., 2006; Xie et al., 2020; He et al., 2020; Du et al., 2021a). It would be interesting to explore whether there could be further benefits with the combination of active learning and self-training.

1.3 Outline

According to these three aspects, we explore various methods to enable better structured predictors in resource-limited scenarios. The outline of the thesis is summarized as follows:

Background. In Chapter 2, we briefly review some of the basic backgrounds for structured prediction in NLP, including problem setups, typical tasks and models, as well as data resources. This will provide background information for the remaining chapters.

Part I: Structured Modeling. This part mainly concerns structured output modeling with neural networks for structured prediction tasks. It consists of the following chapters:

- Chapter 3 describes an empirical investigation of the effectiveness of structured output modeling together with the utilization of deep neural models. We take two typical structured prediction tasks, dependency parsing and semantic role labeling, as our study examples. With extensive experiments on various datasets, we find that though the improvement brought by structured output modeling is modest when utilizing strong neural models, there are still benefits on sentence-level complete-match metrics and with less powerful but more efficient models.
- Chapter 4 further provides an analysis of the interactions of the effectiveness of decoding with structural constraints and the amount of available training data for structured prediction tasks in NLP. Our exploration adopts a simple protocol that enforces constraints upon constraint-agnostic local models at testing time. With evaluations on three typical structured prediction tasks (named entity recognition, dependency parsing, and event argument extraction), we find that models trained with fewer data predict outputs with more structural violations in greedy decoding mode.

Part II: Transfer Learning. This part explores how the external resources from related datasets can be utilized to help the target task. It includes the following chapters:

- Chapter 5 investigates cross-lingual transfer, positing that an order-agnostic model will perform better when transferring to distant foreign languages. With rigorous experiments on dependency parsing over 30 languages, we find that order-sensitive sequential RNN-based models transfer well to languages that are close to the source, while order-agnostic self-attentive models have better overall cross-lingual transferability and perform especially well on distant languages.
- Chapter 6 further extends the cross-lingual transfer method to the SRL task and explores the helpfulness of syntactic supervision for cross-lingual SRL within a simple multi-task learning scheme. With comprehensive evaluations across ten languages (in addition to

English) and three SRL benchmark datasets, including both dependency- and span-based SRL, we show the effectiveness of syntactic supervision in low-resource scenarios.

- Chapter 7 further explores transferring knowledge from SRL resources to help the task of event argument extraction. With a template-based slot querying scheme, we show that SRL models can achieve reasonable zero-shot performance for event argument extraction, and the method could be promisingly extended to more settings such as semi-supervised learning and cross-domain cases.

Part III: Active Learning. In this part, we perform a systematic investigation of active learning for structured prediction in NLP:

- Chapter 8 provides a literature review of active learning (AL) for its applications in natural language processing (NLP). In addition to a fine-grained categorization of query strategies, we also investigate several other important aspects of applying AL to NLP problems. These include AL for structured prediction tasks, annotation cost, model learning (especially with deep neural models), and starting and stopping AL.
- Chapter 9 provides a pragmatic method that reduces the annotation cost for structured label spaces using active learning. Our approach leverages partial annotation, which further reduces labeling costs for structured outputs by selecting only the most informative sub-structures for annotation. We also utilize self-training to incorporate the current model's automatic predictions as pseudo-labels for unannotated sub-structures. In evaluations spanning four typical structured prediction tasks, we show that our combination of partial annotation and self-training using an adaptive selection ratio reduces annotation cost over strong full annotation baselines under a fair comparison scheme that takes reading time into consideration.

Conclusions and Future Directions. In Chapter 10, we conclude this thesis by summarizing our explorations as well as discussing future directions.

Chapter 2

Background

2.1 Basics

As introduced in the previous chapter, there are abundant structured problems in natural language processing, ranging from basic linguistic annotation tasks like tagging and syntactical parsing to various downstream tasks like those in information extraction as well as text generation. In this thesis, we focus on certain well-defined structured tasks, mostly consisting of classical linguistic tasks and information extraction ones. Formally speaking, structured prediction involves the prediction of a structured output object $\mathbf{y} \in \mathcal{Y}$ given an input object $\mathbf{x} \in \mathcal{X}$. Here, the inputs are usually sequences of words or tokens, for example, a sentence consisting of L words: $\mathbf{x} = \{w_1, w_2, \ldots, w_L\}$. While being task-specific, an output object consists of multiple inter-dependent variables that form certain structures for an input object. For example, in sequence labeling, the output consists of a sequence of tags: $\mathbf{y} = \{t_1, t_2, \ldots, t_L\}$, where each tag corresponds to each input word of the same index. To solve the task, we aim to obtain a structured predictor that could map an input object to an output object. Generally speaking, we need a model m that could assign a scalar value score$(\mathbf{y})$ to each output structure[1][2] and the prediction aims to find the highest scored output: $\hat{\mathbf{y}} = \mathrm{argmax}_y$ score$(\mathbf{y})$.

How to model the scoring function from the inputs to the structured outputs is one of the central problems in structured prediction tasks. In the remainder of the section, we will introduce typical structured modeling formalisms.

Parameterized Modeling and Supervised Learning

In typical language structured prediction as well as general NLP and machine learning tasks, the model specifies a collection of functions. It transforms the inputs into the outputs, and this procedure is realized by the application of these functions. The differences in these functions specify the types of models. For example, traditional linear models consist of a manually designed feature extractor and a linear transformation function, while recent neural models consist

[1]If the model is explicitly probabilistic, the score can be the probability of $\mathbf{y}$ given input $\mathbf{x}$. Nevertheless, we view the model as a scoring function for generality, which could cover both probabilistic and non-probabilistic cases.

[2]The scoring function depends on the input $\mathbf{x}$ and the model m as well, which we omit for brevity in this thesis if the context is clear.

of multiple layers of non-linear transformations. Despite the differences, the model is usually parameterized, that is, its functions contain learnable and usually input-independent parameters, which we refer to as θ_m. For example, for a linear function f that transforms an input vector $\mathbf{v}_{in}$ to an output vector $\mathbf{v}_{out}$ by multiplying a weight matrix $\mathbf{W}$ and adding a bias vector $\mathbf{v}_b$: $f(\mathbf{v}_{in}) = \mathbf{v}_{out} = \mathbf{W}^T\mathbf{v}_{in} + \mathbf{v}_b$, its parameters are $\theta_f = \{\mathbf{W}, \mathbf{v}_b\}$ and their values are changeable and can be adjusted by a learning procedure.

The basic form of learning is data-driven supervised learning, where we train the models with a training corpus D, which contains a collection of input-output pairs:

$$D = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}, \text{ where } (\mathbf{x}_i, \mathbf{y}_i) \in (\mathcal{X} \times \mathcal{Y})$$

This is inspired by the learning process of humans: people usually make decisions based on knowledge learned from previous experience. Similarly, in machine learning, the model could utilize the gold input-output supervision in the training data to adjust (learn) the values of its parameters. In this thesis, although we are interested in scenarios where such training data resources are limited, we still mainly follow the data-driven supervised learning paradigm.

Two Aspects: Factorization and Normalization

Structured prediction problems are different than plain classification or regression problems in that the outputs are complex structured objects. Specifically, they have the following two unique properties:

- **Intractable to enumerate.** Since the output consists of multiple variables, the possible output space can be exponentially large and it is usually intractable to explicitly enumerate all possible items. To handle the prediction in such a large output space, we need to shrink the space that is modeled at one time.
- **Structurally constrained.** The variables in the output structure are usually explicitly or implicitly connected to the inputs and are interdependent. In many problems, there are also explicit structural constraints over them. For example, in parsing, the output should form a tree, where there are no cycles with the predicted edges.

According to these characteristics, we examine two aspects for the output modeling of structured prediction problems: ***factorization*** and ***normalization***.

1. Factorization denotes how the complex output structures are factorized into manageable smaller parts. This directly aims to tackle the problem brought by *the difficulty of enumeration*. Since the full output space is too large to explicitly enumerate, we have to split the full problem into smaller pieces to consider at one time. This is the main idea of factorization or decomposition. Specifically, we usually decompose the full output structure $\mathbf{y}$ into a collection of smaller parts (sub-structures) p and the score of $\mathbf{y}$ is the sum of the scores of the sub-structures:

$$\text{score}(\mathbf{y}) = \sum_{p \in \mathbf{y}} \text{score}(p)$$

Factorization is usually the ever first thing that needs to be specified when modeling a structured object and its realization is highly task-dependent. In NLP, the factorization usually follows the structure of the input sequences. For example, for the modeling of a sequence of tags, each of which corresponds to one input token, a natural choice of factorization is to decompose the full structure by assigning a score to each of the tags:

$$\text{score}(\mathbf{y}) = \text{score}(\{t_1, t_2, \ldots, t_L\}) = \sum_{i \in [1,L]} \text{score}(t_i | \{t_{\text{dep}}\})$$

A central decision to be made for factorization is what kind of ***independence assumption*** we would like to assume, that is, for the scoring of each sub-structure, how much it depends on the other output sub-structures. Again taking sequence labeling as an example, if we do not make any independence assumption and let the scoring of one tag depend on all other tags, we still face an exponential problem if we want to find the exact global optimal structure. On the other hand, if assuming that given the input, each of the output tags is independent of each other, each scoring function will only need to consider the output space of a single tag, whose size is T. This makes the search problem much easier, but the model will be less powerful since this assumption is too strict. There can be options between these two extremes, for example, a classical way to do factorization for sequence labeling is adopting an n-gram Markov assumption: the decision of one tag only depends on previous $n-1$ tags. In this way, the model becomes more powerful since nearby tags, which are usually the most influential ones, are considered, while there can still be efficient algorithms to tackle the search problem.

2. Normalization denotes how we want to compare different choices of the sub-structures that we decompose according to certain factorization. This is closely related to the *structural constraints* of the problem. In the descriptions above, we use a general scoring function[3] to denote the model without assuming a probabilistic treatment. To better illustrate the idea of normalization, we adopt probabilistic models and things can usually be generalized to non-probabilistic ones. Typical choices of normalization for structured prediction include ***locally*** and ***globally*** normalized modeling. The former performs comparisons to the local variants of sub-structures while the latter compares overall global structures. Taking the sequence labeling with bi-gram Markov assumption as an example, with local normalization over each tag, we have:

$$p_l(\mathbf{y}) = \prod_{i \in [1,L]} p(t_i | t_{i-1}) \;\; \text{where} \;\; p(t_i | t_{i-1}) = \frac{\exp \text{score}(t_i | t_{i-1})}{\sum_{t'_i \in [1,T]} \exp \text{score}(t'_i | t_{i-1})}$$

Here, we use $\text{score}(t'_i | t_{i-1})$ to denote the model-assigned score of each tag depending on its previous tag.[4] The probability of the full structure is factorized into the probability of each tag, which is locally normalized to the tagging space. In this way, we only need to compare different choices (and specify loss function) for this single tag at one time and thus this model is regarded to be locally normalized. On the other hand, global normalization aims to compare full structures over the full output space:

$$p_g(\mathbf{y}) = \frac{\exp \text{score}(\mathbf{y})}{\sum_{\mathbf{y}' \in L^T} \exp \text{score}(\mathbf{y}')} = \frac{\exp \sum_{i \in [1,L]} \text{score}(t_i | t_{i-1})}{\sum_{\mathbf{y}' \in L^T} \exp \sum_{j \in [1,L]} \text{score}(t'_j | t'_{j-1})}$$

[3]With probabilistic models, the probability of the full structure is the multiplication of its sub-parts. We could simply take the logarithm of the probability as the score which turns multiplication into summation.

[4]We simply assume a dummy tag t_0 before the first real tag.

Notice that in this formula, we no longer have individual local probabilities for each tag, but there is one overall probability for the full structure and thus the comparisons are performed in the global output space. Although in the denominator, we still need to calculate a summation of exponentially sized items, thanks to the independence assumption, there are usually efficient polynomial-time algorithms. One clear advantage of global models is that they could take arbitrary ***structural constraints*** into consideration since the normalization is performed directly in the global space. For example, in parsing, we have a constraint that the final output structure should be a well-formed tree, which is difficult to specify if adopting local models. However, with global models, this can be naturally reflected by only including valid trees in the denominator. Notice that the difference between local and global models also depends on the factorization, for example, in the single-tag factorization case without any output tag interactions, local and global models are actually the same (by re-formulating the denominator with distributive law).

With these two aspects specified, we define a model for a structured prediction problem, which could assign scores (or probabilities) to an output structure given an input object.

Three Problems: Representation, Decoding, and Learning

Once we specify a model, to apply it to the structured prediction task, we still face several problems. According to McDonald et al. (2005a), there are mainly three basic questions:

- **Representation:** What representations to use to transform the inputs to the output scores?
- **Decoding:** How to find the highest scored output structure for an input?
- **Learning:** How to learn the model parameters from the training data?

In this sub-section, we provide some brief discussions on these central questions.

1. Representation. Previously we've been mainly discussing aspects with regard to the structured output modeling, another important facet of the model is the representations of the inputs and the transformation to the output scores. As in many other NLP tasks, the representation method evolves from manually specified sparse features to automatic neural representations. For an NLP task, the inputs are usually sequences of discrete symbols, which could be represented as *one-hot vectors*. For an input sequence of L words $\mathbf{x} = \{w_1, w_2, \ldots, w_L\}$ where each word is one symbol from a vocabulary of size V, we could give each w_i an integer that is the index of the specific word in the input vocabulary. Then for each word, we could represent it with a sparse vector $\mathbf{v}_s = [0, \ldots, 1, \ldots, 0]$ where there is only one item that is 1 and the item's index is w_i. Traditional *feature-based methods* further specify other features and higher-order feature combinations with manually designed templates. For example, in addition to the current word's identity, we could also have various n-gram features which take surrounding words into consideration. One crucial feature type in classical models involves higher-order combinations of atomic features, which are the main source of the model's discriminative power. We could view each feature as an entry in the feature vocabulary and specify a sparse feature vector to represent the object to be scored. In NLP applications where features are usually binary, the feature vector is a sparse vector, which most contains 0s except for a few 1s where the specific features are activated. Following this feature extractor, traditional methods usually adopt a simple linear model, which contains a parameter weight $\mathbf{w}$, to score the outputs. Each parameter in the weight is associated with one feature in the feature vocabulary and scoring is realized by multiplying the feature vector and the weight:

$$\text{score}(\mathbf{y}) = \mathbf{w} \cdot \mathbf{f}(\mathbf{y})$$

Here, $\mathbf{f}(\mathbf{y})$ denotes the feature-extracting function that generates the feature vector according to the object to be scored. Although this representation and transformation method is simple and intuitive, there are two main disadvantages: 1) Each feature is individually represented and potential connections between them are mostly ignored. For example, in natural languages, many words can express the same meaning, and if using one-hot features for representation, such word similarity relations are not well considered. 2) Feature combinations are quite limited since things can explode if moving to higher orders. In this way, most of the sparse features only consider local word windows and could not effectively model longer contexts. The *neural-based methods* come around with distributed dense representations, that could mitigate these problems. Instead of sparse feature vectors, neural methods utilize low-dimensional vectors for representations, which could effectively capture the relations of different features, such as word similarities. Moreover, manually specified feature combinations are replaced by deep non-linear transformations where layers of parameterized non-linear functions (usually a combination of linear transformation plus non-linear activation) are applied to allow automatic feature combinations. In this way, neural models could more effectively capture the information from a wider range of input contexts.

2. Decoding. If we can only use one formula to show the essence of structured prediction, probably that will be the decoding objective, that is, finding the highest-scored output:

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \mathcal{Y}(\mathbf{x})}{\text{argmax}}\, \text{score}(\mathbf{y})$$

Here, $\mathcal{Y}(\mathbf{x})$ denotes the set of potential output structures for the input. As discussed above, the main difficulty of decoding is that the output space is usually exponentially large corresponding to the input size and explicit enumeration is nearly impossible. Moreover, the output usually needs to be structurally constrained, which makes decoding even harder. There can be two ways to mitigate the hardness of enumeration. Firstly, if we put strong independence assumptions, there are usually efficient algorithms to solve the task. With independence assumptions that enable factorization of the output into smaller pieces, which are usually easy to solve individually, the final solution can be obtained by merging the answers of the smaller pieces. This is actually the idea behind *dynamic programming*, and as we will discuss with detailed examples later, lots of classical NLP algorithms could solve the decoding problems in such a way. On the other hand, if we want to include more output dependencies and assume few or even no independence assumptions, decoding can still be feasible but we usually need to concede to approximate but reasonably good solutions rather than the exact best solution. Many algorithms are designed in this way, in which *greedy-* and *beam-* search are probably the most widely used ones. These searching algorithms decide the output variables in an incremental way, usually according to the adopted factorization. In the decoding process, they maintain an agenda storing the partial hypotheses so far and only keep the highest scored one or ones. There can be search errors where the actual highest-scored candidates get pruned because their prefixes get relatively low scores. Nevertheless, in NLP applications, they usually empirically work well and are the typical solutions for approximate searches.

3. Learning. Learning is the procedure of deciding the values of the model parameters with the training data. Usually, a loss function L is specified for an input-output pair, and learning is another optimization problem to get the optimal parameters leading to the minimal loss:

$$\hat{\theta} = \operatorname*{argmin}_{\theta} \frac{1}{N} \sum_{i} L(\mathbf{x}_i, \mathbf{y}_i; \theta)$$

One of the widely adopted loss functions is negative log-likelihood (NLL) of the gold output $L_{NLL} = -\log p(\mathbf{y}|\mathbf{x})$, leading to the well-known maximum likelihood estimation method. Another widely utilized loss function for structured prediction is the max-margin loss, which requires the score of the gold output to be higher than others by certain margins. In structured prediction, the loss function is closely related to the normalization aspect discussed above. For local models, the loss function is usually easy to calculate with the price of less global modeling power. Global models directly normalize over the output space and can adopt various structural constraints, but the calculation of the loss is usually more difficult. The optimization problem in the learning is usually hard to solve directly and iterative gradient-descent methods are utilized. In the simplest form, at one iteration t, the current parameters θ^t are updated to θ^{t+1} along the opposite direction of their gradients with regards to the loss function L:

$$\theta^{t+1} = \theta^t - \alpha \nabla_\theta L$$

where α is the learning rate specified as a hyper-parameter. For the neural-network-based models, stochastic gradient descent (SGD) and its variants are the default optimization algorithms. Especially, for structured prediction, it requires not only clever decoding algorithms to find the highest-scored structure in testing time but also similar inference algorithms for the purpose of gradient calculation in training time. For example, if adopting the margin-based loss function, performing loss-augmented decoding is a requirement to calculate the (sub-)gradient. And for global normalized probabilistic models, the calculation of marginal probabilities of the sub-parts is usually required, which can be solved with similar inference algorithms as in the decoding process. In this way, structured prediction also distinguishes itself by requiring careful treatment in the learning process.

In this section, we give a brief overview of the basics of structured prediction, covering the basic parameterization and data-driven formalism, two aspects of structured output modeling and three central problems. Most of these are well-known and widely-studied topics and more details can be found in related textbooks (Manning and Schutze, 1999; Jurafsky, 2000; BakIr et al., 2007; Smith, 2011; Goodfellow et al., 2016).

2.2 Tasks

In this section, we will describe several typical structured prediction tasks in NLP that are the main topics in this thesis and the corresponding widely-utilized models for them.

Sequence Labeling

Considering the structural nature of languages, sequence labeling, which gives a sequence of tags to an input sequence of tokens, is a fundamental task form. Here, we take two typical tasks, POS tagging, and NER, as examples:

Tokens:	The	student	wrote	a	paper	at	Carnegie	Mellon	University
POS:	DET	NOUN	VERB	DET	NOUN	ADP	PROPN	PROPN	PROPN
NER:	O	B-TITLE	O	O	O	O	B-ORG	I-ORG	I-ORG

The POS task is straightforward: assigning a POS tag to each of the input tokens, such as NOUN or VERB. While the NER task is more complex since it involves not only labeling but also segmenting: entities can be multi-word expressions that span several tokens. Nevertheless, for the usual case where there are no overlapping entity spans, the outputs for NER can still be represented as a label sequence with specific schemes. Here, we show the widely utilized "BIO" scheme (Ramshaw and Marcus, 1995), specifying tags for the **B**eginning, the **I**nside and the **O**utside of an entity span. For example, for the organization (ORG) "Carnegie Mellon University", "Carnegie" is the beginning token of the entity, thus receiving "B-ORG" while the remaining ones get "I-ORG", denoting that they are inside this entity. There could also be other representation schemes such as BILOU, which specifies extra tags for the **L**ast token and **U**nit-length chunks and the choice of the schemes could have impacts on system performances (Ratinov and Roth, 2009).

For the structured modeling of sequence labeling, a typical choice of factorization is to consider the bi-gram interactions of nearby tokens. Classical models such as Hidden Markov Models (HMM) (Rabiner, 1989), maximum entropy Markov model (MEMM) (McCallum et al., 2000) and conditional random field (CRF) (Lafferty et al., 2001) usually adopt such decomposition. HMM is a generative model for the joint probability of input and output $p(\mathbf{x}, \mathbf{y})$ and it performs local normalization. On the other hand, MEMM and CRF are discriminative models directly targeting at modeling $p(\mathbf{y}|\mathbf{x})$. The main difference is that MEMM normalizes locally while CRF is globally normalized. Thanks to the Markov assumption, there are efficient square-time algorithms for the inference algorithms for these models, including *Viterbi algorithm* (Viterbi, 1967) for decoding and *forward-backward algorithm* (Baum et al., 1970) for marginal inference in learning. While traditionally these models are based on sparse representations and linear models, recent developments of neural models have brought clear improvements. LSTM-CRF (Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016) is a typical example that utilizes *long-short term memory* (LSTM) (Hochreiter and Schmidhuber, 1997) network to encode the context of the full input sequence. More recently, models based on pre-trained contextualized models, such as BERT (Devlin et al., 2019) have been shown to bring further benefits. In addition, BERT-based models could surprisingly obtain good results on NER even when factorizing the sequence into individual tags without considering any tag-to-tag dependencies. Nevertheless, it has been shown that structural modeling can still bring benefits (Wei et al., 2021).

Dependency Parsing

Dependency parsing is the task of processing input sentences into dependency tree structures (Kübler et al., 2009). The history of the underlying dependency grammar can be traced back to the old time before the Common Era, while the modern dependency grammar was generally thought to start from the work of Tesnière (1959). Here is an example dependency tree, which is auto-parsed by UDPipe (Straka et al., 2016):

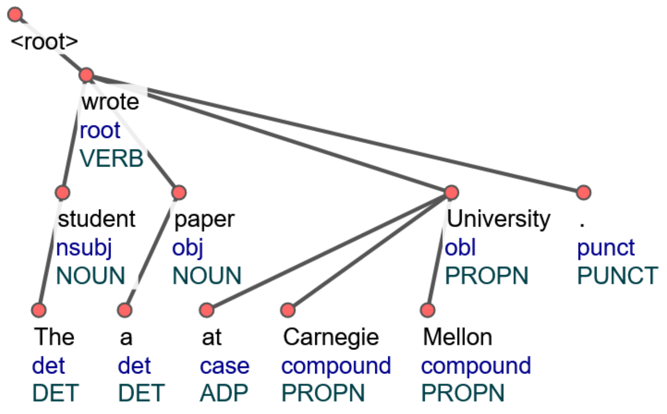


Figure 2.1: An example dependency tree.

Here, each node corresponds to a word while an edge between two words forms a dependency relation. These syntactic relations are represented by directed and labeled edges going from *parents (heads)* to *dependents (modifiers)*. Conventionally, an extra artificial “<root>” is added as the root node of the tree. There are certain constraints for a well-formed dependency tree:

- No multi-edges: there is at most one edge between two nodes.
- Single-headed: there is one and only one incoming edge for each node, except the artificial root node whose in-degree is zero.
- Acyclic: there should not be any cycles in the graph.
- (Optional) Projective: simply speaking, a tree is said to be projective if there are no crossing arcs if drawn into a plane. Notice that this property is optional since there can be non-projective arcs in languages like Czech and certain constructions in even highly projective languages like English.

All these properties or constraints will influence the algorithms for decoding and learning, which we will especially investigate more in later chapters.

Dependency parsing is the task of obtaining a well-formed dependency tree for an input sentence. Traditionally speaking, there are mainly two categories of parsing methods: transition-based (Yamada and Matsumoto, 2003; Nivre, 2003) and graph-based (Eisner, 1996; McDonald et al., 2005a). We can view these two groups of models from the aspect of factorization. The transition-based models decompose the prediction of the parse tree into a series of incremental transition actions while the graph-based ones decompose the full structures into individual sub-trees to score. We will provide some brief descriptions for them and please refer to related texts for more details (Kübler et al., 2009; McDonald and Nivre, 2014).

Transition-based models derive the output structures by incrementally transiting from one state to another, similar to the shift-reduce parsing paradigm in analyzing programming languages (Aho and Ullman, 1972). In the parsing procedure, we maintain some configurations as states, including a *stack* which stores the already-built structures and a *buffer* which stores the to-be-processed tokens. In each parsing step, the model decides an *action* that transits the current state into the next. These actions usually include *Arc* actions which build dependency arcs with the top nodes in the stack or buffer and *Shift* actions which move the top token from the buffer into the stack. There are various different transition schemes, like ArcStandard (Yamada and Matsumoto, 2003), ArcEager (Nivre, 2003, 2004), ArcHybrid (Kuhlmann et al., 2011) and so on. These schemes differ in the definitions and the effects of the allowed transition actions, which we will not discuss in more detail. Interested readers could refer to Bohnet et al. (2016) for a generalized description of transition-based systems. Notice that transition-based models are not constrained to left-to-right shift-reduced styled systems, and there can be non-directional models such as the easy-first parser (Goldberg and Elhadad, 2010), which maintains a pending list for current non-attached tokens and allows building arcs to any two nearby tokens in the pending list. Recently, there is also work casting dependency parsing as sequence labeling (Strzyz et al., 2019; Gómez-Rodríguez et al., 2020) with special encoding schemes. In some way, these systems can also be regarded as transition-based considering the similarities between the labels and the transition actions.

Graph-based models directly decompose the full dependency tree into individually scored sub-structures (sub-trees) and obtain the final decision with specific global parsing algorithms. In its simplest form, that is, the first-order or edge-factored model, the score of a dependency tree is the summation of all its edges' scores, which are estimated individually:

$$\text{score}(T) = \sum_{(h,m)\in T} \text{score}(h, m)$$

Here, (h, m) denotes a dependency edge from a **h**ead word to a **m**odifier word in a tree T. According to the single-headed constraint, each token (except the artificial root) has one parent node and thus one incoming dependency edge with a score. First-order graph-based parsers are well-studied and there are corresponding algorithms for the decoding and learning procedures: Eisner's dynamic programming algorithm (Eisner, 1996) and Chu-Liu-Edmonds' maximum spanning tree algorithm (Chu and Liu, 1965; Edmonds, 1967) for decoding and a variation of Inside-Outside algorithm (Paskin, 2001) and Matrix-Tree Theorem (Koo et al., 2007; Smith and Smith, 2007; McDonald and Satta, 2007) for marginal inference in learning. We will provide more detailed descriptions and analyses of algorithms and constraints for first-order graph-based parsing in Chapter 3. Naturally, the model could be extended with the factorization of larger sub-structures involving multiple edges, like second-order including sibling nodes (McDonald and Pereira, 2006) and grand-parent nodes (Carreras, 2007) as well as third-order (Koo and Collins, 2010) and fourth-order (Ma and Zhao, 2012) models. Nevertheless, higher-order parsers pay the price of higher computational complexities, and thus probably the most widely utilized graph-based parsers are still the simplest first-order ones (Qi et al., 2018, 2020).

For the model architectures and representations, dependency parsing also evolves from traditional sparse-feature-based models to recent neural counterparts. Chen and Manning (2014) first applied a feed-forward neural network to score the actions for transition-based parsing and Pei et al. (2015) similarly applied neural networks for the sub-part scoring in graph-based models. Dyer et al. (2015) utilized better LSTM-based neural models to naturally model the sequential decisions for transition-based parsing while previous work also showed the benefits of structured modeling (Weiss et al., 2015; Zhou et al., 2015a; Andor et al., 2016). Later on, (Kiperwasser and Goldberg, 2016) indicated the effectiveness of encoding the full input sequence with bidirectional LSTM model while (Dozat and Manning, 2017) provided further improvements with a deep biaffine-attention-based scoring component. Notice that the architecture from the Dozat and Manning (2017)'s parser is nowadays one of the most widely utilized neural architectures for parsing. More recently, the development of pre-trained contextualized language models brought large improvements to dependency parsing, which also interestingly seems to make the behavior of transition- and graph-based parsers converge (Kulmizev et al., 2019).

Shallow Semantic Parsing

While syntactic parsing only cares about syntactic relations such as subjects and objects, the goal of natural language understanding is to realize the semantic meanings. Properly understanding semantic roles, which is the task of shallow semantic parsing or semantic role labeling, is one crucial step to achieve this goal. Considering the following ways of describing roughly the same meaning:

- $\underline{\text{The student}}$ **wrote** $\underline{\text{a paper}}$. (*plain*)
- $\underline{\text{A paper}}$ is **written** by $\underline{\text{the student}}$. (*passive*)
- This is $\underline{\text{a paper}}$ that $\underline{\text{the student}}$ **wrote**. (*relative clause*)
- This is $\underline{\text{a paper}}$ **written** by $\underline{\text{the student}}$. (*relative clause* + *passive*)

There are various syntactic ways to realize the fact that there is a "writing" action where the writer is "the student" and the thing that is written is "a paper". If looking at the syntactic tree, the structures of these sentences are quite different, for example, the writer "student" can be linked to the center verb with different syntactic relations. However, we know that in the underlying semantics, the writer is always the same, that is, "the student" always acts as the role of the writer. This is the idea behind *semantic roles*, which express the roles that are taken by the arguments of action-like or eventive predicates.

A predicate and its arguments form a structure that is usually called a semantic frame, and the task of shallow semantic parsing aims to extract such semantic frames from input texts. Although the task itself is natural, it turns out to be quite difficult to specify a standard set of semantic frames and especially their role labels. For example, in the writing scenario in the above examples, we could have role labels at different granularities, either more general ones like Agent and Patient or more specific ones like Author or Text. This leads to different semantic role schemes, where PropBank and FrameNet are two representatives. PropBank (Palmer et al., 2005) takes generalized and theory-neutral semantic roles of Arg0, Arg1, etc. (Arg0 and Arg1 usually respectively correspond to the proto-agent and proto-patient concepts from Dowty (1991)) and the actual meaning of them is frame-specific. On the other hand, FrameNet (Baker et al., 1998a) takes a more fine-grained approach by collecting and grouping frames and frame elements and giving them natural-language names. For example, for the same input sentence, we could parse it into similarly structured semantic frames according to these two schemes:

- PropBank: [The student]$_{\text{A0}}$ **wrote**$_{\text{write.01}}$ [a paper]$_{\text{A1}}$ [at Carnegie Mellon University]$_{\text{AM-LOC}}$
- FrameNet: [The student]$_{\text{Author}}$ **wrote**$_{\text{Text\_creation}}$ [a paper]$_{\text{Text}}$ [at Carnegie Mellon University]$_{\text{Place}}$

Here, in the PropBank scheme, "write.01" indicates that the predicate's lemma is "write", it takes its first sense "01", and its arguments have role labels of A0 and A1 which are specified to the "write.01" frame as well as a general location modifier "AM-LOC". In FrameNet, it assigns a "Text_creation" frame to the predicate and more fine-grained roles to its arguments. Based on what text provenances to extract, we could also categorize SRL into span-based and dependency-based SRL. The former is closely related to constituency syntactic parsing and requires the extraction of full-text spans (constituencies) of the arguments. While in natural language, the semantic meaning of an argument span usually resides at certain head words. This inspires dependency-based SRL, popularized by 2008 and 2009 CoNLL shared tasks (Surdeanu et al., 2008; Hajič et al., 2009), which only requires the extraction of the head word for each argument.

No matter what scheme to take, to perform automatic semantic role labeling (Gildea and Jurafsky, 2002; Palmer et al., 2010), there are generally two steps: 1) predicate identification and disambiguation, and 2) argument identification and labeling. The first step can be cast as a classification or sequence labeling task, while the second role labeling step is of more interest where relational predicate-argument links are required to be extracted. Traditional SRL systems usually take syntactic trees as inputs (Xue and Palmer, 2004; Màrquez et al., 2008) and they treat constituents from the tree as argument candidates. And again, recent developments of neural networks allow end-to-end models that simplify the processing pipeline and get performance improvements (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Strubell et al., 2018; Shi and Lin, 2019). We will provide more discussions on the output modeling of SRL in Chapter 3 as well as the connection between syntax and SRL in Chapter 6.

2.3 Data Resources

In this section, we introduce widely utilized data resources for the structured prediction tasks described above, which also include the main datasets we use in the later chapters.

Penn Treebank (PTB). PTB (Marcus et al., 1993) is probably one of the most representative linguistically-annotated corpus. The English PTB project manually annotates POS tags and constituency syntactic trees over millions of words. Especially, the Wall Street Journal (WSJ) portion of it is nowadays the most standard benchmark for the tasks of POS tagging and syntactic parsing. While the syntactic trees annotated in PTB are constituency-based, there are tools utilizing specific head-finding rules (Collins, 2003) to convert them into dependency trees. Older conversions include Penn2Malt[5] and LTH Constituent-to-Dependency Conversion Tool[6] (Johansson and Nugues, 2007), while recently Standard Dependencies (de Marneffe et al., 2006;

[5]https://cl.lingfil.uu.se/~nivre/research/Penn2Malt.html
[6]http://nlp.cs.lth.se/software/treebank_converter

De Marneffe and Manning, 2008) become the canonical conversion target. Following the English PTB, similar treebanks for other languages have been created, such as the Penn Chinese Treebank (CTB; Xue et al., 2005) and the Penn Arabic Treebank (PATB; Maamouri et al., 2004), providing valuable data resources for the studying of syntactic parsing in multiple languages.

Universal Dependencies (UD). While previous treebanks adopt language-specific annotations in different languages, the UD project (Nivre et al., 2016b, 2020) aims to provide consistent annotations of grammar across different human languages. Its latest version (v2.12) adopts 17 universal POS (Petrov et al., 2012) tags and 37 universal syntactic relations (de Marneffe et al., 2014) labels as annotating target sets, covering over 200 treebanks in more than 100 languages. This provides valuable data resources and benchmarks for multilingual and cross-lingual analyses of the structures of human languages.

Proposition Bank (PropBank). PropBank builds data resources annotated with semantic roles (Palmer et al., 2005). Most of the PropBank resources create an SRL layer upon the syntactic layer and adopt verb-sense-specific generic argument roles (Arg0, Arg1, Arg2, and so on). Originally focusing on English, PropBank-styled annotations have been also extended to other languages, such as Chinese (Xue, 2008), Arabic (Palmer et al., 2008), Finnish (Haverinen et al., 2015) and so on. While the original version of PropBank focuses on verb predicates, a related project, NomBank (Meyers et al., 2004), extends the annotations to nominal predicates. Recently, PropBank has been updated to cover more predicate types (Bonial et al., 2014) and provide unified role-sets and data resources (O'Gorman et al., 2018; Pradhan et al., 2022), which are aligned with Abstract Meaning Representation (AMR; Banarescu et al., 2013), another widely-adopted meaning representation scheme.

FrameNet. The Berkeley FrameNet project produces thousands of English lexicon items as well as annotated corpus (Baker et al., 1998a). Different from the PropBank approach, FrameNet groups similar words together into frames and assign fine-grained frame-specific roles (called frame elements) to each frame. For example, as discussed in the previous section, we would have a "Text_creation" frame with roles of "Author" and "Text" rather than a "write.01" frame with "Arg0" and "Arg1". In some way, FrameNet and PropBank aim to capture the same underlying language phenomena but with different labels. Naturally, there is work trying to map among different SRL frame sets, such as SemLink (Palmer, 2009; Stowe et al., 2021).

OntoNotes. The OntoNotes project (Hovy et al., 2006; Weischedel et al., 2013) creates a large-scale corpus with multiple levels of linguistic structures for text, covering syntax, SRL, word senses, named entities, and coreference. The annotations include texts from multiple genres (such as newswire, conversation, weblogs, etc.) and multiple languages (English, Chinese, and Arabic), creating a resource that could be broadly applicable.

CoNLL Shared Tasks. Since 1999, the Conference on Computational Natural Language Learning (CoNLL) has included a shared task each year, in which for a specific language task, training and testing data are collected and provided by the organizers to allow the evaluation and comparisons of current NLP systems in a systematic way. Here we list the ones that are related to the main topics for structured prediction in this thesis:

- CoNLL-2000: Chunking. CoNLL-2000 (Tjong Kim Sang and Buchholz, 2000) includes the task of text chunking, which can be formalized as a sequence labeling problem. The datasets are prepared with the WSJ portion of PTB.
- CoNLL-2002 & 2003: Named Entity Recognition. CoNLL 2002 (Tjong Kim Sang, 2002) and 2003 (Tjong Kim Sang and De Meulder, 2003) investigate the task of NER in four languages. Especially, the English dataset in CoNLL-2003 taken from the Reuters Corpus is probably the most standard and widely-used NER benchmark.
- CoNLL-2004 & 2005: Semantic Role Labeling. CoNLL-2004 (Carreras and Màrquez, 2004) and 2005 (Carreras and Màrquez, 2005) focus on SRL, utilizing the English PropBank. The CoNLL-2005 dataset is one of the standard SRL benchmarks, including a fresh test set from the Brown corpus to test cross-domain performance.
- CoNLL-2006 $\sim$ 2009: Syntactic and Semantic Dependency Parsing. In four years, the shared tasks are devoted to dependency parsing, greatly encouraging the development of this task. The first two (Buchholz and Marsi, 2006; Nivre et al., 2007) only concern syntactic dependency parsing, while the latter two (Surdeanu et al., 2008; Hajič et al., 2009) include joint syntactic and semantic dependencies.
- CoNLL-2011 & 2012: Coreference. CoNLL 2011 (Pradhan et al., 2011) and 2012 (Pradhan et al., 2012) investigate the task of coreference, taking data from the OntoNotes. Notice that the standard SRL benchmark with OntoNotes also takes the dataset splits of the CoNLL-2012 task.
- CoNLL-2017 & 2018: Parsing into Universal Dependencies. CoNLL 2017 (Zeman et al., 2017) and 2018 (Zeman et al., 2018) revisit the task of syntactic dependency parsing, but this time focusing on Universal Dependencies with cross-lingual consistent annotations.

These tasks usually involve language structure predictions and provide standard benchmarks for future research on these topics.

Notice that these data resources are usually costly and time-consuming to build. In NLP applications, we usually meet problems that suffer from data distribution shifts from these existing large-scale datasets, such as those in different domains and different languages where direct target data is limited. Therefore, it would be important to develop better NLP models and structured predictors in such resource-limited scenarios. In the remaining chapters, we will discuss our investigations on this topic.

Part I

Structured Modeling

Chapter 3

An Empirical Investigation of Structured Output Modeling for Language Structured Prediction

As discussed in Chapter 1, while recent development of neural models has brought great improvements for structured prediction tasks, the costs and benefits of structured output modeling have become less clear. To explore its roles, in this chapter, we provide an empirical investigation of two typical structured prediction tasks:

- In Section 3.1, we study first-order graph-based dependency parsing with different ways of *normalization*, which correspond to different structural constraints. Experiments on 14 treebanks show that generally global models can obtain better performance, especially on the metric of sentence-level complete-match scores.
- In Section 3.2, we study semantic role labeling (SRL) with different *factorization* methods for span extraction. We investigate five strategies, which fall into three categories: BIO-based, span-based, and two-step approaches. With extensive experiments on PropBank SRL datasets, we find that more structured methods outperform BIO-tagging when using static word embeddings across all experimental settings.

3.1 Constraints in Dependency Parsing

In the past few years, dependency parsers, equipped with neural network models, have led to impressive empirical successes in parsing accuracy (Chen and Manning, 2014; Weiss et al., 2015; Dyer et al., 2015; Andor et al., 2016; Kiperwasser and Goldberg, 2016; Kuncoro et al., 2016; Dozat and Manning, 2017; Ma et al., 2018). Among them, the deep-biaffine attentional parser (**BiAF**) (Dozat and Manning, 2017) has stood out for its simplicity and effectiveness. BiAF adopts a simple bi-directional LSTM neural architecture (Ma and Hovy, 2016; Kiperwasser and Goldberg, 2016) with the first-order graph parsing algorithm (McDonald et al., 2005a,b). Simple as it appears to be, BiAF has led to several record-breaking performances in multiple treebanks and languages (Dozat et al., 2017).

In their pioneering work, besides the neural architecture, Dozat and Manning (2017) adopt a simple head-selection training object (Zhang et al., 2017a) by regarding the original structured prediction task as a head-classification task in training. Although practically this simplification works well, there are still problems with it. Due to local normalization in the training objective (see §3.1.1), no global tree-structured information can be back-propagated during training. This can lead to the discrepancy between training and testing, since during testing, the MST (Maximum Spanning Tree) algorithm (McDonald et al., 2005b) is used to ensure valid tree structures. This problem raises concerns about the structured output layer. Several previous neural graph parsers utilized structured techniques (Pei et al., 2015; Kiperwasser and Goldberg, 2016; Zhang et al., 2016; Wang and Chang, 2016; Ma and Hovy, 2017), but their neural architectures might not be competitive to the current state-of-the-art BiAF parsing model. In this section, building upon the BiAF-based neural architecture, we empirically investigate the effectiveness of utilizing classical structured prediction techniques of output modeling for graph-based neural dependency parsing. Specifically, we focus on the *normalization* aspect of structured modeling and explore various normalization methods corresponding to different structural constraints. We empirically show that structured output modeling can obtain better performance, especially on sentence-level metrics. However, the improvements are modest, probably because neural models make the problem easier to solve locally.

3.1.1 Output Modeling

As discussed in Chapter 2, a common way to score the complex output structure is to factorize it into sub-structures, which is referred to as *factorization*. A further step of *normalization* is needed to compare different sub-structure choices and form the final score of an output structure. We will explain more details about these concepts in the situation of graph-based dependency parsing.

The output structure of dependency parsing is a collection of dependency edges forming a single-rooted tree. Graph-based dependency parsers factorize the outputs into specifically-shaped sub-trees (factors). Based on the assumption that the sub-trees are independent to each other, the score of the output tree structure (T) is the combination of the scores of individual sub-trees in the tree. In the simplest case, the sub-trees are the individual dependency edges connecting each modifier and its head word ((m, h)). This is referred to as first-order factorization (Eisner, 1996; McDonald et al., 2005a), which is adopted in (Dozat and Manning, 2017) and the neural parsing models in this work. There are further extensions to higher-order factors, considering more complex sub-trees with multiple edges (McDonald and Pereira, 2006; Carreras, 2007; Koo and Collins, 2010; Ma and Zhao, 2012).

After obtaining the individual scores of the sub-structures, we need to compute the score of the whole output structure. The main question is on what scale to normalize the output scores. For graph-based parsing, there can be mainly three options: Global, Local, or Single, following different structured output constraints and corresponding to different loss functions.

Global Global models directly normalize at the level of overall tree structures, whose scores are obtained by directly summing the raw scores of the sub-trees without any local normalization. This can be shown clearly if further taking a probabilistic CRF-like treatment, where a final normalization is performed over all possible trees:

$$\text{Score}_g(T) = \log \frac{\exp \sum_{(m,h)\in T} \text{Score}(m,h)}{\sum_{T'} \exp \sum_{(m,h)\in T'} \text{Score}(m,h)}$$

Here, the normalization is carried out in the exact output space of all legal trees (T'). Max-Margin (Hinge) loss (Taskar et al., 2004) adopts a similar idea, though there is no explicit normalization in its formulation. The output space can be further constrained by requiring the projectivity of the trees (Kübler et al., 2009). Several manual-feature-based (McDonald et al., 2005b; Koo and Collins, 2010) and neural-based dependency parsers (Pei et al., 2015; Kiperwasser and Goldberg, 2016; Zhang et al., 2016; Ma and Hovy, 2017) utilize global normalization.

Local Local models, in contrast, ignore the global tree constraints and view the problem as a head-selection classification problem (Fonseca and Aluísio, 2015; Zhang et al., 2017a; Dozat and Manning, 2017). The structured constraint that local models follow is that each word can be attached to one and only one head node. Based on this, the edge scores are locally normalized over all possible head nodes. This can be framed as the softmax output if taking a probabilistic treatment:

$$\text{Score}_l(T) = \sum_{(m,h)\in T} \log \frac{\exp \text{Score}(m,h)}{\sum_{h'} \exp \text{Score}(m,h')}$$

In this way, the model only sees and learns head-attaching decisions for each individual word. Therefore, the model is unaware of the global tree structures and may assign probabilities to non-tree cyclic structures, which are illegal outputs for dependency parsing. In spite of this defect, the local model enjoys its merits of simplicity and efficiency in training.

Single (Binary) If further removing the single-head constraint, we can arrive at an even more simplified binary-classification model for every single edge, referred to as the "Single" model, which predicts the presence and absence of dependency relation for every pair of words. Eisner (1996) first used this model in syntactic dependency parsing, and Dozat and Manning (2018) applied it to semantic dependency parsing. Here, the score of each edge is normalized against a fixed score of zero, forming a sigmoid output:

$$\text{Score}_s(T) = \sum_{(m,h)\in T} \log \frac{\exp \text{Score}(m,h)}{\exp \text{Score}(m,h) + 1}$$

Here, we only show the scoring formula for brevity. In training, since this binary classification problem can be quite imbalanced, we only sample part of the negative instances (edges). Practically, we find a ratio of 2:1 makes a good balance, that is, for each token, we use its correct head word as the positive instance and randomly sample two other tokens in the sentence as negative instances.

The normalization methods that we describe above actually indicate the output structured constraints that the model is aware of. The global model is aware of all the constraints to ensure a legal dependency tree. The local model maintains the single-head constraint while there are almost no structured constraints under the single model. To be noted, for all these normalization methods, we can take various loss functions. In this section, we study two typical ones: probabilistic Maximum-Likelihood loss (Prob), which requires actual normalization over the output space, and Max-Margin Hinge loss (Hinge), which only requires loss-augmented decoding in the same output space.

Normalization	Constraints	Loss	Algorithm
Single	NoMultiEdge	Prob	–
Local	+ SingleHeaded	Prob	–
Global-NProj	+ Acyclic	Prob Hinge	Matrix-Tree Theorem Chu-Liu-Edmonds
Global-Proj	+ Projective	Prob Hinge	Inside-Outside Eisner's

Table 3.1: Summarization of the explored methods and their corresponding algorithms.

Table 3.1 summarizes the methods (normalization and loss function) that we investigate in our experiments. For global models, we consider both Projective (Proj) and Non-Projective (NProj) constraints. Specific algorithms are required for probabilistic loss (a variation of Inside-Outside algorithm for projective (Paskin, 2001) and Matrix-Tree Theorem for non-projective parsing (Koo et al., 2007; Smith and Smith, 2007; McDonald and Satta, 2007)) and hinge loss (Eisner's algorithm for projective (Eisner, 1996) and Chu-Liu-Edmonds' algorithm for non-projective parsing (Chu and Liu, 1965; Edmonds, 1967; McDonald et al., 2005b)). For Single and Local models, we only utilize probabilistic loss, since in preliminary experiments we found hinge loss performed worse. No special algorithms other than simple enumeration are needed for them in training. In testing, we adopt non-projective algorithms for the non-global models unless otherwise noted.

3.1.2 Experiments

Settings

We evaluate the parsers on 14 treebanks: English Penn Treebank (PTB), Penn Chinese Treebank (CTB), and 12 selected treebanks from Universal Dependencies (v2.3) (Nivre et al., 2018a). We follow standard data-preparing conventions as in Ma et al. (2018). For PTB, we follow the dataset splitting convention: Sections 2-21 for training, Section 22 for validation, and Section 23 for testing. Dependency trees are obtained using the converter in Stanford Parser version 3.3.0. The POS tags were predicted using the Stanford POS tagger (Toutanova et al., 2003) with 10-fold jackknifing on the training data. For CTB, we follow the splitting of (Zhang and Clark, 2008), and the dependencies are converted using the Penn2Malt converter. Following previous work, gold segmentation and POS tags are used. For UD, we select 12 relatively large treebanks of UD version 2.3 (Nivre et al., 2018a) and also use gold POS tags.

For evaluation, we report LAS (Labeled Attachment Score) and LCM (Labeled Complete

Match). The evaluations on PTB and CTB exclude punctuations,[1] while on UD we evaluate on all tokens (including punctuations) as the setting of the LAS metric in the CoNLL shared tasks (Zeman et al., 2017, 2018).

For the neural architecture, we also follow the settings in Dozat and Manning (2017) and Ma et al. (2018) and utilize the deep BiAF model. For the input, we concatenate representations of words, part-of-speech (POS) tags, and characters. Word embeddings are initialized with the pre-trained fasttext word vectors[2] for all languages. For POS tags and Character information, we use POS embeddings and a character-level Convolutional Neural Network (CNN) for the encoding. For the encoder, we adopt three layers of bi-directional LSTM to get contextualized representations, while our decoder is the deep BiAF scorer as in Dozat and Manning (2017). We only slightly tune hyper-parameters on the Local model and the development set of PTB and then use the same ones for all the models and datasets. Note that our exploration only concerns the final output layer which does not contain any trainable parameters in the neural model, and all our comparisons are based on exactly the same neural architecture and hyper-parameter settings. Only the output normalization methods and the loss functions are different.

We run all the experiments with our own implementation, which is written with PyTorch. All experiments are run with one TITAN-X GPU. In training, global models take around twice the time of the local and single models; while in testing, their decoding costs are similar.

Results

Method	Single	Local	Global-NProj		Global-Proj	
	Prob	Prob	Prob	Hinge	Prob	Hinge
PTB	93.43/44.67	93.75/46.65	93.84†/47.17	93.91†/47.78†	93.79/47.16	**93.96**†/**48.47**†
CTB	87.03/31.26	88.16/33.16	88.26/33.73	87.92/32.77	**88.46**†/**35.11**†	88.14/34.00†
bg-btb	89.97/39.25	90.06/39.99	90.35†/**41.25**†	**90.42**†/40.83	90.15/40.98	90.20/40.53
ca-ancora	91.23/25.03	91.54/26.35	91.73†/27.19†	**91.73**†/26.65	91.39/**27.39**†	91.51/27.19†
cs-pdt	90.95/43.07	91.51/45.62	**91.69**†/**46.60**†	91.52/46.02†	91.10/44.43	91.18/44.02
de-gsd	**83.68**†/22.65	83.43/22.42	83.65†/22.86	83.66†/22.93	83.39/23.37†	83.63/**23.51**†
en-ewt	88.01/55.93	88.33/56.46	88.52†/57.29†	**88.59**†/57.33†	88.52†/**58.29**†	88.41/57.31†
es-ancora	90.82/27.27	91.05/27.41	91.12/27.89	**91.14**/27.35	90.84/**28.41**†	91.03/27.70
fr-gsd	88.00/20.03	88.13/20.83	88.43†/21.71	88.22/20.27	**88.59**†/**23.80**†	88.41†/21.88
it-isdt	91.71/44.05	92.01/44.26	92.16/45.30	92.08/45.02	**92.49**†/**48.27**†	92.37†/46.75†
nl-alpino	88.31/33.11	88.81/33.67	**88.94**/34.62	88.94/**35.12**†	88.37/33.05	88.45/33.00
no-bokmaal	92.89/53.60	92.89/53.58	**93.02**†/**54.36**†	92.78/53.09	92.82/53.57	92.70/52.71
ro-rrt	85.10†/12.85†	84.58/11.57	84.85†/12.44	85.04†/13.03†	84.89†/12.94†	**85.16**†/**13.76**†
ru-syntagrus	92.76/48.67	93.29/50.69	**93.36**†/**50.97**	93.29/50.72	93.11/50.79	93.19/50.17
Average	89.56/35.82	89.82/36.62	**89.99**†/37.39†	89.95†/37.07†	89.85/**37.68**†	89.88/37.21†

Table 3.2: Results (LAS/LCM) on the test sets (averaged over three runs).

We run all the models three times with different random initialization, and the averaged results on the test sets are shown in Table 3.2. Here, '†' means that the result of the model

[1]Tokens whose gold POS tag is one of {“ ” : , .} for PTB or “PU” for CTB

[2]https://fasttext.cc/docs/en/pretrained-vectors.html

is statistically significantly better (by permutation test, $p < 0.05$) than the Local-Prob model. Overall, the global models[3] perform better consistently, especially on the metrics of Complete Match, showing the effectiveness of being aware of global structures. However, the performance gaps between global models and local models are small. More surprisingly, the single models that ignore all the structures only lag behind by around 0.4 on average. In some way, this shows that input modeling, including the distributed input representations, contextual encoders, and parts of the decoders, makes the structured decision problem easier to solve locally. Neural models seem to squeeze the improvement space that structured output modeling can bring.

Analysis

We further analyze output constraints and input modeling. For brevity, we only analyze PTB and use probabilistic models. Single models are excluded for their poorer performance.

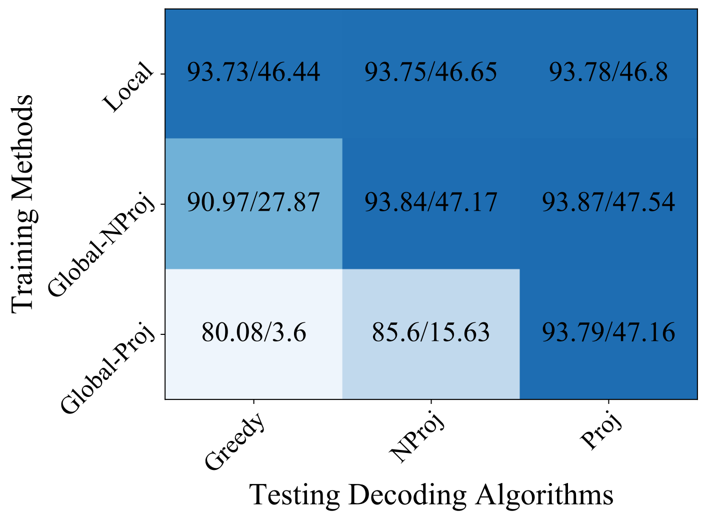


Figure 3.1: Results (LAS/LCM), on the PTB test set) of different models (with prob loss) and decoding algorithms.

Firstly, we study the influence of output constraint differences in training and testing. Here, we include a naive "Greedy" decoding algorithm that simply selects the most probable head for each token. This does not ensure that the outputs are trees and corresponds to the head-classification method adopted by local models. The results of different models and training/testing algorithms are shown in Figure 3.1. Here, rows represent the methods used in training and columns denote the decoding algorithms in testing. Darker colors represent better scores. Interestingly, the discrepancies in training and testing are only detrimental when the output constraint in testing is looser than that in training (the left corner in the figure), as shown by the poorer results in the training-testing pairs of "NProj-Greedy", "Proj-Greedy" and "Proj-NProj". Generally, projective decoding is the best choice since PTB contains mostly (99.9%) projective trees.

[3]Projective global models perform averagely poorer than non-projective ones since some of the treebanks (for example, only 88% of the trees in 'cs-pdt' are projective) contain a non-negligible portion of non-projective trees.

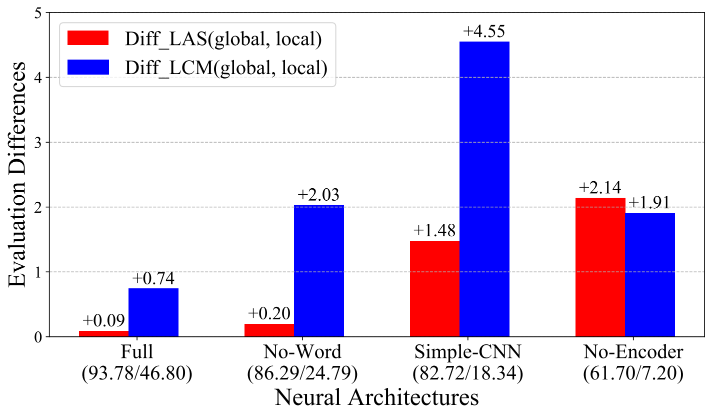


Figure 3.2: Evaluation differences (on the PTB test set) between global and local methods when adopting various "weaker" neural architectures.

Next, we study the interactions of "weaker" neural architectures (for input modeling) and output modeling. We consider three "weaker" models: (1) "No-Word" ignores all the lexical inputs and is a pure delexicalized model; (2) "Simple-CNN" replaces the RNN encoder with a much simpler encoder, which is a simple single-layer CNN with a window size of three for the purpose of studying weak models; (3) "No-Encoder" completely deletes the encoder, leading to a model that does not take any contextual information. Here, since we are testing on PTB which almost contains only projective trees, we use projective decoding for all models. As shown in Figure 3.2 (numbers below x-axis labels denote the evaluation scores (LAS/LCM) of the local models), when input modeling is weaker, the improvements brought by the global model generally get larger. Here, the LCM for "No-Encoder" is an outlier, probably because this model is too weak to get reasonable complete matches. The results show that with weaker input modeling, the parser can generally benefit more from structured output modeling. In some way, this also indicates that better input modeling can make the problem depend less on the global structures so that local models are able to obtain competitive performance.

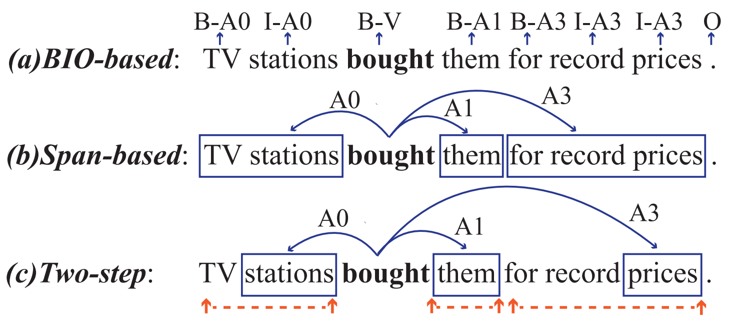
3.1.3 Discussion

We only explore first-order graph parsing in this section, that is, for the *factorization* part, we do not consider high-order sub-subtree structures. Recently, there has been work exploring the effectiveness of higher-order parsing together with neural models. Falenska and Kuhn (2019) suggest that with powerful neural models, second-order features seem redundant. Fonseca and Martins (2020) show that the higher-order features provide small gains, but they are consistent for long-range dependencies and long sentences. Zhang et al. (2020b) provide an efficient implementation for second-order TreeCRF parser and show that it can bring improvements. Nevertheless, the helpfulness of higher-order modeling does seem modest if utilizing powerful neural models that capture rich input features, which in some way is consistent with our results on the modeling of constraints.

In this section, we call the models that are aware of the whole output structures "global". In fact, with the neural architecture that can capture features from the whole input sentence, actually, all the models we explore have a "global" view of inputs. Our experiments show that with this kind of global input modeling, good results can be obtained even when ignoring certain output structures, and further enhancement of global output structures only provides small benefits. This might suggest that input and output modeling can capture certain similar information and have overlapped functionalities for structured decisions.

3.2 Span Extraction in Semantic Role Labeling

In this section, we take a look at the task of semantic role labeling (SRL), a core natural language processing (NLP) task that aims to identify predicate-argument structures in text (Gildea and Jurafsky, 2002; Palmer et al., 2010). Following the neural encoder-decoder paradigm, we can view an SRL model as combining an encoder, which builds hidden representations for the input words, with a decoder, which extracts the argument spans based on the encoded representations. While recent SRL models achieve high performance on popular benchmarks (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Strubell et al., 2018; Shi and Lin, 2019), most improvements come from better neural encoders, such as the Transformer (Vaswani et al., 2017) and pre-trained contextualized word representations, such as BERT (Devlin et al., 2019). However, the influence on end-task performance due to the choice of the decoder has become less clear.



B-A0 I-A0 B-V B-A1 B-A3 I-A3 I-A3 O
(a)BIO-based: TV stations bought them for record prices .
A0
A1
A3
(b)Span-based: TV stations bought them for record prices .
A0
A1
A3
(c)Two-step: TV stations bought them for record prices .

Figure 3.3: Illustration of the explored decoding methods in this section.

We perform an empirical investigation of different factorization methods for span extraction, as illustrated in Figure 3.3. Here, for the predicate "**bought**", we identify argument spans by (a) BIO-based sequence labeling; (b) direct span-based extraction; (c) two-step approach: first identifying head words, then expanding to full spans by deciding left and right boundaries.

The most common strategy casts the task as a sequence labeling problem using the BIO-tagging scheme (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Strubell et al., 2018; Shi and Lin, 2019). While this approach is simple, it does not directly model the arguments at the span level. Alternatively, the span-based method directly builds representations for all possible[4] spans and selects among them (He et al., 2018a; Ouchi et al., 2018). Though this approach is straightforward for explicitly modeling span-level information, composing a representation for

[4]Up to a fixed length, decided as a hyperparameter.

every span can lead to higher computational costs. Inspired by dependency-based SRL (Surdeanu et al., 2008; Hajič et al., 2009), a third option first identifies a head word and then decides the span boundaries. This two-step strategy has been explored in previous work on information extraction (Peng et al., 2015; Lin et al., 2019b; Zhang et al., 2020c), and we apply it here to SRL. Compared with the sequential BIO-tagger, the latter two approaches more directly model the argument span structures; we thus refer to them as more *structured* decoders.

We perform careful comparisons of these decoding methods upon the same encoding backbone, based on a deep Transformer encoder. We first experiment in the standard fully-supervised settings on English PropBank datasets (CoNLL-2005 and CoNLL-2012). The results show that more structured decoders, especially the two-step approach with syntactic guidance, consistently perform better than BIO-tagging when using static word embeddings. However, if including strong contextualized BERT embeddings, the benefits of more structured decoding are diminished and the simplest BIO-tagging method performs well across different experimental settings. Error analysis shows that contextualized embeddings help in deciding span boundaries. We also perform speed comparisons and analyze the accuracy-efficiency trade-offs among different decoding methods.

3.2.1 Model

For a given predicate, SRL aims to extract all argument spans and assign them role labels. To model this task, we follow the neural encoder-decoder paradigm: the encoder produces hidden representations for the input words, upon which the decoder decides the structured outputs. All our models adopt the same encoding architecture: a deep Transformer encoder (Vaswani et al., 2017), which has been shown effective for SRL (Tan et al., 2018; Strubell et al., 2018). For a given input sequence of words $\{w_1, \ldots, w_n\}$, we obtain their contextualized representations $\{h_1, \ldots, h_n\}$ from the encoder. Upon these, we stack different decoders to extract the argument spans corresponding to different extraction strategies, which will be described in the following.

1) BIO-based

Since argument spans do not overlap in the datasets we explore, the BIO-tagging scheme (Ramshaw and Marcus, 1995) can be utilized to extract them, casting SRL as a sequence labeling problem.

For each word, we feed its representation h to a multi-layer perceptron (MLP) based scorer, which assigns the scores of the BIO tags. Assuming that we have k possible argument roles in the output space, each of them will have its "B-" and "I-" tags. Together with the "O" (NIL) tag, the tagging space has a dimension of $2k + 1$.

Furthermore, we consider the option of adopting a standard linear-chain conditional random field (CRF; Lafferty et al., 2001) to model pairwise tagging transitions. If adopting the CRF (BIO w/ CRF), we train the model with sequence-level negative log-likelihood and use the Viterbi algorithm for inference. If not using the CRF (BIO w/o CRF), we simply use tag-level cross entropy as the learning objective and perform argmax greedy decoding at inference time, following Tan et al. (2018).

2) Span-based

In the span-based method, we build neural representations for all candidate spans and directly select and assign role labels (or NIL). Following He et al. (2018a), for a span a, we compose its representation from start and end points, soft head-word vectors, and span width features by concatenation:

$$\mathbf{g}(a) = [h_{start(a)}, h_{end(a)}, \text{soft}(a), \text{width}(a)]$$

Here, soft(a) denotes a soft-head representation obtained from an attention mechanism:

$$\text{soft}(a) = \sum_{start(a) \leq i \leq end(a)} \text{att}(i, a) h_i \qquad \text{att}(i, a) = \frac{w_{att}^T h_i}{\sum_{start(a) \leq i' \leq end(a)} w_{att}^T h_{i'}}$$

and width(a) denotes a width embedding corresponding to the span size (width).

All valid candidate spans are first assigned an unlabeled score, using an MLP scorer. This unary score is then used as the criterion for beam pruning to reduce the computational costs of full labeling. Since each predicate will not have too many arguments (most have less than 5), we adopt a fixed beam size of 10. We also limit the maximum width of candidate spans to 30, which covers around 99% of the cases. Surviving candidates are further assigned label scores with another MLP scorer, with which we decide output arguments.

3) Two-step

In this approach, we decompose the problem into two steps: head selection and boundary decision. In the first step, each individual word is directly scored for argument labels (or NIL). We again adopt an MLP classifier to obtain the probability that a word can be the head of an argument with label r (r can be NIL). The non-NIL labeled words are selected as the head words of the arguments. Since the annotations usually do not contain head words for the argument spans, we further consider two strategies to provide supervision for training:

HeadSyntax A straightforward method is to adopt guidance from syntax. Following dependency-style SRL (Surdeanu et al., 2008; Hajič et al., 2009), we use syntactic dependency parse trees and select the highest word (the one that is closest to the root) in the span as the head. In training, we only assign the argument role to the syntactic head word, and all other words in the span get a label of NIL.

HeadAuto In this strategy, all words in an argument span can be considered as the potential head word. We adopt the *bag loss* from Lin et al. (2019b) to train the model to automatically identify head words. Specifically, for a word w_i inside an argument span a which has the role r, the loss is computed as:

$$\text{Loss}(w_i) = \delta_i \cdot [-\log p(r|h_i)] + (1 - \delta_i) \cdot [-\log p(\text{NIL}|h_i)]$$
$$\delta_i = \frac{p(r|h_i)}{\max_{start(a) \leq j \leq end(a)} p(r|h_j)}$$

Here, words that are more indicative of the argument will be assigned higher probabilities to the argument role. This will give them larger loss weights (δ) and thus further encourage them to be the heads. In this way, the head words are decided automatically by the model.

In the second step, we determine span boundaries for these head words. Here we adopt the span selection method from extractive question answering (Wang and Jiang, 2016; Devlin et al., 2019) using two classifiers to decide the start and end words ($[s, e]$) of a span:

$$p(s, e) = p_{start}(s) \cdot p_{end}(e)$$
$$p_{start}(s) = \frac{\exp \text{score}_{start}(h'_s)}{\sum_i \exp \text{score}_{start}(h'_i)} \qquad p_{end}(e) = \frac{\exp \text{score}_{end}(h'_e)}{\sum_i \exp \text{score}_{end}(h'_i)}$$

Here, we first add indicator embeddings to the head word's encoder representations to mark its positions, and then stack one self-attention layer to obtain head-word-aware representations for the input sequence: $\{h'_1, \cdots, h'_n\}$. We further introduce two linear scorers to assign the start and end scores for each word, which are further normalized across the input sequence. For training, the objective is minimizing the sum of negative log-likelihoods of picking the correct start and end positions. When decoding, we select the maximum scoring span whose boundaries s and e satisfy $s \leq e$. We observe that at inference time, sometimes different head words may expand to overlapping spans, which do not appear in the datasets we explore. To deal with this, we adopt a greedy post-processing procedure to remove overlapping argument spans: iterating through all argument spans ranked by model score and only keeping the ones that do not overlap with previous surviving ones.

3.2.2 Experiments

Settings

Data The models are evaluated on standard PropBank datasets from the CoNLL-2005 shared task (Carreras and Màrquez, 2005) and the CoNLL-2012 subset of OntoNotes 5.0 (Pradhan et al., 2013). Table 3.3 lists the relevant statistics. For CoNLL-2005, we follow the splits from the CoNLL-2005 shared task.[5] For the English part of CoNLL-2012, we adopt the data from Pradhan et al. (2013)[6] but follow the splits of the CoNLL-2012 shared task.[7] For evaluation, we adopt the standard evaluation script of `srl-eval.pl`.[8] For the "HeadSyntax" method that requires dependency trees, we convert the original constituencies to Universal Dependencies (Nivre et al., 2020) using Stanford CoreNLP (Manning et al., 2014) version 4.1.0. Notice that we only need syntactic information to be provided during training, since the model predicts head words itself at test time.

Input Features and Encoder For a fair comparison, we adopt the same input features, deep Transformer-based encoders, and training schemes across all experiments. We consider two

[5] https://www.cs.upc.edu/~srlconll/
[6] https://cemantix.org/data/ontonotes.html
[7] https://conll.cemantix.org/2012/
[8] https://www.cs.upc.edu/~srlconll/soft.html

	CoNLL 2005				CoNLL 2012		
	Train	Dev	Test	Brown	Train	Dev	Test
Sent.	39.8k	1.3k	2.4k	0.4k	75.2k	9.6k	9.5k
Pred.	90.8k	3.2k	5.3k	0.8k	188.9k	23.9k	24.5k
Arg.	333.7k	11.7k	19.6k	3.0k	622.5k	78.1k	80.2k

Table 3.3: Statistics of the datasets: Number of sentences (Sent.), predicates (Pred.) and arguments (Arg.).

types of word features: static word embeddings and pre-trained contextualized embeddings[9] from BERT$_{\text{base}}$. We adopt fastText[10] embeddings (Mikolov et al., 2018) and frozen features from `bert-base-cased`. Before feeding the word-level features to the encoder, we concatenate them and apply a linear layer to project them to the encoding dimension. We further add indicator embeddings to let the model be aware of the positions of the predicates. For both cases of static embedding and BERT features, we adopt a 10-layer Transformer module as the encoder. The head number, model dimension, and feed-forward dimension are set to 8, 512, and 1024, respectively. In addition, we adopt relative positional encodings for the Transformer (Shaw et al., 2018) since we found slightly better performance in preliminary experiments.

Training We use the Adam optimizer (Kingma and Ba, 2014) for training. The learning rate is linearly increased towards 2e-4 with the first 8k steps for warming up. After this, we decay the learning rate by 0.75 each time the performance on the development set does not increase for 10 checkpoints. We train the model for a maximum of 150k steps and do validation every 1k steps to select the best model. One model contains around 40M parameters (excluding BERT). For each update, the batch size is around 4096 tokens. We apply dropout rates of 0.2 to the hidden layers. For models using static embeddings, we further replace input words with a special UNK token with a probability of 0.5 if it appears less than 3 times in the training set. At test time, a word is represented by UNK if it is not found in the collection of static word embeddings. All the models are trained and evaluated on one TITAN-RTX GPU, and training one model takes around 1 day in our environment.

Performance Comparisons

Table 3.4 lists the comparisons of our test results (BIO w/ CRF using BERT features) to previous work. Generally, our model can obtain comparable results, which verifies the quality of our implementation.

Table 3.5 lists our main comparisons on the test sets. The overall trends are very similar. For BIO-tagging, incorporating a structured CRF layer is generally helpful, which can improve the F1 scores by around 0.5 points. When not using BERT features, more structured decoders

[9]We concatenate layers 7, 8, and 9 of BERT hidden representations. For words that are split into sub-tokens, we utilize the representations of the first sub-token.

[10]`https://fasttext.cc/docs/en/english-vectors.html`

Model	WSJ	Brown	OntoNotes
He et al. (2018a)	87.4	80.4	85.5
Ouchi et al. (2018)	87.6	78.7	86.2
Shi and Lin (2019)	**88.8**	82.0	86.5
Ours (BIO w/ CRF)	87.9	**82.1**	**86.6**

Table 3.4: Comparisons of F1 scores with previous work in the fully-supervised settings (with single model).

	CoNLL 2005 In-domain (WSJ)			Out-of-domain (Brown)			CoNLL 2012 (OntoNotes)		
	P	R	F1	P	R	F1	P	R	F1
Without BERT									
BIO (w/o CRF)	84.42	84.94	84.68$_{\pm 0.25}$	73.56	73.03	73.29$_{\pm 0.43}$	81.74	82.98	82.35$_{\pm 0.24}$
BIO (w/ CRF)	85.04	85.35	85.20$_{\pm 0.12}$	74.25	73.92	74.08$_{\pm 0.31}$	82.79	**84.11**	83.44$_{\pm 0.21}$
Span	85.68	84.62	85.14$_{\pm 0.32}$	75.88	74.23	75.05$^{\dagger}_{\pm 0.42}$	83.42	83.49	83.46$_{\pm 0.15}$
HeadSyntax	**85.84**	**85.38**	**85.61**$^{\dagger}_{\pm 0.11}$	**75.92**	**74.74**	**75.33**$^{\dagger}_{\pm 0.58}$	**83.55**	83.82	**83.68**$^{\dagger}_{\pm 0.11}$
HeadAuto	85.30	85.17	85.23$_{\pm 0.14}$	74.98	73.85	74.41$_{\pm 0.50}$	83.09	83.71	83.40$_{\pm 0.09}$
With BERT									
BIO (w/o CRF)	87.21	87.95	87.58$_{\pm 0.28}$	81.26	81.79	81.52$_{\pm 0.23}$	85.33	86.97	86.14$_{\pm 0.10}$
BIO (w/ CRF)	87.54	**88.32**	**87.93**$_{\pm 0.16}$	81.91	**82.37**	**82.14**$_{\pm 0.20}$	85.93	**87.32**	**86.62**$_{\pm 0.14}$
Span	87.75	87.33	87.54$_{\pm 0.14}$	81.87	81.60	81.73$_{\pm 0.77}$	85.97	86.26	86.12$_{\pm 0.09}$
HeadSyntax	**87.76**	87.96	87.86$_{\pm 0.08}$	**82.10**	81.60	81.85$_{\pm 0.90}$	**86.17**	86.77	86.47$_{\pm 0.10}$
HeadAuto	87.70	88.15	**87.93**$_{\pm 0.12}$	81.52	81.36	81.44$_{\pm 0.37}$	86.00	86.84	86.42$_{\pm 0.09}$

Table 3.5: Main test results. All the results are averaged over five runs with different random seeds, and standard deviations of the F1 scores are also reported.

generally perform better than BIO-tagging. With the head word oracles from the syntax trees, "HeadSyntax" performs the best overall. This agrees with Strubell et al. (2018) and Swayamdipta et al. (2018), showing the helpfulness of syntactic information for SRL. However, when utilizing BERT features, the benefits of more structured decoders are diminished and the simple BIO-tagger robustly performs well. It seems that with a powerful encoder, the choice of the decoder plays a smaller role in the final performance.

To further investigate this phenomenon, we perform error analysis on the development outputs of "BIO (w/ CRF)" and "HeadSyntax," which are the two that perform the best overall. We group the errors into four categories: "Boundary" denotes that the predicted head words and role labels match the gold ones but the span boundaries are incorrect; "Label" denotes that the predicted spans are correct but the role labels are wrong; "Attachment" denotes the errors caused by incorrect phrase attachments, while "Others" denotes the remaining errors, which are other missing and over-predicted arguments. The results are shown in Figure 3.4. When not using BERT features, the main advantages of "HeadSyntax" over "BIO" are on the "Boundary" and "Attachment" errors, where the former makes 11% fewer "Boundary" and 17% fewer "Attachment" errors. Notice that these two types of errors are closely related to syntax, and they are

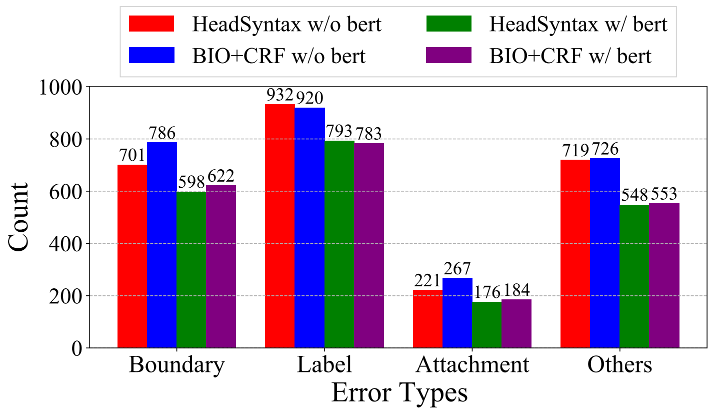


Figure 3.4: Error breakdown for "BIO" and "HeadSyntax" on the CoNLL-2005 development set.

mainly caused by incorrect phrase boundary predictions. In this way, it seems natural that incorporating syntactic information with head words can be helpful in this scenario. Nevertheless, when utilizing BERT features, these advantages are reduced to a negligible level. This indicates that BERT may provide sufficient information overlapping with syntax to help with boundary decisions.

Speed Comparisons

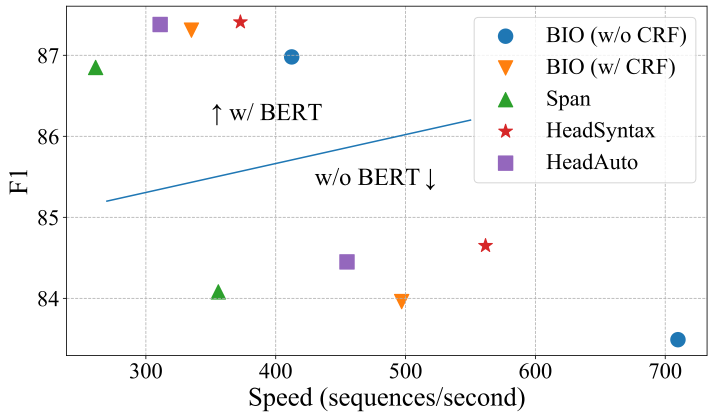


Figure 3.5: Comparing speed vs. F1 with different decoding methods (on CoNLL05 development set).

Finally, we compare the decoding speed of different extraction methods against F1 scores in Figure 3.5. Greedy BIO-tagging (w/o CRF) obtains the highest speed. However, this comes with a drop of around 0.5 F1 points without BERT and 0.3 F1 points with BERT. Although the two-step approaches require two decoding steps, they are still efficient thanks to the simplicity of both steps. When trained with syntactic information, this model is the second best in terms of decoding speed. On the other hand, even with beam pruning, the span-based decoder still needs to score a number of span candidates quadratic in the input sequence length, making it less efficient compared to other decoders.

3.2.3 Related Work and Discussion

Argument Extraction Before the incorporation of end-to-end neural models, traditional SRL systems usually depend on input constituency trees to obtain argument candidates (Xue and Palmer, 2004; Màrquez et al., 2008). Although straightforward, this may suffer from error propagation from syntax parsers. Recent neural systems utilize end-to-end models to solve the task. Casting SRL as BIO-based sequence labeling problem is the most common decoding scheme and can obtain impressive results (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Strubell et al., 2018; Shi and Lin, 2019). On the other hand, span-based methods (He et al., 2018a; Ouchi et al., 2018) directly select and label among argument span candidates. This is actually similar to the traditional approaches, though the argument candidates are obtained by the model rather than from input syntax trees. In addition to span-based SRL, the focus of this section, there is another category of dependency-style SRL, which only requires the extraction of head words of argument spans (Surdeanu et al., 2008; Hajič et al., 2009). Inspired by this, for span-based SRL, we can extract argument head words as the first step and then expand to the full spans in the second step. This idea has also been applied in information extraction, such as coreference resolution (Peng et al., 2015), entity detection (Lin et al., 2019b), and event argument extraction (Zhang et al., 2020c). Another interesting direction is considering the structured constraints of the arguments, including work on integer linear programming (Punyakanok et al., 2004, 2008), dynamic programming (Täckström et al., 2015) and structure-aware tuning (Li et al., 2020d).

Syntax and SRL There has been discussion of the relation between syntax and SRL (Gildea and Palmer, 2002; Punyakanok et al., 2008), considering the close connections between these two tasks. Though syntax trees are usually the inputs to traditional SRL systems, recent work finds that syntax-agnostic neural models also work well (Marcheggiani et al., 2017; Cai et al., 2018). Nevertheless, with recent neural models, syntax information has still been found helpful for SRL in various ways, including multi-task learning (Swayamdipta et al., 2018; Strubell et al., 2018), argument pruning (He et al., 2018b), and tree-based modeling (Marcheggiani and Titov, 2017; Li et al., 2018; Marcheggiani and Titov, 2020). In this work, our “HeadSyntax” decoder incorporates syntax in a partial way, utilizing dependency trees to decide the head words in training. This method indeed performs the best overall if only adopting static word embeddings. However, the incorporation of BERT features diminishes the advantages. This indicates that BERT may already cover much of the syntactic (surface) features of the input sentences, as suggested by recent work on BERT interpretation (Goldberg, 2019; Hewitt and Manning, 2019; Tenney et al., 2019; Clark et al., 2019).

In this section, we empirically compare several span extraction methods for SRL. Extensive results show that in fully supervised settings, simple BIO-tagging is a robustly good option when utilizing BERT features. We also analyze the accuracy-efficiency trade-offs for different decoders; although methodologically more complex, two-step approaches are still efficient in decoding.

3.3 Conclusion

In this chapter, we provide an empirical investigation of the roles of structured output modeling with two study cases. Specifically, we explore the *normalization* aspect in dependency parsing with different structural constraints incorporated in the modeling and the *factorization* aspect in semantic role labeling with different ways to perform span extraction. Generally, we find that with powerful neural models that could capture the full input contexts, the benefits of better output structured modeling become modest. Nevertheless, we still observe improvements when evaluated on full-structure metrics and when utilizing less powerful but more efficient models.

The scenarios studied in this chapter are mainly fully-supervised, that is, we assume abundant training instances. A natural question is what are the roles of structured modeling in resource-limited scenarios and how does it interact with the amount of available resources? We will further investigate this topic in the next chapter.

Chapter 4

On the Interactions of Structural Constraints and Data Resources for Structured Prediction

In the previous chapter, we empirically investigate the costs and benefits of structured output modeling in fully-supervised settings. A natural question to ask is what role it will play in low-resource scenarios. In this chapter, we provide an analysis of the interactions of the effectiveness of decoding with structural constraints and the amount of available training data for structured prediction tasks in NLP.

4.1 Introduction

Recently, neural models, especially those based on pre-trained contextualized representations, have brought impressive improvements for a variety of structured prediction tasks in NLP (Devlin et al., 2019; Kulmizev et al., 2019; Shi and Lin, 2019; Li et al., 2020c). More interestingly, the incorporation of powerful neural models seems to decrease the potential benefits brought by more complex structured output modeling. For example, for sequence labeling, it has been shown that reasonably good performance could be obtained even without any explicit modeling of the interactions of the output tags (Tan et al., 2018; Devlin et al., 2019). For dependency parsing, models that ignore tree constraints and cast the problem as head selections in training can still obtain impressive results (Dozat and Manning, 2017). Most of these previous results are obtained in fully supervised settings. While they show that with abundant training signals, better input modeling and representation learning could shadow the benefits brought by more complex structured modeling, it remains unclear for the cases where data resources are limited.

One of the most salient and important properties of structured prediction is that the output objects should follow specific structural constraints. For example, the output of a syntactic parser should be a well-formed tree and the output labels of an information extraction system need to follow certain type restrictions. In this chapter, we focus on the facet of structural constraints and explore their influence on structured prediction problems under scenarios with different amounts of training data. On the one hand, since we know the target outputs should conform to certain constraints, *explicitly* enforcing these constraints will likely bring benefits and sometimes even be a requirement. On the other hand, as neural models are developed to better represent input contexts, they might already be able to *implicitly* capture the output constraints by learning from the data. Especially, if given enough training data, it would be unsurprising that the model could directly produce outputs that conform to the constraints without explicit enforcement since the training instances are presented in such ways.

On the interactions of explicit incorporation of constraints and the amounts of training data, we ask the following three research questions (RQ), which we aim to explore in this chapter:

RQ1: What is the influence of constraints with different amounts of training data?
With powerful neural networks and abundant training data, the model can be trained to implicitly capture structural constraints even without explicit enforcement. Nevertheless, it still remains unclear for the cases with limited data. We aim to explore how the incorporation of constraints influences the outputs and how such influences change with different amounts of training data.

RQ2: What is the influence of constraints when using more efficient models?
Although neural models can obtain impressive results, one shortcoming is that they are usually computationally expensive. Recently, there has been work on improving model efficiency. Knowledge distillation is one of the most widely-utilized methods, learning a smaller student model from a larger teacher model (Kim and Rush, 2016; Sanh et al., 2019; Jiao et al., 2020). An interesting question to explore is how these more efficient models interact with the explicit incorporation of structural constraints.

RQ3: What is the influence of constraints for out-of-domain generalization?
We usually expect the model to be able to generalize to scenarios that can be different from those represented by the training data, for example, to different domains or text genres. It will be interesting to explore how the constraints influence predictions for these cases and especially whether there are specific patterns with regard to the discrepancies between the source and the target.

To answer these questions, we conduct extensive experiments on three typical structured prediction tasks, including named entity recognition (NER), dependency parsing (DPAR), and an information extraction task of event argument extraction (EAE). We find that models trained with less training data tend to produce outputs that contain more structural violations when using constraint-agnostic greedy decoding. Further applying constrained decoding brings consistent performance improvements and the benefits are more prominent in fewer data scenarios (§4.3.2). A similar trend can be found with regard to model sizes: smaller models tend to output more violations with greedy decoding and benefit more from constrained decoding (§4.3.3). Finally, in cross-genre settings, we find a weak pattern with regard to genre discrepancies: more structural violations tend to be made with greedy decoding when transferring to more distant genres (§4.3.4).

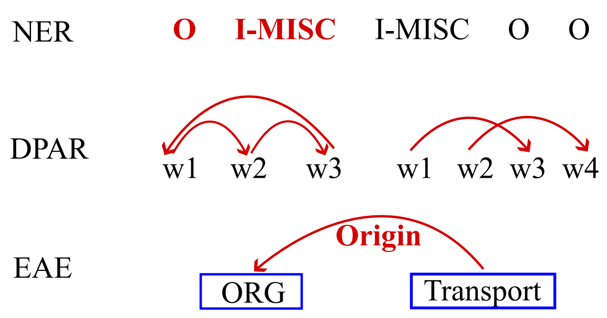


Figure 4.1: Examples of structural violations (marked in red).

4.2 Tasks and Models

4.2.1 Named Entity Recognition

Our first task is named entity recognition (NER), which aims to extract entity mentions from raw texts and can be typically cast as a sequence labeling problem. We adopt a simple NER model that utilizes a pre-trained BERT model as the encoder and a softmax layer to predict the output tags. We adopt the typical BIO tagging scheme (Ramshaw and Marcus, 1995), specifying tags for the **B**eginning, the **I**nside, and the **O**utside of an entity span.

More specifically, for an input sequence of words $[w_1, w_2, \cdots, w_n]$, our model aims to assign a sequence of BIO tags $[t_1, t_2, \cdots, t_n]$ for them. The probability of each output tag is locally normalized for each word:

$$p(t_i|w_i) = \frac{\exp \text{score}(t_i|w_i)}{\sum_{t' \in \mathcal{T}} \exp \text{score}(t'|w_i)}$$

Here, the function of score is realized as a linear layer stacked upon the word representations[1] and $\mathcal{T}$ denotes the output tag space.

With the BIO tagging scheme, there are hard constraints between tags of consecutive tokens: the **I** tag must follow a **B** or **I** tag of the same entity type. For example, the tagging sequence of "O I-MISC I-MISC O O" is erroneous because the transition of "O $\rightarrow$ I-MISC" is illegal. One solution to mitigate this problem is to forbid such illegal transitions in decoding. This can be achieved by incorporating a transition matrix $M \in \mathcal{R}^{|\mathcal{T}| \times |\mathcal{T}|}$, where the entries corresponding to illegal tag transitions are filled with $-\infty$ and the legal ones are filled with 0. For the decoding process, we define the score of a tag sequence as:

$$s(t_1, t_2, \cdots, t_n) = \sum_i \log p(t_i|w_i) + \sum_i M_{t_i, t_{i+1}}$$

In this way, the highest-scored tag sequence will not contain transition violations. This decoding problem can be solved by the Viterbi algorithm (Viterbi, 1967). If not enforcing these constraints,

[1]If a word is split into multiple tokens, we simply take its first sub-token.

the second term of the sequence score can be dropped and the decoding will be greedily finding the maximally-scored tag for each token individually.

Notice that this treatment resembles the conditional random field (CRF) based models (Lafferty et al., 2001), whereas the main difference is that we utilize a locally normalized model, and the transition matrix is manually specified to exclude illegal transitions. In our preliminary experiments, we also tried CRF models but did not find obvious benefits compared with the local models when adopting the same underlying pre-trained model.

4.2.2 Dependency Parsing

We further consider dependency parsing (DPAR) (Kübler et al., 2009), which aims to parse the input sentence into well-formed tree structures. We adopt the widely utilized first-order graph-based parser (McDonald et al., 2005a). Similar to NER, we adopt the pre-trained BERT encoder to provide the contextualized representations for the input tokens and stack a biaffine scorer Dozat and Manning (2017) to assign scores for the dependency edges. For training, we adopt a local model which views the problem as a head-finding classification task for each input token (Dozat and Manning, 2017; Zhang et al., 2017a). At testing time, we further consider tree constraints with specific decoding algorithms. Since we are mainly interested in structural tree constraints, we only perform unlabeled parsing.

More specifically, for an input sequence of words $[w_1, w_2, \cdots, w_n]$, we aim to find the dependency head words $[h_1, h_2, \cdots, h_n]$ for the input word sequence. With local normalization, this can be viewed as a head classification problem:

$$p(h_i|w_i) = \frac{\exp \text{score}(h_i|w_i)}{\sum_{h' \in \{R, w_1, w_2, \cdots, w_n\}} \exp \text{score}(h'|w_i)}$$

Here we add an artificial target R to the output space to cover the cases of the root nodes. The score function is realized with a biaffine module that produces head-modifier scores for the input pair of words.

We consider two constraints for the output structures. First, there should not be any cycles in the output graphs, otherwise, they will not be trees. Moreover, we consider the projective constraint[2], which specifies that there are no edges that cross each other. We adopt Eisner's algorithm (Eisner, 1996) for the constrained decoding, which is a dynamic programming algorithm that searches the highest-scored trees in the constrained output space. If not considering any of these constraints, we greedily predict the head word for each token based on the head classification probabilities.

4.2.3 Event Argument Extraction

Finally, we consider event argument extraction (EAE), an information extraction task that aims to extract arguments for the event mentions from the texts (Ahn, 2006). For a pair of event trigger and entity mention, this task aims to link them with an argument role indicating that the entity

[2]We only perform experiments on English, which is a highly projective language. Extensions to non-projective languages are left to future work.

can play such a role in the event frame. If no such role is possible, then no links are added. We again adopt a pre-trained BERT encoder for encoding and further stack a task-specific predictor, which is a biaffine scorer, similar to dependency parsing. The main difference is that here we perform local normalization for each event-entity pair since there are no constraints on how many other mentions that one mention can be linked to for event argument extraction. To better explore real application scenarios, we train an extra sequence labeler to extract event and entity mentions rather than using gold ones. This mention-detection model is the same as the one utilized in our NER experiments.

More specifically, our model takes a pair of event trigger and entity mention (m_t and m_e) and assigns the probabilities of argument roles to them:

$$p(r|m_t, m_e) = \frac{\exp \text{score}(r|m_t, m_e)}{\sum_{r' \in \mathcal{R} \cup \{\epsilon\}} \exp \text{score}(r'|m_t, m_e)}$$

Here, $\mathcal{R}$ denotes the role labeling space and we further include an option of ϵ to denote there are no argument relations between the event trigger and entity mention. The score function is realized with a biaffine module that produces argument scores for the input mention pair. Since a mention may contain multiple words, we concatenate the word representations of the starting and ending words to form the mention's input vector.

In event extraction, there are constraints on the mention (event and entity) types and argument role labels. For example, the PERSON role of a MARRY event should have the entity type of PER, while the DESTINATION or ORIGIN roles of a TRANSPORT should have entity types denoting places (GPE, LOC or FAC). We adopt a simple method to incorporate such constraints in decoding by ignoring (masking out) the roles that are not possible according to the event and entity types. The role constraints are manually collected according to the event annotation guideline LDC (2005). If not considering these role constraints, we simply adopt greedy prediction for each event-entity pair.

4.3 Experiments

4.3.1 Settings

Data. Our experiments are conducted on widely utilized English datasets. In our main experiments, we adopt the CoNLL-2003 English dataset[3] (Tjong Kim Sang and De Meulder, 2003) for NER and the English Web Treebank (EWT) from Universal Dependencies[4] v2.10 (Nivre et al., 2020) for DPAR. In the genre transfer experiments for NER and DPAR, we utilize OntoNotes 5.0 dataset[5] (Weischedel et al., 2013) and split the data according to text genres. For the event task, we adopt the ACE05 dataset[6] (Walker et al., 2006), using the scripts from Lin et al. (2020) for the pre-processing.[7] Table 4.1 shows the data statistics.

[3] https://www.clips.uantwerpen.be/conll2003/ner/
[4] https://universaldependencies.org/
[5] https://catalog.ldc.upenn.edu/LDC2013T19
[6] https://catalog.ldc.upenn.edu/LDC2006T06
[7] http://blender.cs.illinois.edu/software/oneie/

Data	Split	#Sent.	#Token	#Event	#Entity	#Argument	#Relation
CoNLL03	train	14.0K	203.6K	-	23.5K	-	-
	dev	3.3K	51.4K	-	5.9K	-	-
	test	3.5K	46.4K	-	5.6K	-	-
UD-EWT	train	12.5K	204.6K	-	-	-	-
	dev	2.0K	25.1K	-	-	-	-
	test	2.1K	25.1K	-	-	-	-
ACE05	train	14.4K	215.2K	3.7K	38.0K	5.7K	6.2K
	dev	2.5K	34.5K	0.5K	6.0K	0.7K	0.8K
	test	4.0K	61.5K	1.1K	10.8K	1.7K	1.7K

Table 4.1: Data statistics of the datasets utilized in our main experiments.

Task	Model	5K	20K	100K
NER	Local	$84.27_{0.8}$	$88.91_{0.6}$	$91.24_{0.3}$
	Global	$84.73_{0.1}$	$88.92_{0.4}$	$91.38_{0.2}$
DPAR	Local	$84.57_{0.1}$	$89.46_{0.2}$	$91.95_{0.1}$
	Global	$82.65_{0.3}$	$88.92_{0.3}$	$91.65_{0.2}$

Table 4.2: Comparisons between local and global models for NER (F1%) and DPAR (UAS%).

Model and training. Unless otherwise specified, we adopt the pre-trained BERT$_{\text{base}}$ as the contextualized encoder for our models. The encoder is fined-tuned with the task-specific decoders in all the experiments. The number of model parameters is around 110M. We follow common practices for the settings of other hyper-parameters. Adam (Kingma and Ba, 2014) is utilized as the optimizer. The learning rate is initially set to 1e-5 for NER and 2e-5 for DPAR and EAE. It is further linearly decayed to 10% of the initial value throughout the training process. The models are trained for 20K steps with a batch size of around 512 tokens. We pick final models by the performance on the development set of each task. The original development sets are also down-sampled accordingly to the training sets to simulate scenarios with different data amounts. All the reported results are averaged over five runs with different random seeds.

Local normalization. In our main experiments, we choose locally normalized models instead of more complex global models. Table 4.2 provides comparisons between the local and global models for NER and DPAR. For the global models, we use standard linear-chain CRF (Lafferty et al., 2001) for NER and tree-CRF (Paskin, 2001) for DPAR. For these results, constrained decoding is applied since it is found to be helpful for both local and global models. The results show that there are no clear benefits of using global models over the simpler local models, probably due to the strong input context modeling capabilities of the underlying pre-trained encoders. Therefore, we simply adopt local models in our main experiments.

Evaluation. We adopt standard evaluation metrics for the tasks: labeled F1 score for NER, unlabeled attachment score (UAS) for DPAR, labeled argument F1 score for EAE (Lin et al., 2020).

4.3.2 RQ1: On Training Data

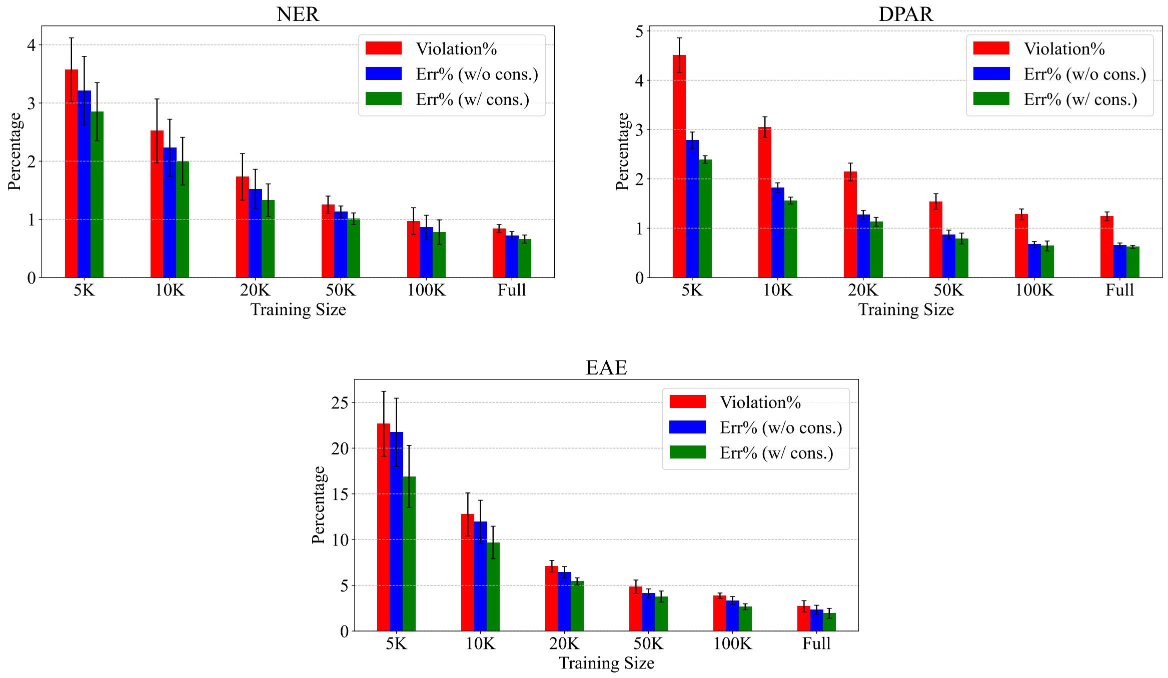


Figure 4.2: Illustrations of constraint violations and related error rates.

We first investigate the effectiveness of incorporating constraints in decoding, plotting the rates of structural violations and related errors in Figure 4.2. For all the predicted items (all non-'O' tags for NER, all dependency edges for DPAR, and all predicted argument links for EAE), we calculate the percentage of items that violate the structural constraints when using greedy decoding ("Violation%"). For NER, we perform analysis at the tag level and count the illegal tag transitions. For DPAR, we include the edges that are inside a loop (violating the acyclic constraint) or go across another edge (violating the projective constraint). For EAE, we count the argument links whose role does not comply with the types of the event and the entity that it connects. We further calculate "Err%", which denotes the percentage of the items that contain violations in greedy decoding and are wrongly predicted at the same time. Such error rates are calculated for both greedy (w/o cons.) and constrained (w/ cons.) modes, and the comparisons between these two can illustrate the amount of error reduction that constrained decoding can bring.

The overall trends are consistent on all the tasks. As we have more training data, there are fewer structural violations without explicitly enforcing constraints, which indicates that the model can implicitly learn the constraints if given enough training data. Moreover, although

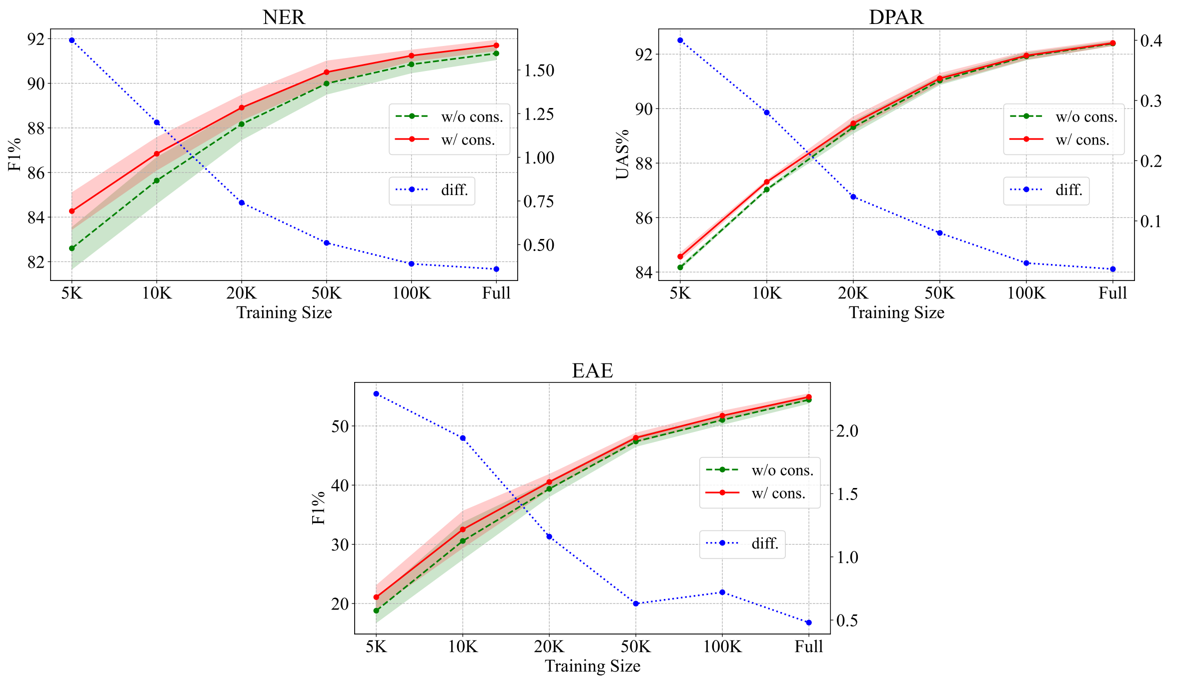


Figure 4.3: Test results with or without applying constraints against different training sizes.

constrained decoding can eliminate such violations, they do not always lead to the correct predictions; only a small portion of incorrect items can be corrected with constrained decoding, and such improvements are more prominent with less training data.

We further show the main test results in Figure 4.3. The general trends are again similar for all three tasks: constraints provide consistent benefits for the model performance, and such benefits are larger as we have less training data. This corresponds well to the violation analysis in Figure 4.2: with enough training data, the model will implicitly learn the structural constraints from the data and further enhancement of constrained decoding will make little difference; however, with less training data, explicitly enforcing constraints can be more beneficial.

Takeaways: Without incorporating constraints, there are more constraint violations from the predictions of the models trained with fewer data. By enforcing constraints in decoding, there can be consistent benefits for model performance and such improvements are greater with models learned with less training data.

4.3.3 RQ2: On Efficient Models

We further explore the influence of using more efficient models. We take the distilled versions of the BERT models from Turc et al. (2019) and repeat our previous experiments. Specifically, we consider five models (L=Layer Number, H=Dimension Size): Tiny (L=2, H=128), Mini (L=4, H=256), Small (L=4, H=512), Medium (L=8, H=512), and Base (L=12, H=768). We plot "Violation%" and performance differences in Figure 4.4 and Figure 4.5, respectively.

First, if looking at the axis of the training data size, the overall trends are similar to what

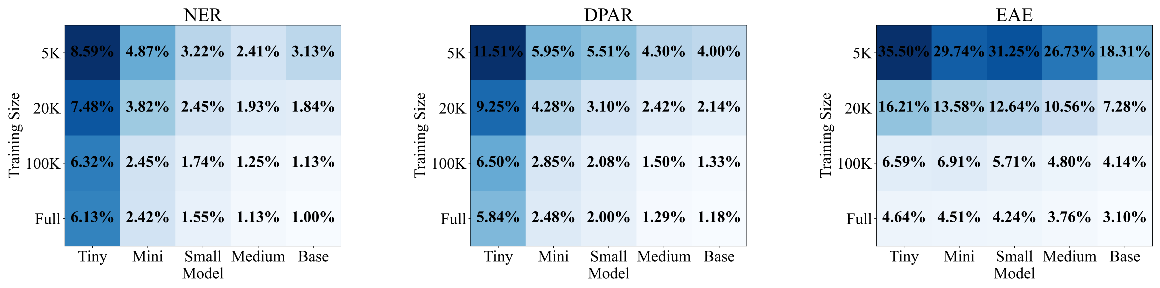


Figure 4.4: "Violation%" (percentages of predicted items that violates constraints with greedy decoding) with different models and amounts of training data.

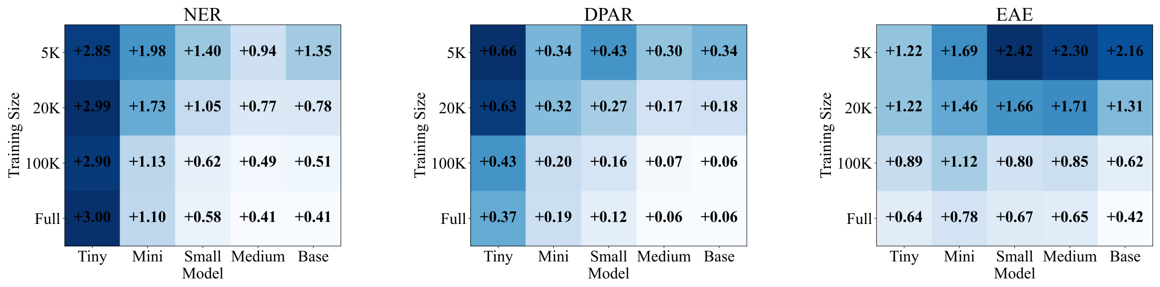


Figure 4.5: Performance improvements brought by constrained decoding with different models and amounts of training data.

we found previously: there are more violations with less training data, and enforcing constraints helps more in the lower-resource scenarios. This trend is generally consistent across all the underlying models. Moreover, comparing across the model axis brings more interesting findings. Overall, the smaller models tend to output predictions with more violations if adopting greedy decoding and incorporating constraints generally bring more performance improvements for smaller models. The reason for this trend might be that smaller models contain fewer parameters to learn all the patterns in the training data and such under-parameterization may bring difficulties in implicitly capturing the constraints.

Another interesting question is how decoding speed is influenced by the underlying model and the decoding algorithm. Table 4.3 presents the time required to decode one sentence for NER and DPAR. Here, we do not analyze the EAE task, since there are no complex algorithms involved for our constrained decoding for EAE and we did not find obvious speed differences between decoding methods with or without constraints. Generally, constrained decoding requires more computational cost compared with the constraint-agnostic greedy methods. This is not surprising since the constraint-agnostic decoding method simply predicts the locally maximally scored items while constrained decoding needs to invoke algorithms with higher complexity. With smaller models, constrained decoding brings relatively more cost because there are less intense computational requirements for the underlying encoder. This trend is especially obvious for the NER task, where constrained decoding costs nearly twice the time as greedy decoding when using the Tiny model. When adopting larger models, the encoder starts to require more

	Tiny	Mini	Small	Medium	Base
NER$_{w/o}$	0.29	0.32	0.36	0.53	1.19
NER$_{w/}$	0.56	0.59	0.64	0.80	1.45
DPAR$_{w/o}$	0.23	0.26	0.33	0.47	1.07
DPAR$_{w/}$	0.28	0.31	0.36	0.50	1.10

Table 4.3: Decoding speed (ms per sentence) without ($_{w/o}$) or with ($_{w/}$) constraints.

computations, and thus the relative extra cost brought by constrained decoding takes a smaller proportion.

Takeaways: When using smaller and more efficient models such as distilled versions of BERT, they tend to output predictions with more structural violations with greedy decoding and constrained decoding generally brings more benefits.

4.3.4 RQ3: On Genre Transfer

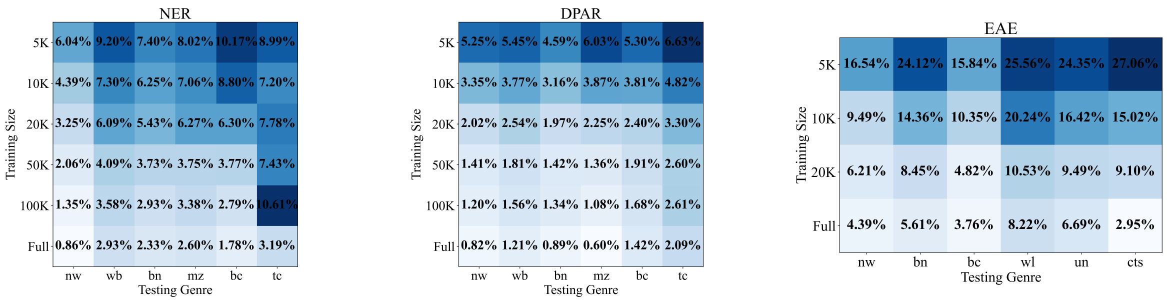


Figure 4.6: "Violation%" (percentages of predicted items that violates structural constraints) on different testing genres with different amounts of source newswire training data.

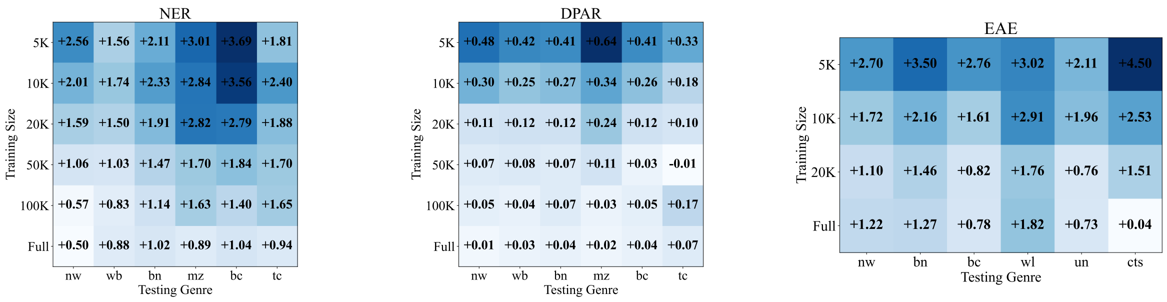


Figure 4.7: Performance improvements brought by constrained decoding on different testing genres with different amounts of source training data.

Finally, we explore a transfer-learning scenario where there are discrepancies between the training and testing data distributions. Specifically, we consider transferring across different text genres. For these experiments, we utilize OntoNotes for NER and DPAR, and ACE05 for EAE. We take the newswire (nw) portion as the source for training and directly test the source-trained model on the test sets of other genres (in a zero-shot transferring manner).

The results are shown in Figure 4.6 and 4.7, where the notations are similar to those in §4.3.3. In these results, similar patterns along the data size axis can be found: incorporating constraints is more helpful in the cases with less training data and such trends generally hold for out-of-distribution testing scenarios (target genres that are not "nw") as well.

One more interesting aspect to inspect is the patterns along the axis of genres. In the figures, we sort the testing genres according to their similarities to the source (nw). To calculate the similarities between genres, we simply utilize the overlapping rate of vocabularies since lexical overlaps can be one important factor for the effectiveness of the transfer. Overall, there is a weak trend that when transferring to more distant genres, greedy decoding tends to produce outputs with more structural violations. However, this pattern is not consistent across all the cases and one potential reason might be the instability of the model transfer. Moreover, there can be more appropriate measurements than our simple lexicon-based similarity that may better reflect how the predictions are influenced by constrained decoding across genres. We leave more explorations to future work.

Takeaways: The previous patterns still generally hold for testing on out-of-domain instances with genre discrepancies: models trained with fewer training data tend to make more violations with greedy decoding and benefit more from constrained decoding. There is also a weak pattern that when transferring to more distant genres, greedy decoding tends to produce more violations.

4.4 Related Work

For structured prediction tasks, one important property is that the prediction outputs are complex objects with multiple interdependent variables. How to model such inter-dependencies is an important question for traditional NLP research. Classical algorithms for decoding and learning have been developed for various structured prediction tasks, including Viterbi algorithm (Viterbi, 1967) and forward-backward algorithm (Baum et al., 1970) for sequence labeling, maximum spanning tree algorithm (Chu and Liu, 1965; Edmonds, 1967), Inside-Outside algorithm (Paskin, 2001) and Matrix-Tree Theorem (Koo et al., 2007; Smith and Smith, 2007; McDonald and Satta, 2007) for dependency parsing, as well as more complex algorithms for tasks involving more complicated graph structures (Rush and Collins, 2012; Burkett and Klein, 2013; Martins et al., 2015; Gormley and Eisner, 2015). Though recent developments of neural models and pre-trained language models have boosted the performance of simple local models, better modeling of the structured outputs have still been shown effective for various structured prediction tasks (Wang et al., 2019c; Fonseca and Martins, 2020; Zhang et al., 2020b; Wei et al., 2021).

For the output modeling of structured prediction tasks, the hard structural constraint is a key factor for the development of decoding and learning algorithms. To enhance general explicitly stated constraints, Roth and Yih (2004) tackle the decoding problem with Integer Linear Programming (ILP) and such paradigm has been applied to a range of structured NLP tasks (Denis and Baldridge, 2007; Roth and Yih, 2007; Clarke and Lapata, 2008; Punyakanok et al., 2008). In addition to enforcing well-formed output structures for decoding, constraints can be also incorporated to enhance model learning (Chang et al., 2008; Li et al., 2020d; Pan et al., 2020; Wang et al., 2020a, 2021a). While we mainly focus on simply applying constrained decoding with local models trained with different amounts of data, it would be interesting to explore the influences when further incorporating constraints at model training time.

4.5 Conclusion

In this chapter, we explore the interactions of constraint-based decoding algorithms and the amounts of training data for typical structured prediction tasks in NLP. Specifically, we train local models with different amounts of training data and analyze the influence of whether to adopt constrained decoding or not. The results show that when the model is trained with less data, the predictions contain more structural violations with greedy decoding and there are more benefits to model performance by further applying constrained decoding. Such patterns also generally hold with more efficient models and when transferring across text genres, where there are further interesting patterns with regard to model sizes and genre distances.

Part II

Transfer Learning

Chapter 5

On Difficulties of Cross-Lingual Transfer with Order Differences: A Case Study on Dependency Parsing

Starting from this chapter, we examine transfer learning, that is, utilizing data resources from related datasets to help the target task. In this chapter, we specifically investigate cross-lingual transfer and study the influences of word order differences, taking dependency parsing as the studying case.

5.1 Introduction

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

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

For cross-language transfer, one of the key challenges is the variation in word order among different languages. For example, the Verb-Object pattern in English can hardly be found in Japanese. This challenge should be taken into consideration in model design. RNN is a prevalent family of models for many NLP tasks and has demonstrated compelling performances (Mikolov et al., 2010; Sutskever et al., 2014; Peters et al., 2018). However, its sequential nature makes it heavily reliant on word order information, which exposes it to the risk of encoding language-specific order information that cannot generalize across languages. We characterize this as the "*order-sensitive*" property. Another family of models known as "Transformer" uses self-attention mechanisms to capture context and was shown to be effective in various NLP tasks (Vaswani et al., 2017; Liu et al., 2018c; Kitaev and Klein, 2018). With modification in position representations, the self-attention mechanism can be more robust than RNNs to the change of word order. We refer to this as the "*order-free*" property. We posit that *order-free* models have better transferability than *order-sensitive* models because they suffer less from overfitting language-specific word order features. To test our hypothesis, we first quantify language distance in terms of word order typology, and then systematically study the transferability of order-sensitive and order-free neural architectures on cross-lingual dependency parsing.

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

We conduct evaluations on 31 languages across a broad spectrum of language families, as shown in Table 5.1. Our empirical results show that *order-free* encoding and decoding models generally perform better than the *order-sensitive* ones for cross-lingual transfer, especially when the source and target languages are distant.

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

Table 5.1: The selected languages grouped by language families. "IE" is the abbreviation of Indo-European.

5.2 Quantifying Language Distance

We first verify that we can measure "language distance" based on word order since it is a significant distinctive feature to differentiate languages (Dryer, 2007). The World Atlas of Language Structures (WALS) (Dryer and Haspelmath, 2013) provides a great reference for word order typology and can be used to construct feature vectors for languages (Littell et al., 2017). But since we already have the universal dependency annotations, we take an empirical way and directly extract word order features using directed dependency relations (Liu, 2010). We conduct our study using the Universal Dependencies (UD) Treebanks (v2.2) (Nivre et al., 2018b). As shown above in Table 5.1, we select 31 languages for evaluation and analysis, with the selection criterion being that the total token number in the treebanks of that language is over 100K.

We look at finer-grained dependency types than the 37 universal dependency labels[1] in UD v2 by augmenting the dependency labels with the universal part-of-speech (POS) tags of the head and modifier nodes. Specifically, we use triples "(ModifierPOS, HeadPOS, DependencyLabel)" as the augmented dependency types. With this, we can investigate language differences in a fine-grained way by defining directions on these triples (i.e. modifier before head or modifier after head). We conduct feature selection by filtering out rare types as they can be unstable. Our filtering criterion is that the type's frequency is larger than 0.1% and it appears in at least 20 languages. For each dependency type, we collect the statistics of directionality (Liu, 2010; Wang and Eisner, 2017). Since there can be only two directions for an edge, for each dependency type, we use the relative frequency of the left direction (modifier before head) as the directional feature. By concatenating the directional features of all selected triples, we obtain a word-ordering feature vector for each language. We calculate the *word-ordering distance* using these vectors. In this chapter, we simply use the Manhattan distance, which we empirically find works well.

We perform hierarchical clustering based on the word-ordering vectors for the selected languages, following Östling (2015). As shown in Figure 5.1, the grouping of the ground truth language families is almost recovered. The two outliers, German (de) and Dutch (nl), are indeed different from English. For instance, German and Dutch adopt a larger portion of Object-Verb order in embedded clauses. The above analysis shows that word order is an important feature to characterize differences between languages. Therefore, it should be taken into consideration in the model design.

5.3 Models

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

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

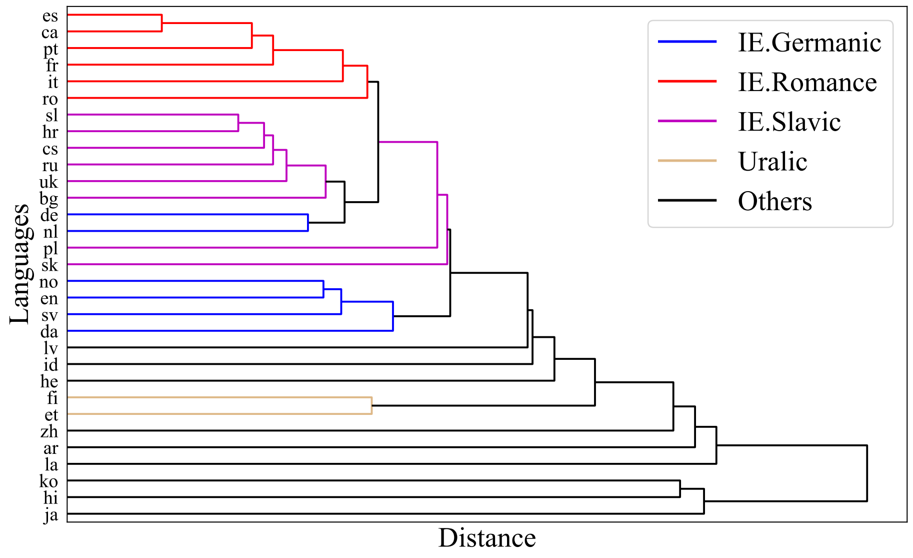


Figure 5.1: Hierarchical clustering (with the Nearest Point Algorithm) dendrogram of the languages by their word-ordering vectors.

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

5.3.1 Contextual Encoders

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

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

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

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

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

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

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

Therefore, the learnable relative postion representations have $k+1$ types rather than $2k+1$: we have $w^K = (w_0^K, \ldots, w_k^K)$, and $w^V = (w_0^V, \ldots, w_k^V)$.

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

5.3.2 Structured Decoders

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

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

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

5.4 Experiments and Analysis

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

5.4.1 Setup

	Layer	Hyper-Parameter	Value
Input	Word	dimension	300
	POS	dimension	50
RNN	Encoder	encoder layer	3
		encoder size	300
	MLP	arc MLP size	512
		label MLP size	128
	Training	Dropout	0.33
		optimizer	Adam
		learning rate	0.001
		batch size	32
Self-Attention	Encoder	encoder layer	6
		d_{model}	350
		d_{ff}	512
	MLP	arc MLP size	512
		label MLP size	128
	Training	Dropout	0.2
		optimizer	Adam
		learning rate	0.0001
		batch size	80

Table 5.2: Hyper-parameters in our experiments.

Settings In our main experiments (those except Section 5.4.3), we take English as the source language and 30 other languages as target languages. We only use the source language for both training and hyperparameter tuning. During testing, we directly apply the trained model to target languages with the inputs from target languages passed through pre-trained multilingual embeddings that are projected into a common space as the source language. The projection is done by the offline transformation method (Smith et al., 2017) with pre-trained 300d monolingual embeddings from FastText (Bojanowski et al., 2017). We freeze word embeddings since fine-tuning them may disturb the multi-lingual alignments. We also adopt gold UPOS tags for the inputs. Table 5.2 shows our main hyper-parameters, which are similar to the ones in the Biaffine Graph Parser (Dozat and Manning, 2017) and the Stack-Pointer Parser (Ma et al., 2018). Throughout our experiments, we adopted the language-independent UD labels and a sentence length threshold of 140. The evaluation metrics are unlabeled attachment score (UAS) and labeled attachment score (LAS) with punctuations excluded. We trained our cross-lingual models five times with different initializations and reported average scores.

Systems As described before, we have an *order-free* (Self-Attention) and an *order-sensitive* (BiLSTM-RNN) encoder, as well as an *order-free* (Biaffine Attention Graph-based) and an *order-sensitive* (Stack-Pointer) decoder. The combination gives us four different models, named in the format of "Encoder" plus "Decoder". For clarity, we also mark models with their encoder-decoder order sensitivity characteristics. For example, "SelfAtt-Graph (OF-OF)" refers to the model with a self-attention order-free encoder and a graph-based order-free decoder. We benchmark our models with a baseline shift-reduce transition-based parser, which gave previous state-of-the-art results for single-source zero-resource cross-lingual parsing (Guo et al., 2015). Since they used older datasets, we also re-trained the model on our datasets with their implementation and compared it with their results. We also list the supervised learning results using the "RNN-Graph" model on each language as a reference of the upper bound for cross-lingual parsing.

5.4.2 Results

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

Table 5.3: Main results (UAS%/LAS%) on the test sets.

The main results on the test sets are shown in Table 5.3. The languages are ordered by their order typology distance to English, as shown in the second column. '*' refers to the results of delexicalized models, '†' means that the best transfer model is statistically significantly better (by paired bootstrap test, $p < 0.05$) than all other transfer models. Models are marked with their encoder and decoder order sensitivity, OF denotes order-free and OS denotes order-sensitive. In preliminary experiments, we found our lexicalized models performed poorly on Chinese (zh) and Japanese (ja). We found the main reason was that their embeddings were not well aligned with English. Therefore, we use delexicalized models, where only POS tags are used as inputs. The delexicalized results[2] for Chinese and Japanese are listed in the rows marked with "*".

Overall, the "SelfAtt-Graph" model performs the best in over half of the languages and beats the runner-up "RNN-Graph" by around 1.3 in UAS and 1.2 in LAS on average. When compared with "RNN-Stack" and "SelfAtt-Stack", the average difference is larger than 1.5 points. This shows that models that capture less word order information generally perform better at cross-lingual parsing. Compared with the baseline, our superior results show the importance of the contextual encoder. Compared with the supervised models, the cross-lingual results are still lower by a large gap, indicating space for improvements. We also ran our models on Google Universal Dependency Treebanks v2.0 McDonald et al. (2013), which is an older dataset that was used by Guo et al. (2015). As shown in Table 5.4, our models perform better consistently.

Language	SelfAtt-Graph	RNN-Graph	SelfAtt-Stack	RNN-Stack	(Guo et al., 2015)
German	65.03/55.03	64.60/54.57	63.63/54.40	**65.51/55.82**	60.35/51.54
French	74.45/63.28	**76.75/65.20**	73.63/62.76	75.13/64.44	72.93/63.12
Spanish	72.00/61.50	73.99/63.46	71.73/61.42	**74.13/64.00**	71.90/62.28

Table 5.4: Comparisons (UAS%/LAS%) on Google Universal Dependency Treebanks v2.0.

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

5.4.3 Analysis

We further analyze how different modeling choices influence cross-lingual transfer. Since we have not touched the training sets for languages other than English, in this subsection, we evaluate and analyze the performance of target languages using training splits in UD. The performance of English is evaluated on the test set. We verify that the trends observed in the test set are similar to those in the training sets. As mentioned in the previous section, the bilingual embeddings for Chinese and Japanese do not align well with English. Therefore, we report the results with delexicalizing.

Encoder Architecture

We assume models that are less sensitive to word order perform better when transfer to distant languages. To empirically verify this point, we conduct controlled comparisons on various en-

[2]We found delexicalized models to be better only at zh and ja, for about 5 and 10 points respectively. For other languages, they performed worse at about 2 to 5 points. We also tried models without POS and found them worse by about 10 points on average.

coders with the same graph-based decoder. Table 5.5 shows the average performances in all languages.

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

Table 5.5: Comparisons of different encoders (averaged results over all languages on the original training sets).

To compare models with various degrees of sensitivity to word order, we include several variations of self-attention models. The "SelfAtt-NoPosi" is the self-attention model without any positional information. Although it is most insensitive to word order, it performs poorly possibly because of the lack of access to the locality of contexts. The self-attention model with absolute positional embeddings ("SelfAtt-Absolute") also does not perform well. In the case of parsing, relative positional representations may be more useful as indicated by the improvements brought by the directional relative position representations ("SelfAtt-Relative+Dir") (Shaw et al., 2018). Interestingly, the RNN encoder ranks between "SelfAtt-Relative+Dir" and "SelfAtt-Absolute"; all these three encoders explicitly capture word order information in some way. Finally, by discarding the information of directions, our relative position representation ("SelfAtt-Relative") performs the best (significantly better at $p < 0.05$).

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

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

Performance v.s. Language Distance

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

For each target language, we collect two types of distances when comparing it to English: one is the *word-ordering distance* as described in Section 5.2, the other is the *performance distance*, which is the gap of evaluation scores[3] between the target language and English. The performance

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

distance can represent the general transferability from English to this language. We calculate the correlation of these two distances on all the concerned languages, and the results turn out to be quite high: the Pearson and Spearman correlations are *around 0.90 and 0.87* respectively, using the evaluations of any of our four cross-lingual transfer models. This suggests that word order can be an important factor for cross-lingual transferability.

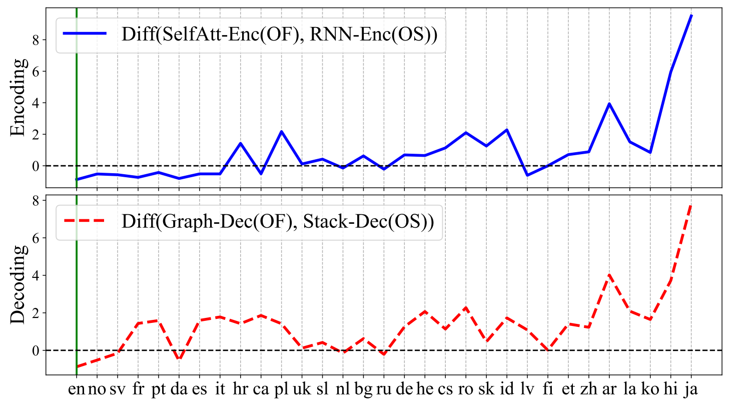
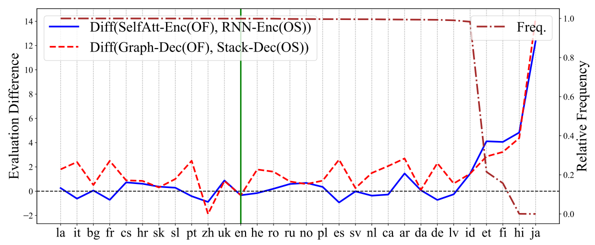


Figure 5.2: Evaluation score differences between Order-Free (OF) and Order Sensitive (OS) modules.

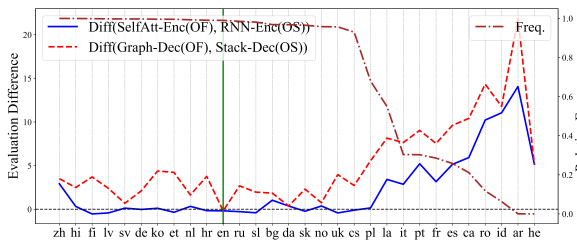
Furthermore, we individually analyze the encoders and decoders of the dependency parsers. Since we have two architectures for each of the modules, when examining one, we take the highest scores obtained by any of the other modules. For example, when comparing RNN and Self-Attention encoders, we take the best evaluation scores of “RNN-Graph” and “RNN-Stack” for RNN and the best of “SelfAtt-Graph” and “SelfAtt-Stack” for Self-Attention. Figure 5.2 shows the score differences of encoding and decoding architectures against the languages’ distances to English. We show the results of both encoder (blue solid curve) and decoder (dashed red curve). Languages are sorted by their word-ordering distances to English from left to right. The position of English is marked with a green bar. For both the encoding and decoding modules, we observe a similar overall pattern: the order-free models, in general, perform better than order-sensitive ones in the languages that are distant from the source language English. On the other hand, for some languages that are closer to English, order-sensitive models perform better, possibly benefiting from being able to capture similar word-ordering information. The performance gaps between order-free and order-sensitive models are positively correlated with language distance.

Performance Breakdown by Types

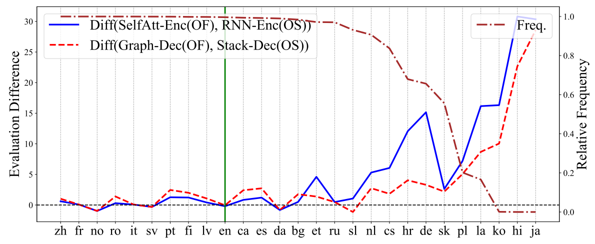
Moreover, we compare the results on specific dependency types using concrete examples. For each type, we sort the languages by their relative frequencies of left direction (modifier before head) and plot the performance differences for encoders and decoders. We highlight the source language English in green. Figure 5.3 shows four typical example types: Adposition and Noun, Adjective and Noun, Auxiliary and Verb, and Object and Verb. To save space, we merge the curves of encoders and decoders into one figure. The blue and red curves and left y-axis represent the differences in evaluation scores, while the brown curve and right y-axis represent the



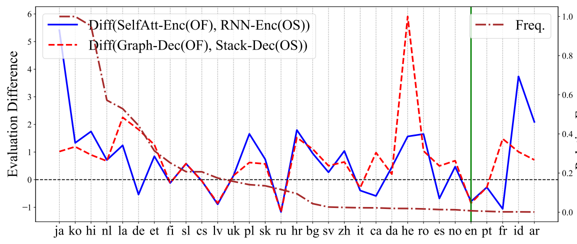
(a) Adposition & Noun (ADP, NOUN, case)



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



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



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

Figure 5.3: Analysis on specific dependency types.

relative frequency of the left direction (modifier before head) on this type. The languages (x-axis) are sorted by this relative frequency from high to low. In Figure 5.3a, we examine the "case" dependency type between adpositions and nouns. The pattern is similar to the overall pattern. For languages that mainly use prepositions as in English, different models perform similarly, while for languages that use postpositions, order-free models get better results. The patterns of adjective modifier (Figure 5.3b) and auxiliary (Figure 5.3c) are also similar.

On dependencies between verbs and object nouns, although in general order-free models perform better, the pattern diverges from what we expect. There can be several possible explanations for this. Firstly, the tokens which are noun objects of verbs only take about 3.1% on average over all tokens. Considering just this specific dependency type, the correlation between frequency distances and performance differences is 0.64, which is far less than 0.9 when considering all types. Therefore, although Verb-Object ordering is a typical example, we cannot take it as the whole story of word order. Secondly, Verb-Object dependencies can often be difficult to decide. They sometimes are long-ranged and have complex interactions with other words. Therefore, merely reducing modeling order information can have complicated effects. Moreover, although our relative-position self-attention encoder does not explicitly encode word positions, it may still capture some positional information with relative distances. For example, the words in the middle of a sentence will have different distance patterns from those at the beginning or the end. With this knowledge, the model can still prefer the pattern where a verb is in the middle as in English's Subject-Verb-Object ordering and may find sentences in Subject-Object-Verb languages strange. It will be interesting to explore more ways to weaken or remove this bias.

Analysis on Dependency Distances

We now look into dependency lengths and directions. Here, we combine dependency length and direction into dependency distance d, by using negative signs for dependencies with left-direction (modifier before head) and positive for right-direction (head before modifier). We find a seemingly strange pattern at dependency distances $|d|$=1: for all transfer models, evaluation scores on d=-1 can reach about 80, but on d=1, the scores are only around 40. This may be explained by the relative frequencies of dependency distances as shown in Table 5.6, where there is a discrepancy between English and the average of other languages at d=1. About 80% of the dependencies with $|d|$=1 in English is the left direction (modifier before head), while overall other languages have more right directions at $|d|$=1. This suggests an interesting future direction of training on more source languages with different dependency distance distributions.

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

Table 5.6: Relative frequencies (%) of dependency distances.

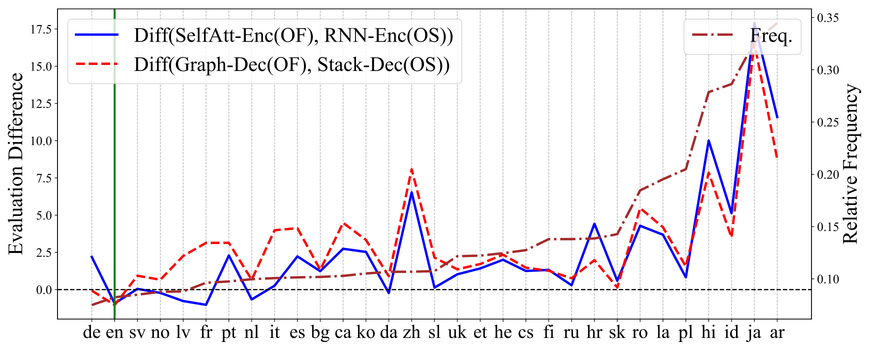


Figure 5.4: Evaluation differences of models on d=1 dependencies.

We further compare the four models on the d=1 dependencies and as shown in Figure 5.4. Here, notations are the same as in Figure 5.3, languages are sorted by percentages (represented by the brown curve and right y-axis) of d=1 dependencies. The order-free models perform better at the languages which have more d=1 dependencies. Such a finding indicates that our model design of reducing the ability to capture word order information can help with short-ranged dependencies of different directions to the source language. However, the improvements are still limited. One of the most challenging parts of unsupervised cross-lingual parsing is modeling cross-lingually shareable and language-unspecific information. In other words, we want flexible yet powerful models. Our exploration of the order-free self-attentive models is the first step.

Transfer between All Language Pairs

Finally, we investigate the transfer performance of all source-target language pairs.[4] We first generate a performance matrix A, where each entry (i, j) records the transfer performance from a

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

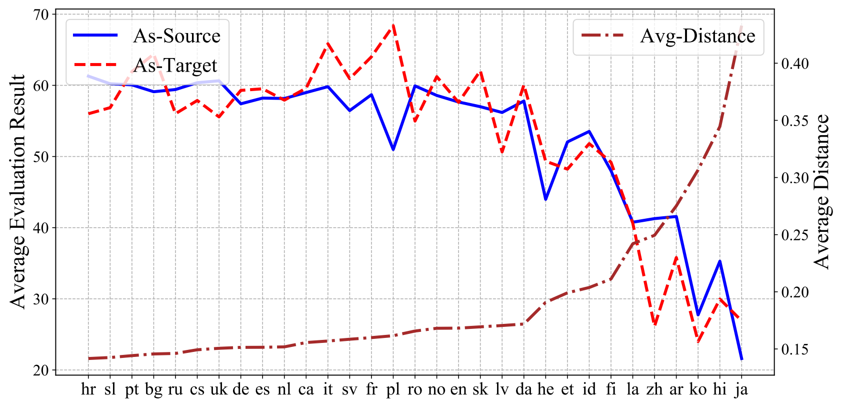


Figure 5.5: Transfer performance of all source-target language pairs.

source language i to a target language j. We then report the following two aggregate performance measures on A in Figure 5.5: 1) *As-source* reports the average over columns of A for each row of the source language and 2) *As-target* reports the average over rows of A for each column of the target language. As a reference, we also plot the average word-order distance between one language to other languages. Results show that both *As-source* (blue line) and *As-target* (red line) highly are anti-correlated (Pearson correlation coefficients are -0.90 and -0.87, respectively) with average language distance (brown line).

5.5 Related Work

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

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

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

5.6 Conclusion

In this chapter, we conduct a comprehensive study on how the design of neural architectures affects cross-lingual transfer learning. We examine two notable families of neural architectures (sequential RNN v.s. self-attention) using dependency parsing as the evaluation task. We show that *order-free* models perform better than *order-sensitive* ones when there is a significant difference in the word order typology between the target and source languages.

Chapter 6

On the Benefit of Syntactic Supervision for Cross-lingual Transfer in Semantic Role Labeling

In this chapter, we continue our study on cross-lingual transfer and further consider learning with cross-task training signals. Specifically, we investigate the connections between syntax and SRL, showing that syntactic supervision can help facilitate cross-lingual transfer in SRL when lacking abundant direct target training data.

6.1 Introduction

The task of semantic role labeling (SRL) annotates predicate-argument structures in text and is thus a desirable output of natural language processing (NLP) pipelines designed to extract information from text (Gildea and Jurafsky, 2002; Palmer et al., 2010). Recent developments in neural architectures (Vaswani et al., 2017) and pre-trained contextualized representations (Devlin et al., 2019; Liu et al., 2019) have greatly improved the performance of SRL systems (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Shi and Lin, 2019). However, most previous work focuses on high-resource English SRL scenarios, and it remains a challenge to extend these approaches, which require plentiful supervised examples, to other languages where training resources may be limited.

A popular approach addressing this challenge is cross-lingual learning: leveraging the shared structures across human languages to transfer knowledge from high-resource languages to low-resource ones. Model transfer, where an SRL model is directly transferred across languages using shared representations (Kozhevnikov and Titov, 2013, 2014; Fei et al., 2020b), is a particularly promising approach thanks to recent developments in multilingual contextualized representations (Lample and Conneau, 2019; Conneau et al., 2020), which have proven effective for cross-lingual transfer (Wu and Dredze, 2019; Pires et al., 2019).

Another common strategy for improving SRL model performance in both high- and low-resource scenarios is incorporating syntactic information. Syntactic analysis was until recently considered a prerequisite for most SRL systems (Gildea and Palmer, 2002; Punyakanok et al.,

2008) and has been shown to benefit recent neural models as well (Marcheggiani and Titov, 2017; He et al., 2018b; Swayamdipta et al., 2018; Strubell et al., 2018). Despite much work exploring cross-lingual learning and incorporating syntactic information into SRL systems, most such previous work explores these two avenues separately, though there are numerous reasons that carefully incorporating syntax into a cross-lingual system for SRL could provide further benefits: First, whereas SRL annotations are limited to only about a dozen languages, much richer resources are available for syntax, thanks to the development of the Universal Dependencies (UD) framework and accompanying corpora (Nivre et al., 2016b, 2020), which defines syntactic annotations that are consistent across languages, with treebanks in over 100 languages to date. Second, UD treebanks in particular have the potential to increase beneficial sharing of information across languages by providing a unified syntactic structure to ground cross-lingual representations.

Most previous work utilizing syntax for cross-lingual SRL has incorporated syntactic information only as an input to the model, either as sparse features (Kozhevnikov and Titov, 2013; Pražák and Konopík, 2017) or as structures for tree encoders (Fei et al., 2020b). These strategies require syntactic pre-processing by an additional model and can suffer from error propagation. In this chapter, we explore an alternative approach that has yet to be explored in the cross-lingual setting: adopting syntactic annotations as auxiliary supervision and performing multitask learning (Caruana, 1997) together with SRL (Swayamdipta et al., 2018; Strubell et al., 2018; Cai and Lapata, 2019).

To evaluate the extent to which syntactic supervision can help facilitate cross-lingual transfer in SRL, we perform a comprehensive empirical analysis on three SRL benchmark datasets, covering ten languages (in addition to English). We evaluate our models in both zero-shot and semi-supervised scenarios and on both dependency- and span-based SRL. Highlights of our findings include: 1) Training SRL models with syntactic supervision is consistently helpful in low-resource SRL scenarios. (§6.3.2, §6.3.3, §6.3.4, §6.3.5); 2) When lacking direct syntactic annotations for the target language, available treebanks from related languages can be used instead to improve SRL performance (§6.3.4); 3) For span-based SRL, a syntax-aware SRL decoder out-performs BIO-tagging when combined with syntactic training. (§6.3.5)

6.2 Model

We adopt the typical encoder-decoder paradigm for multi-task learning to perform syntactic dependency parsing and SRL together in one model. A shared encoder gives the hidden representations for the input words and each task has its own decoder that takes those shared representations as inputs and predicts task-specific labels. We hypothesize that syntactic training can provide helpful signals for SRL through the shared encoder.

6.2.1 Encoder

We adopt multilingual pre-trained contextualized models as our encoder, following previous work reporting strong performance for SRL (Shi and Lin, 2019; He et al., 2019; Conia and Navigli, 2020) and cross-lingual learning (Wu and Dredze, 2019; Pires et al., 2019). For an input sequence of words $w_1, \ldots, w_n$, the encoder produces their contextualized representations $h_1, \ldots, h_n$. These pre-trained models take sub-word tokens as input, but our SRL and syntactic data have word-level annotations, so we take the first sub-token of a word as its representation. These representations are then provided to task-specific decoders.

6.2.2 Syntax Decoder

For the syntactic (dependency) parsing task, we ignore the single-head constraints in training and view it as a pairwise labeling task into the space of dependency labels $\mathcal{R}_d$:

$$p(r_d|w_H, w_M) = \frac{e^{\text{score}_{r_d}(h_H, h_M)}}{\sum_{r' \in \mathcal{R}_d \cup \{\epsilon\}} e^{\text{score}_{r'}(h_H, h_M)}}$$

where $p(r_d|w_H, w_M)$ denotes the probability that the head w_H has a dependency relation r_d to the modifier w_M (or ϵ, which means no syntactic relation). Following Dozat and Manning (2017), we use biaffine modules for the scoring (score_{r_d}), which take the encoder representations and produce relation scores. For training, we use cross-entropy as the objective. Notice that although this type of pairwise formulation is not widely used for syntactic dependencies, it has been shown effective for semantic dependency parsing (Dozat and Manning, 2018). Our main motivation[1] to utilize it here is to make the syntactic task more similar to SRL. In our syntactic parsing evaluation, we find that this method obtains similar results to the head-selection method.

6.2.3 SRL Decoder

We focus on the end-to-end SRL task, which extracts both the predicates and their arguments (i.e. we do not assume gold predicates unless otherwise noted). For argument extraction, we explore two categories of SRL formalism: dependency-based SRL, which only requires labeling the syntactic head word of an argument, and span-based SRL, which requires labeling full argument spans.

Predicate Identification

Predicate identification is cast as a binary classification task. We use a linear scorer over each word's encoded representations to judge whether it triggers a semantic frame.

Dependency-based SRL

For dependency-based SRL, the problem can be again formalized as a pairwise labeling task, and we treat it in a similar way as in the syntax decoder:

$$p(r_s|w_P, w_A) = \frac{e^{\text{score}_{r_s}(h_P, h_A)}}{\sum_{r' \in \mathcal{R}_s \cup \{\epsilon\}} e^{\text{score}_{r'}(h_P, h_A)}}$$

[1] Another potential benefit is that certain parameters of the output layers may be shareable between syntactic and SRL decoders. Though in preliminary experiments we did not find obvious improvements with a simple method of stacking another task-specific classification layer and sharing the middle biaffine layers, this could be an interesting direction to explore with better parameter-sharing schemes.

Here $p(r_s|w_P, w_A)$ denotes the probability that a predicate w_P takes w_A as an argument with the semantic role r_s (or ϵ, which denotes no semantic relation). Again we use biaffine modules for scoring and cross-entropy as the objective function.

Span-based SRL

Predicting argument spans is usually cast as a sequence labeling problem, with most recent neural SRL models adopting a simple BIO-tagging decoder (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Shi and Lin, 2019). In this chapter, we further consider the two-step syntax-aware approach, as described in Chapter 3 (§3.2). In this method, the first step identifies the argument head and the second step decides span boundaries given the head identified in the first step. Here, the first step is exactly the task of dependency-based SRL and we use the same decoder. For the second step, we adopt the span selection method from extractive question answering (Wang and Jiang, 2016; Devlin et al., 2019) and use two classifiers to decide the start and end of the span given the head word.

6.2.4 Training Scheme

To deal with the multi-task and multilingual scenarios, we adopt a simple training scheme. For each training step, we first sample a task (parsing or SRL), and then a language (source or target). Based on these, we sample a batch of instances from the corresponding dataset and train the model on the selected task. In our experiments, we apply fixed sampling rates for the selection of tasks and languages (1:2 for parsing vs. SRL and 1:1 for source vs. target). In preliminary experiments, we also tried varying sampling rates but did not find obvious improvements.

6.3 Experimental Results

6.3.1 General Settings

Experiments	Languages	SRL Style	Same Frames?	Compatible Roles?	Main Setting
EWT/UPB† (§6.3.2)	de,fr,it,es,pt,fi	dependency	Yes	Yes	Zero-shot
EWT/FiPB (§6.3.3)	fi	dependency	No	Yes	Semi-supervised
CoNLL-2009 (§6.3.4)	cs,zh,es,ca	dependency	No	No	Semi-supervised
OntoNotes (§6.3.5)	zh,ar	span	No	Yes	Semi-supervised

Table 6.1: An overview of our experiments.

We conduct comprehensive experiments with three groups of datasets: 1) English Web Treebank (EWT) (Silveira et al., 2014), Universal Proposition Banks (UPB v1.0) (Akbik et al., 2015, 2016b) and Finnish PropBank (FiPB) (Haverinen et al., 2015); 2) CoNLL-2009 (Hajič et al., 2009); and 3) OntoNotes v5.0 (Hovy et al., 2006; Weischedel et al., 2013). Table 6.1 gives an overview of our main experimental settings. Here "Same Frames?" denotes whether different languages utilize the same semantic frames, and "Compatible Roles?" denotes whether the roles labels are the same. (†UPB is created semi-automatically, while other datasets use directly or are converted from manual annotations.)

We mainly take English as the source language and transfer to other target languages, unless otherwise noted. For experiments on UPB and FiPB, we assemble the English SRL dataset with EWT and its SRL annotations from PropBank v3. For CoNLL-2009 and OntoNotes, we utilize the corresponding English sets. For evaluation, we calculate the labeled F1 score for arguments. Conventionally, predicate senses are also evaluated for dependency-based SRL. However, cross-lingual transfer of sense disambiguation provides a non-trivial challenge (Akbik et al., 2016a), since it is lexicon-based and language-dependent. Moreover, argument labeling can be more related to dependency syntax, while sense disambiguation is more on the semantic side and semantic-oriented signals (like bilingual dictionaries or parallel corpora) may be more directly effective to enhance cross-lingual transfer. Therefore, in this work, we focus on arguments and do not perform or evaluate sense disambiguation, following the conventions of span-based SRL.

For syntactic resources, we use either UD treebanks or convert constituency trees to dependencies using Stanford CoreNLP (Manning et al., 2014). In most of our settings, we assume access to multilingual syntax annotations for both source and target languages. We regard this as a practical setting since UD treebanks are available for a wide range of languages and syntactic annotations may be easier to obtain than semantic ones.

We adopt pre-trained multilingual language models (multilingual BERT Devlin et al. (2019) or XLM-R Conneau et al. (2020)) to initialize our encoders and fine-tune the full models. We use the Adam optimizer Kingma and Ba (2014) with an initial learning rate of 2e-5. We train the models for 100K steps with a batch size of around 1024 tokens for each step. All models are trained and evaluated on one GTX 1080 Ti GPU, and training one model usually takes around half a day.

6.3.2 UPB

UPB annotates[2] target languages with English PropBank frames, which allows us to explore zero-shot experiments without any target SRL training resources. We follow the setting of (Fei et al., 2020a): training the models with English SRL annotations (EWT) and directly applying them to the target languages. In this experiment only we assume predicates are given since UPB is limited to verbal predicates, which leads to discrepancies between source and target predicate annotations. For the syntactic resources, we take the corresponding treebanks (upon which UPB is annotated) from UD v1.4 (Nivre et al., 2016a) and simply include them as additional training data for syntactic supervision.

Comparisons

We first compare several strategies on the usage of syntax, and results on the development set are shown in Table 6.2. Here we utilize multilingual BERT (mBERT) for the basic encoder. The table is split into three groups:

[2]Notice that UPB is created in a semi-automatic way without fully manually validated test sets, but it provides a test-bed for evaluating zero-shot cross-lingual transfer.

Method	DE	FR	IT	ES	PT	FI	AVG
NoSyn	$57.85_{\pm 0.34}$	$51.41_{\pm 0.18}$	$55.79_{\pm 0.42}$	$50.08_{\pm 0.16}$	$52.53_{\pm 0.40}$	$43.78_{\pm 0.57}$	$51.91_{\pm 0.17}$
EnSyn	$57.78_{\pm 0.43}$	$51.64_{\pm 0.16}$	$54.90_{\pm 0.68}$	$50.15_{\pm 0.36}$	$52.65_{\pm 0.22}$	$44.41_{\pm 0.76}$	$51.92_{\pm 0.31}$
TargetSyn	$56.84_{\pm 1.42}$	$57.55_{\pm 1.01}$	$55.78_{\pm 2.04}$	$52.40_{\pm 1.20}$	$56.32_{\pm 1.54}$	$52.32_{\pm 1.20}$	$55.20_{\pm 1.20}$
FullSyn	$59.70_{\pm 0.61}$	$59.38_{\pm 0.37}$	$60.75_{\pm 0.28}$	$55.57_{\pm 0.36}$	$59.78_{\pm 0.28}$	$55.94_{\pm 0.56}$	$58.52_{\pm 0.20}$
SEQ(MLM)	$57.52_{\pm 0.67}$	$52.09_{\pm 0.77}$	$56.56_{\pm 0.55}$	$50.60_{\pm 0.47}$	$53.34_{\pm 0.47}$	$44.56_{\pm 0.80}$	$52.45_{\pm 0.29}$
SEQ(Syn)	$59.73_{\pm 0.39}$	$56.05_{\pm 0.44}$	$61.11_{\pm 0.29}$	$55.32_{\pm 0.29}$	$58.15_{\pm 0.21}$	$55.23_{\pm 0.49}$	$57.60_{\pm 0.15}$
GCN(Gold)	$63.78_{\pm 0.50}$	$56.44_{\pm 0.40}$	$61.96_{\pm 0.73}$	$56.77_{\pm 0.36}$	$59.79_{\pm 0.28}$	$55.29_{\pm 0.33}$	$59.01_{\pm 0.30}$
GCN(Pred)	$61.15_{\pm 0.37}$	$55.44_{\pm 0.36}$	$60.69_{\pm 0.65}$	$54.49_{\pm 0.37}$	$58.09_{\pm 0.27}$	$53.75_{\pm 0.32}$	$57.27_{\pm 0.29}$

Table 6.2: UPB development Arg-F1(%) scores in the English-to-others zero-shot setting (with mBERT).

- The first group varies which **Syn**tactic resources are used. The four rows denote no syntax (NoSyn), only source (English; EnSyn), only targets (other six languages; TargetSyn) and full syntactic resources (English plus other six; FullSyn). Here, only adding source syntax is not helpful, but target syntax information is generally beneficial. Furthermore, combining both source and target syntax leads to the best results.
- The second group explores a **SEQ**uential two-stage fine-tuning scheme (Phang et al., 2018; Wang et al., 2019a): first training the model with an auxiliary task (syntax or others) in an intermediate stage and then with the target task (SRL). Using syntactic parsing as the intermediate task can bring clear improvements, but it is slightly worse than the MTL scheme. Here, we also explore a masked language model (MLM) intermediate objective (Devlin et al., 2019) as a baseline, using the raw texts of the UD treebanks. Though it can slightly improve the results, the gains are much smaller than those brought by syntax.
- In the final group, we utilize syntax as inputs. We stack a Graph Convolutional Network (**GCN**) (Kipf and Welling, 2017) between the encoder and the decoders to encode input dependency trees. Specifically, we adopt the architecture of (Marcheggiani and Titov, 2017). Using gold trees in this setting out-performs the MTL strategy. However, when using predicted syntax[3], error propagation seems to drag the results down. In this way, the MTL scheme is an attractive alternative strategy considering its competitive performance and model simplicity.

Main Results

The test results are listed in Table 6.3. Similar to the trends in the development sets, including syntactic signals brings clear improvements, especially for the more distant Finnish language. Using XLM-R, which is pre-trained on more data than mBERT, is also helpful,[4] upon which syntax can still bring further benefits. We also compare with the results from (Fei et al., 2020a), which translates and projects source SRL instances to target languages for training. The

[3]We obtain predicted syntax trees with our own BERT-based parsers, which achieve strong results (dev-LAS%): 89.6(de), 91.6(fr), 93.7(it), 89.7(es), 92.1(pt) and 93.2(fi).

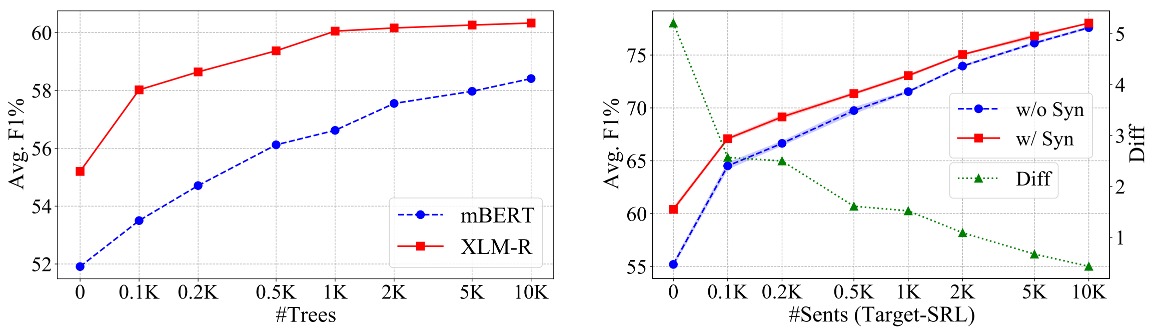
[4]Due to better performance, XLM-R is used in the remaining experiments.

Method	DE	FR	IT	ES	PT	FI
mBERT/NoSyn	55.0	49.9	53.1	49.7	51.0	44.7
mBERT/FullSyn	57.5	56.8	58.3	56.2	58.9	54.4
XLM-R/NoSyn	57.5	50.8	54.3	51.5	53.1	51.8
XLM-R/FullSyn	60.2	56.6	**60.6**	57.3	**59.5**	**59.9**
Fei et al. (2020a)	**65.0**	**64.8**	58.7	**62.5**	56.0	54.5

Table 6.3: UPB test Arg-F1(%) scores in the English-to-others zero-shot setting (averaged over five runs).

translation-based method performs strongly for German, French and Spanish. Considering that German and French are commonly used languages in machine translation research, availability of high-quality translation systems may be one of the contributing factors. Our syntax-enhanced models are generally competitive for other languages.

Varying Training Sizes



(a) Averaged results versus number of trees (in log scale) used per language (using mBERT or XLM-R for encoder).

(b) Averaged results versus number of SRL target sentences (in log scale) utilized per target language (using XLM-R for encoder).

Figure 6.1: Further results on UPB development sets.

We further vary the number of available syntax trees for the auxiliary parsing task, for which Figure 6.1a shows the results. We randomly sample a fixed number of trees for each of the languages (both source and target) and again include them in training. The results indicate that we do not need the full treebanks to obtain good results. Especially with XLM-R, 1K trees from each language can already lead to gains comparable to the 10K case.

We also experiment with semi-supervised settings on the UPB datasets. We still take English as the source and randomly sample SRL training instances for target languages and train the models alongside all these source examples. The results are shown in Figure 6.1b, where adding target SRL annotations can bring obvious improvements. Nevertheless, including syntactic supervision is still helpful, particularly in low-resource scenarios.

6.3.3 FiPB

Similar[5] to the experiments on UPB, we take English SRL annotations from EWT as the source. FiPB adopts (almost) the same argument role set[6] as the English ones and we use a shared SRL decoder for both languages. In preliminary experiments, we find that this sharing strategy performs better than using separate, language-specific decoders. For syntax, we again take corresponding English and Finnish treebanks from UD v1.4.

Results and Analyses

EnSRL	Syntax	0.1K Dev	0.1K Test	1K Dev	1K Test	10K Dev	10K Test
No	No	$43.25_{\pm 0.50}$	$44.76_{\pm 0.82}$	$69.02_{\pm 0.32}$	$70.29_{\pm 0.54}$	$82.51_{\pm 0.40}$	$82.91_{\pm 0.35}$
No	Yes	$58.75_{\pm 0.49}$	$58.32_{\pm 0.80}$	$73.91_{\pm 0.33}$	$74.06_{\pm 0.30}$	$82.71_{\pm 0.13}$	$83.24_{\pm 0.17}$
Yes	No	$60.76_{\pm 0.56}$	$60.42_{\pm 0.89}$	$73.23_{\pm 0.37}$	$73.67_{\pm 0.68}$	$\mathbf{82.92}_{\pm 0.30}$	$\mathbf{83.35}_{\pm 0.27}$
Yes	Yes	$\mathbf{68.36}_{\pm 0.17}$	$\mathbf{67.22}_{\pm 0.39}$	$\mathbf{75.98}_{\pm 0.14}$	$\mathbf{76.13}_{\pm 0.18}$	$82.73_{\pm 0.18}$	$83.34_{\pm 0.22}$

Table 6.4: FiPB Arg-F1(%) scores in English/Finnish settings (with different numbers of Finnish SRL sentences).

The main results on FiPB are listed in Table 6.4. Here, "EnSRL" indicates whether using English SRL, and "Syntax" denotes whether using syntactic annotations. In the lowest-resource scenario (0.1K Finnish SRL sentences), both English SRL and syntax are quite helpful, and combining them leads to further improvements. The trend is similar if given 1K target SRL annotations, but the gaps decrease. Finally, when given enough target training instances as in the 10K scenario, the gains due to extra resources (either English SRL or syntax) are negligible. In this case, the model may have already learned most of the patterns from rich target SRL annotations.

We further perform analysis on the development results in the 1K case, as shown in Table 6.5. Here, the first block denotes breakdowns on argument roles, the second denotes the syntactic distance between predicate and argument words, and the third denotes the syntactic path between them. The numbers in parentheses denote percentages. **Bold** and <u>underlined</u> numbers indicate the best and second-best results respectively.

In the first group of role label breakdowns, adding syntax particularly helps core arguments while adding English SRL helps more on non-core arguments. Finally, combining both leads to the best results overall. In the second group, we break down arguments by their syntactic distance to the predicates. The results show that syntactic supervision is still beneficial when the predicate and the argument are two edges away (d=2). However, when syntax distance is larger, direct syntactic supervision becomes less helpful.

In the third group, we look at the labeled syntactic paths between the arguments and the predicates. For example, "$\xleftarrow{\text{nmod}}$" denotes that the argument is a syntactic modifier of the predicate

[5]Starting from this experiment, we focus on the semi-supervised setting where varying amounts of target SRL annotations are used during training.

[6]Except for two Finnish specific roles (ArgM-CSQ and ArgM-PRT) which only account for around 2% of labels.

	Base	+Syntax	+EnSRL	+Both
ARG0 (12%)	75.72	81.08	80.11	**82.94**
ARG1 (36%)	77.21	83.26	81.29	**84.24**
ARG2 (14%)	65.85	70.41	69.93	**71.84**
ARGM (35%)	60.76	64.73	65.46	**68.33**
d=1 (84%)	75.62	78.45	78.05	**80.35**
d=2 (15%)	58.41	63.26	61.31	**64.22**
d>2 (1%)	24.07	25.84	**28.61**	25.39
$\xleftarrow{\text{nmod}}$ (23%)	61.01	62.95	64.69	**66.80**
$\xleftarrow{\text{nsubj}}$ (14%)	85.72	**88.15**	85.35	87.89
$\xleftarrow{\text{dobj}}$ (13%)	90.10	**93.37**	91.20	93.25
$\xleftarrow{\text{advmod}}$ (9%)	64.40	65.17	68.18	**69.04**
$\xrightarrow{\text{acl}}$ (4%)	83.44	85.17	86.20	**87.13**
$\xrightarrow{\text{cop}}$ (4%)	91.63	**97.79**	94.24	96.63
$\xleftarrow{\text{xcomp}}$ (3%)	70.83	75.21	73.01	**77.29**
$\xleftarrow{\text{aux}}$ (3%)	95.21	97.53	95.67	**97.97**
$\xleftarrow{\text{nsubj}}\xrightarrow{\text{cop}}$ (3%)	95.75	98.37	94.94	**98.60**
$\xleftarrow{\text{advcl}}$ (3%)	51.80	57.83	68.64	**70.31**

Table 6.5: Analysis (F1% breakdown) on the FiPB development set (1K setting).

and the dependency relation is "nmod", while "$\xrightarrow{\text{acl}}$" denotes the argument is the syntactic head of the predicate with the dependency relation of "acl". We show the results on top-ten frequent paths, which cover around 80% of all the arguments. According to the breakdown results, syntactic supervision helps more on the edges of subject, direct object and some functional relations (like copula), while English SRL is more beneficial on the more semantic links, such as adverbial words and clauses. This agrees with our analysis of the argument roles: the syntax helps more on the core arguments, which are usually directly connected as subjects or objects, while English SRL helps more on "ArgM"s, which tend to be adverbial.

Varying Training Sizes

We further vary both syntax and target-SRL training sizes, and the influence on model performance is shown in Figure 6.2. Here, all the models are trained using all English SRL and varying amounts of Finnish SRL sentences. The numbers in parentheses at the y-axis show the F1 scores of baseline models without syntax. As expected, syntax is more helpful when we have less target-SRL and more syntactic resources (towards the right corner of the figure). When we have more target SRL annotations, syntactic resources become less helpful. Nevertheless, in low-resource scenarios, even small quantities of syntactic annotation can bring clear improvements.

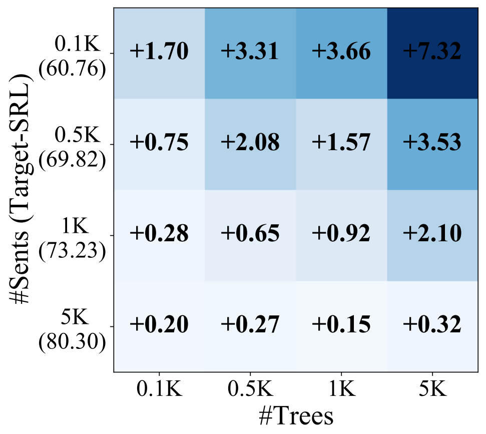


Figure 6.2: Improvements (F1 scores on FiPB development set) over no-syntax baselines (shown in parentheses at the y-axis) with various training sizes.

No Pre-trained Initialization

In the main experiments, we utilize pre-trained multilingual language models to initialize the encoders. Here, we explore the case where no such initialization is performed. All other settings are the same as the previous, except that the models are all randomly initialized. The training scheme is slightly modified: we perform learning-rate warmup for the first 10K steps and increase the maximum learning rate to 1e-4. The results on the development sets are shown in Table 6.6. Here, "EnSRL" indicates whether using English SRL, and "Syntax" denotes whether using syntax. There is no surprise that the scores are much lower than those with pre-trained models. Interestingly, though both English SRL and syntax can provide improvements in both low-resource and high-resource cases, syntax is much more helpful. A possible reason is that the multilingual pre-training provides shared representations across languages, without which the extra supervision from other languages may be much less effective.

EnSRL	Syntax	0.1K	1K	10K
No	No	$3.03_{\pm 0.82}$	$13.58_{\pm 0.59}$	$44.11_{\pm 0.40}$
No	Yes	$28.54_{\pm 1.35}$	$41.50_{\pm 0.68}$	$56.14_{\pm 0.29}$
Yes	No	$5.81_{\pm 0.18}$	$20.29_{\pm 0.42}$	$48.57_{\pm 0.21}$
Yes	Yes	$\mathbf{39.06}_{\pm 0.40}$	$\mathbf{46.05}_{\pm 0.75}$	$\mathbf{58.05}_{\pm 0.34}$

Table 6.6: FiPB development Arg-F1(%) scores in English/Finnish settings (with different numbers of Finnish SRL sentences) with randomly initialized encoders.

6.3.4 CoNLL-2009

The original SRL annotations of CoNLL-2009 are based on language-specific syntax, causing the argument head words to disagree with UD conventions. We thus follow Pražák and Konopík

(2017) and convert them to UD-based argument heads. We take five languages from CoNLL-2009 where we can obtain corresponding UD trees for the source sentences. For English and Chinese, we use Stanford CoreNLP to convert the constituency trees to dependencies, while for Czech, Spanish, and Catalan, we assign dependency trees from corresponding UD v2.7 (Zeman et al., 2020) treebanks (PDT for Czech and AnCora for Spanish and Catalan). Moreover, since role labels are not compatible across languages in CoNLL-2009, we utilize separate SRL decoders for source and target languages.

UD-based Conversion

The SRL annotations of argument heads in CoNLL-2009 are based on Language-Specific Dependency (**LSD**) trees rather than Universal Dependencies (**UD**). To convert argument heads between different syntactic formalisms, we adopt a simple path-based method. Assuming that a predicate p has an argument whose head is a according to the original tree, the conversion aims to find a new head according to the new tree:

1. In the new tree, find the lowest common ancestor c of the predicate p and the original argument head a.
2. Go down from c to a in the new tree, locate the first word (except for the predicate p) that is a descendant of a (or a itself) in the original tree, and make it the new head.

We will illustrate this procedure with the example in Figure 6.3. Here, the predicate is the verb "ran" and it has an "ArgM-LOC" argument, whose full span is "in the park". According to the language-specific dependency tree, the word "in" is the direct child of the verb and thus becomes the argument head. Nevertheless, according to UD, the content word "park" is the direct child and we want to convert the argument head to it. Firstly, we find the lowest common ancestor of "ran" (the predicate) and "in" (the old argument head) in the UD tree, which is "ran" itself. Then we go down from this ancestor ("ran") towards the old argument head ("in"): the visiting path should be $ran \rightarrow park \rightarrow in$. We find that "park" is the first word that is a descendant of "in" in the original tree and therefore "park" is assigned as the new argument head.

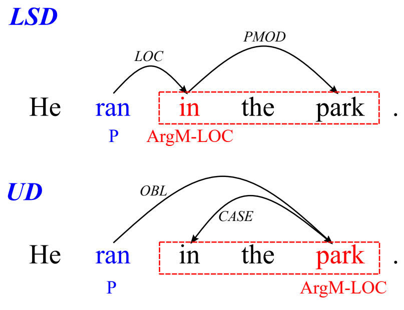


Figure 6.3: An example for the conversion between LSD and UD.

Language	UAS%	Agree%	R-Agree%
English	50.88	72.27	99.05
Czech*	46.36	97.00	73.38
Chinese	60.02	81.94	99.87
Spanish	58.01	69.92	100.00
Catalan	58.99	72.61	100.00

Figure 6.4: Argument agreements between LSD and UD.

In this work, we take five languages from CoNLL-2009: English, Czech, Chinese, Spanish, and Catalan, for which we can obtain or convert to gold UD trees. For Chinese and English, we use CoreNLP to convert from constituencies to UD. For Czech, we use UD_PDT, while for Spanish and Catalan, we use UD_AnCora. Figure 6.4 gives some results on the agreements between different syntactic formalisms. Here, "UAS" denotes the unlabeled attachment scores when comparing LSD and UD trees, "Agree" denotes the agreement rates on argument heads between original argument heads and those converted to UD, while "R-Agree" denotes the agreement rates with **R**ound-trip styled conversions: first converting from LSD to UD and then converting back to LSD. Note that Czech is a special case where the original argument heads seem to mostly agree with UD. Although on overall syntactic attachments, LSD disagrees much with UD (the highest UAS is 60% for Chinese), the argument head agreement rates are much higher (the lowest argument agreement rate is around 70% for Spanish). If adopting our converting method, a round-trip styled conversion (converting from LSD to UD and then back to LSD) can almost recover all the arguments, showing the effectiveness of our method. Notice that Czech is an exception where the original argument heads seem to already mostly follow the UD trees.

Results

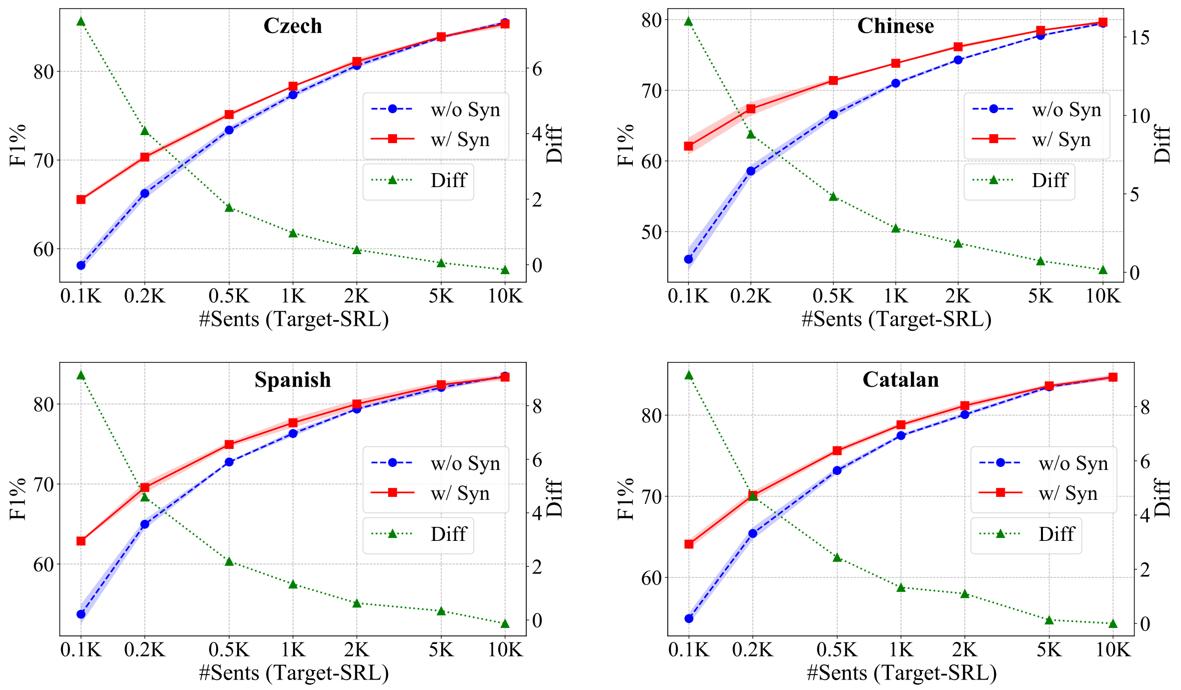


Figure 6.5: Test results of CoNLL-2009 semi-supervised experiments.

We again take English as the resource-rich source language and the other four as lower-resource targets. We run experiments separately for each target language, which means all experiments are bilingual (with the exception of XLM-R pretraining). For syntax, since different languages have different treebank sizes, we randomly sample 10K trees for both source and target languages. The results are shown in Figure 6.5. Here the x-axis denotes the number (in log scale) of target-SRL annotated sentences available for training. The results are averaged over three runs, and the shaded areas indicate the ranges of standard deviations. The patterns are consistent among all languages and similar to previous experiments on FiPB: syntax is clearly helpful in low-resource scenarios, but as we have access to more target SRL annotations, the gaps decrease and finally diminish in the high-resource scenarios.

Using Other Treebanks

Syntax	Spanish		Catalan	
	LAS%	ArgF1%	LAS%	ArgF1%
NoSyntax	-	$54.6_{\pm 1.2}$	-	$54.0_{\pm 0.9}$
Spanish	$\mathbf{86.9}_{\pm 0.1}$	$\mathbf{63.6}_{\pm 0.7}$	$67.9_{\pm 7.1}$	$59.0_{\pm 0.9}$
Catalan	$77.0_{\pm 1.0}$	$61.0_{\pm 0.9}$	$\mathbf{85.7}_{\pm 0.4}$	$\mathbf{63.9}_{\pm 0.2}$
French	$64.2_{\pm 9.1}$	$57.9_{\pm 0.8}$	$58.7_{\pm 2.0}$	$55.4_{\pm 0.8}$
Italian	$66.1_{\pm 3.6}$	$58.1_{\pm 0.5}$	$56.4_{\pm 6.4}$	$57.0_{\pm 0.9}$
Portuguese	$69.5_{\pm 3.0}$	$57.6_{\pm 1.7}$	$58.6_{\pm 5.7}$	$56.8_{\pm 0.8}$

Table 6.7: Development results of Spanish and Catalan CoNLL-2009 semi-supervised experiments (0.1K target-SRL) using syntax from different languages.

We further explore the scenarios where we do not directly have syntactic annotations for the target language. Considering that the parsing task can also benefit from cross-lingual transfer, we can utilize treebanks from nearby languages for syntactic supervision. We take Spanish and Catalan (the 0.1K target SRL case) for this analysis and the results are shown in Table 6.7. We further explore three Romance languages: French, Italian, and Portuguese. As expected, directly using target-language syntax obtains the best results. Spanish and Catalan, which are closely related languages, benefit each other the most. Nevertheless, compared with the NoSyntax baseline, syntactic information from all these languages are helpful. This result is of practical interest when transferring to a truly low-resource language where syntactic annotations may also be limited. Finding a related language with rich syntactic resources for auxiliary training signals is a promising way to improve performance.

Similar to the analysis in Table 6.5, We further provide a breakdown analysis on syntactic paths for the Spanish results, which is shown in Table 6.8. The trends are very similar to the overall ones. Compared with the NoSyntax baseline, adding syntax supervision of either directly target language Spanish or other related languages could bring improvements on most syntactic paths. And there is no surprise that direct Spanish syntax is the best auxiliary signal, followed by Catalan. One potential factor that influences the effectiveness of the transfer could be the word order difference between different languages. To investigate this, we calculate the word order differences for syntactic paths and calculate the correlations between them and SRL performance differences. Unfortunately, we did not find clear correlations between these two. There may be other factors that have more impacts on the transfer, for example, Catalan has more vocabulary overlaps with Spanish than other languages, which can be one of the reasons why Catalan supervision is more helpful for Spanish SRL.

	NoSyntax	Spanish	Catalan	French	Italian	Portuguese
$\xleftarrow{\text{nsubj}}$ (29%)	68.96	**75.41**	74.61	72.74	71.57	71.24
$\xleftarrow{\text{obj}}$ (25%)	62.91	**67.76**	67.14	67.15	66.94	65.73
$\xleftarrow{\text{obl}}$ (17%)	36.20	44.85	**45.31**	38.63	39.46	39.07
$\xleftarrow{\text{advmod}}$ (6%)	47.82	**55.73**	52.79	49.06	49.84	47.42
$\xleftarrow{\text{ccomp}}$ (4%)	63.28	**76.29**	70.45	64.30	63.61	66.08
$\xrightarrow{\text{cop}}$ (4%)	80.71	**92.13**	88.81	85.26	84.72	87.02
$\xleftarrow{\text{advcl}}$ (4%)	14.54	**31.93**	27.30	14.94	21.52	18.91
$\xleftarrow{\text{xcomp}}$ (3%)	53.36	**63.36**	58.79	56.31	52.66	57.57
$\xleftarrow{\text{nsubj}}\xrightarrow{\text{cop}}$ (2%)	71.22	**90.37**	82.94	79.18	76.31	77.82

Table 6.8: Analysis (F1% breakdown) on the CoNLL-2009 Spanish development set.

6.3.5 OntoNotes

Finally, we turn to span-based SRL where the extraction of full argument spans is required. Utilizing OntoNotes annotations, we still mainly take English as the source and Chinese or Arabic as the target. Similar to FiPB, the argument roles are compatible with PropBank-style English roles and we use a shared SRL decoder for both the source and target languages. We adopt the data split from the CoNLL12 shared task (Pradhan et al., 2012). For English and Chinese, we convert constituencies to dependencies with Stanford CoreNLP. For Arabic, we assign dependency trees from Arabic-NYUAD (Taji et al., 2017) treebank of UD v2.7.

Results and Analyses

Method	Syntax	0.1K		1K		10K	
		Dev	Test	Dev	Test	Dev	Test
Chinese							
BIO	No	$50.59_{\pm 0.38}$	$49.67_{\pm 0.28}$	$62.81_{\pm 0.15}$	$62.58_{\pm 0.24}$	$70.04_{\pm 0.18}$	$70.37_{\pm 0.08}$
TwoStep	No	$49.26_{\pm 0.44}$	$48.53_{\pm 0.67}$	$63.08_{\pm 0.05}$	$63.22_{\pm 0.12}$	$70.43_{\pm 0.12}$	$70.80_{\pm 0.12}$
BIO	Yes	$53.47_{\pm 0.24}$	$52.81_{\pm 0.20}$	$64.55_{\pm 0.21}$	$64.49_{\pm 0.11}$	$70.26_{\pm 0.18}$	$70.58_{\pm 0.22}$
TwoStep	Yes	$\mathbf{56.16}_{\pm 0.14}$	$\mathbf{55.52}_{\pm 0.23}$	$\mathbf{65.36}_{\pm 0.22}$	$\mathbf{65.65}_{\pm 0.15}$	$\mathbf{70.66}_{\pm 0.13}$	$\mathbf{71.04}_{\pm 0.12}$
Arabic							
BIO	No	$46.14_{\pm 0.73}$	$44.87_{\pm 1.10}$	$59.72_{\pm 0.51}$	$58.80_{\pm 0.26}$	$69.67_{\pm 0.12}$	$67.87_{\pm 0.42}$
TwoStep	No	$46.33_{\pm 0.26}$	$45.28_{\pm 0.48}$	$59.91_{\pm 0.39}$	$59.53_{\pm 0.62}$	$70.17_{\pm 0.25}$	$\mathbf{68.46}_{\pm 0.30}$
BIO	Yes	$49.23_{\pm 0.27}$	$49.13_{\pm 0.31}$	$61.36_{\pm 0.23}$	$60.89_{\pm 0.21}$	$70.02_{\pm 0.25}$	$67.92_{\pm 0.22}$
TwoStep	Yes	$\mathbf{51.68}_{\pm 0.33}$	$\mathbf{51.50}_{\pm 0.50}$	$\mathbf{61.81}_{\pm 0.45}$	$\mathbf{61.70}_{\pm 0.61}$	$\mathbf{70.19}_{\pm 0.16}$	$68.28_{\pm 0.31}$

Table 6.9: OntoNotes Arg-F1(%) scores in English-sourced semi-supervised settings (with different numbers of target SRL training sentences).

In this experiment, we specifically compare two SRL decoders. The first one casts the task as a BIO-based sequence labeling problem. We further add a standard linear-chain conditional random field (CRF) (Lafferty et al., 2001), which we found consistently helpful in preliminary experiments. The other one is the two-step decoder described in §6.2.3. As shown in Table 6.9, the trends are similar for both Chinese and Arabic. With regard to auxiliary syntactic supervision, we find similar trends to previous experiments: in low-resource scenarios, syntactic supervision is beneficial for both decoders, but as the availability of target SRL resources increases, the gaps become smaller until diminished. The more interesting comparisons are between the two decoders: when not using syntactic supervision, their performances are comparable; but when trained with auxiliary signals from syntax, the syntax-aware two-step decoder performs better than the BIO tagger, especially in low-resource cases.

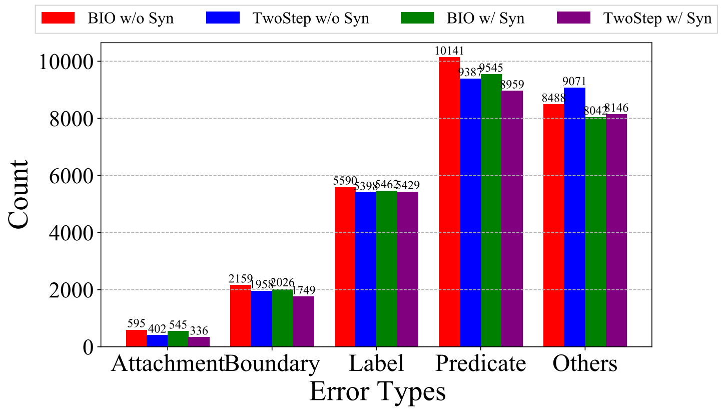


Figure 6.6: Error breakdowns of arguments on the OntoNotes Chinese development set.

We further perform error analysis on the Chinese development set in the 1K setting. The error categories are similar to those in §3.2.2. As shown in Figure 6.6, syntactic supervision and the syntax-aware TwoStep decoder make fewer errors related to phrasal attachments, span boundaries, and predicate identification. Notice that the first two categories are closely related to syntax, which may explain why syntax-informed models make fewer such errors. In particular, the two-step model trained with syntactic supervision makes the fewest syntax-related errors. Together with its generally better overall F1 scores, these demonstrate the benefits of utilizing syntactic information alongside a suitable syntax-aware model.

Other Languages as Source

We further explore experiments taking Chinese or Arabic as the source and English as the target. The development results are shown in Table 6.10. The general trends are very similar to those in the English-as-source experiments, where syntax supervision is generally helpful, especially in low-resource scenarios.

Syntax with Genre Mismatches

Since English and Chinese OntoNotes also annotate six different genres of text, we further explore scenarios where the syntax and SRL datasets have genre mismatches. We still take all En-

Method	Syntax	0.1K	1K	10K
Chinese → English				
BIO	No	$57.29_{\pm 0.70}$	$71.41_{\pm 0.53}$	$79.43_{\pm 0.05}$
TwoStep	No	$57.17_{\pm 1.17}$	$72.64_{\pm 0.09}$	$\mathbf{79.95}_{\pm 0.09}$
BIO	Yes	$56.67_{\pm 0.52}$	$72.03_{\pm 0.28}$	$79.41_{\pm 0.10}$
TwoStep	Yes	$\mathbf{60.42}_{\pm 0.27}$	$\mathbf{73.53}_{\pm 0.14}$	$79.77_{\pm 0.06}$
Arabic → English				
BIO	No	$59.78_{\pm 0.37}$	$72.07_{\pm 0.20}$	$79.34_{\pm 0.05}$
TwoStep	No	$59.71_{\pm 0.71}$	$72.58_{\pm 0.22}$	$79.61_{\pm 0.21}$
BIO	Yes	$58.03_{\pm 0.73}$	$72.48_{\pm 0.09}$	$79.28_{\pm 0.08}$
TwoStep	Yes	$\mathbf{60.92}_{\pm 0.14}$	$\mathbf{73.23}_{\pm 0.10}$	$\mathbf{79.66}_{\pm 0.24}$

Table 6.10: OntoNotes English development Arg-F1(%) scores in semi-supervised settings (with different number of target SRL training sentences), using Chinese or Arabic as the source language.

glish instances for multilingual training but split the Chinese corpus according to genres, including broadcast conversation (bc), broadcast news (bn), magazine (mz), newswire (nw), telephone conversation (tc) and web (wb). We focus on the low-resource scenario where 0.1K Chinese SRL sentences on the target genre are available. The development results are shown in Figure 6.7. When the genre of syntactic supervision matches the target SRL, the improvements are the largest. Nevertheless, even in the case of genre mismatches, syntax can still be beneficial, especially within similar genres. We further find a positive correlation (Pearson correlation is 0.73; Spearman is 0.78) between these improvements and genre similarities calculated by the centroids of mBERT representations (Aharoni and Goldberg, 2020). This may provide a mechanism for selecting the most beneficial syntactically annotated instances.

6.4 Related Work

Recently there has been increasing interest in cross-lingual SRL, where SRL annotations from high-resource languages are utilized to help low-resource ones. One straightforward method is data transfer, using either annotation projection (Yarowsky and Ngai, 2001) or translation (Tiedemann and Agić, 2016) to create SRL instances for target languages (Padó and Lapata, 2009; Akbik et al., 2015; Aminian et al., 2019; Fei et al., 2020a). A related idea is to utilize parallel corpus to introduce cross-lingual signals (Daza and Frank, 2019, 2020; Cai and Lapata, 2020). Another method is model-transfer which we focus on in this work: directly applying the model trained with source languages to target ones (Kozhevnikov and Titov, 2013, 2014; Fei et al., 2020b). This method requires shared representations for different languages, which recent multilingual pre-trained encoders (Devlin et al., 2019; Conneau et al., 2020) are good at (Wu and Dredze, 2019; Pires et al., 2019). We take these multilingual encoders as the backbone of our models since they have been shown effective for SRL across multiple languages (He et al., 2019; Conia and Navigli, 2020). In the semi-supervised settings where some target SRL annotations are available, the models are actually trained in a polyglot way (Mulcaire et al., 2018, 2019). Our

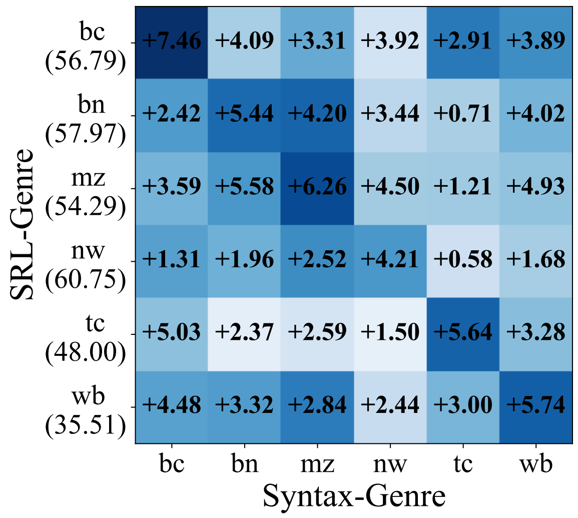


Figure 6.7: Improvements (F1 scores on Chinese OntoNotes development set) over no-syntax baselines (shown in parentheses at the y-axis) with syntactic supervision of different genres.

work differs in the focus on low-resource scenarios.

Cross-lingual SRL still remains challenging due to data scarcity and annotation heterogeneity. To create multilingual SRL data, Akbik et al. (2015) utilize parallel corpus to create target SRL annotations with filtered projection, and Daza and Frank (2020) create the X-SRL dataset through translation and projection with multilingual contextualized representations, offering a multilingual parallel SRL corpus. Tripodi et al. (2021) create UniteD-SRL, providing a unified dataset for span- and dependency-based multilingual and cross-lingual SRL. To deal with heterogeneous SRL formalism, Jindal et al. (2020) adopt an argument regularizer to encourage cross-lingual argument matching, and Conia et al. (2021) introduce a unified model, which may implicitly learn to align heterogeneous linguistic resources.

Even with recent developments in neural network modeling, with which syntax-agnostic models have been shown to match linguistically-informed counterparts (Marcheggiani et al., 2017; Cai et al., 2018), syntax has also been found helpful for SRL (Marcheggiani and Titov, 2017; Swayamdipta et al., 2018; Strubell et al., 2018; He et al., 2018b; Cai and Lapata, 2019; Shi et al., 2020; Fei et al., 2021). In this work, we further explore the helpfulness of syntax for cross-lingual SRL. While previous work on this topic mainly uses syntax as input features (Kozhevnikov and Titov, 2013; Pražák and Konopík, 2017; Fei et al., 2020b), we adopt a simpler strategy utilizing it as an auxiliary training signal via multitask learning (Caruana, 1997; Ruder, 2017), which has also been found beneficial for monolingual SRL (Swayamdipta et al., 2018; Strubell et al., 2018; Cai and Lapata, 2019).

6.5 Conclusion

In this chapter, we provide a comprehensive empirical exploration of the helpfulness of syntactic supervision for cross-lingual SRL. With extensive evaluations across a variety of datasets and settings, we show that auxiliary syntactic signals are generally beneficial, especially in low-resource SRL cases. We hope that our investigation can shed some light on the relations between syntax and SRL in cross-lingual scenarios.

Chapter 7

Transfer Learning from Semantic Role Labeling to Event Argument Extraction with Template-based Slot Querying

In the previous chapter, we explore the relationship between syntax and SRL. In this chapter, we further study transfer learning between related tasks and extend our method to the task of event argument extraction in information extraction. We show that SRL annotations can be valuable training resources for predicting event arguments.

7.1 Introduction

Event argument extraction (EAE) is a key component in the task of event extraction (Ahn, 2006) that aims to identify the arguments that serve as roles for event frames. While recent developments in neural network models have enabled impressive improvements on this task in the fully-supervised setting (Wang et al., 2019b; Pouran Ben Veyseh et al., 2020; Ma et al., 2020; Li et al., 2021a), EAE remains challenging when abundant annotations are not available. In particular, event schemes are usually *specific* to the target scenarios. For example, events in biomedical domains, like GENE-EXPRESSION in GENIA (Kim et al., 2008), can be quite different than the ones in ACE (LDC, 2005), such as ATTACK and CONTACT. It is costly and inefficient to annotate large amounts of data for every new application.

Compared with the *specific* schemes in EAE, semantic role labeling (SRL; Gildea and Jurafsky, 2002; Palmer et al., 2010) extracts predicate-argument structures with more *general* and broad-coverage frame ontologies. SRL also enjoys rich and carefully annotated resources, such as PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998b), covering a wide range of semantic frame types. As shown in the example in Figure 7.1, SRL closely resembles EAE: they both specify semantic frames triggered by predicate words and aim at finding arguments for participating roles. Therefore, it is natural to consider applying transfer learning[1] (Pan and Yang,

[1]Because of the similarities, EAE and SRL may be arguably viewed as two versions of the same task. Even in this case, we can still view this as a special form of transductive transfer learning, if not inductive transfer on different tasks.

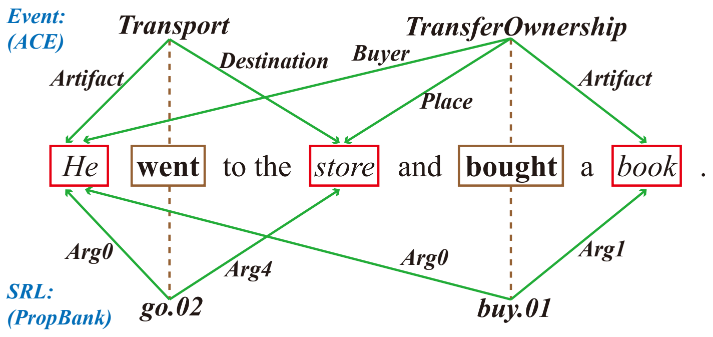


Figure 7.1: Example annotations with ACE events and PropBank semantic frames.

2009; Ruder et al., 2019) to enhance EAE with general SRL resources.

Notwithstanding the similarities, there are two main discrepancies between SRL and EAE structures that should be managed in order to facilitate transfer between them. The first is *label mismatch*. For example, ACE adopts role names with natural language words, such as BUYER and PLACE, whereas PropBank utilizes generalized labels like ARG0 and ARGM-LOC. PropBank also provides more specific role descriptions, but many are inconsistent and not well-formed for direct use as role names. Although FrameNet also adopts natural language role names, it is laborious and sometimes challenging to find all the direct mappings to the target event frames. Moreover, SRL resources do not typically annotate *distant arguments*, where there are no explicit syntactic encodings expressing the argument relation.[2] For example, in the sentence depicted in Figure 7.1, though it can be understood that the "store" is very likely to be the place where the "buying" happens, SRL annotations do not include this semantically inferred link, whereas it is considered an argument in event annotations.

Although there has been previous work utilizing SRL for argument linking (O'Gorman, 2019), it remains unclear how to best directly transfer from SRL to EAE, especially with recent pre-trained models. In this chapter, we provide a comprehensive investigation of the transfer from SRL to EAE. We view the tasks as a role querying problem within a *unified framework*, which covers various different argument extraction methods, including classification-based methods (Ouchi et al., 2018; Ebner et al., 2020), machine reading comprehension (MRC)-based methods (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020b; Feng et al., 2020; Lyu et al., 2021; Liu et al., 2021a) as well as sequence-to-sequence generation based ones (Li et al., 2021a; Hsu et al., 2022; Lu et al., 2021). We further explore a template-based slot querying strategy, by querying argument roles using contextualized representations of the corresponding role slots in the frame template. We tackle the label-mismatch problem by forming the queries in templated natural language, which allows for the same query representation to be shared across varied schemes. To mitigate the lack of distant argument annotations in SRL, we apply two argument augmentation techniques: Data augmentation by shuffling input texts, which reduces the model's reliance on local syntax, and knowledge distillation from question-answering (QA) data, which incorporates

[2]These are also known as *implicit arguments* (O'Gorman, 2019). While there are more fine-grained linguistic criteria, we take a simplified approximate approach by checking the syntactic distances between triggers and arguments.

distant argument signals.

With experiments on the standard ACE and ERE English event benchmarks, we show that SRL annotations are valuable resources for EAE. With the template-based querying strategy, a model trained with SRL can reach nearly 80% of the fully-supervised F1 score in the zero-shot scenario, and an intermediate-training scheme provides further benefits in the low-resource setting. The model also obtains promising results in extensions to cross-domain and multi-lingual scenarios, demonstrating its generalizability. Our results highlight the utility of SRL annotations in the context of downstream applications with limited direct annotations.

7.2 Method

7.2.1 Querying Methods

For either semantic roles or event arguments, we can view the extraction task as a role querying problem. Specifically, we are given a sequence of words $\mathbf{s} = \{w_1, ..., w_n\}$ as input contexts as well as a predicate or event trigger word w_e and the semantic frame or event type t. Each type is associated with a list of participating roles to be filled and the task is to extract arguments from the input contexts for each role. We adopt one specific modeling simplification, that is, our model only predicts the syntactic head word of an argument. For EAE, a heuristic method is further adopted to expand from head words to spans: We simply include the head word's child that is linked with an MWE dependency relation[3] and has an uppercase first letter. We find that this heuristic works well in practice, expanding correctly to 95% of the argument spans in the ACE and ERE event datasets. We take this approach to make it easier to transfer across different schemes, which may have different annotation criteria on argument spans.

In this way, we can view both SRL and EAE as role querying problems over the input words (all the queries depend on the predicates, which we assume are given and omit for brevity). Specifically, the probability of a candidate word w to be the argument filling a role r is:

$$p_r(w) = \frac{\exp(\lambda \mathbf{h}_w^T \mathbf{q}_r)}{\sum_{w' \in \mathbf{s} \cup \{\epsilon\}} \exp(\lambda \mathbf{h}_{w'}^T \mathbf{q}_r)}$$

Here, $\mathbf{h}_w$ denotes the representation vector of the word w, and $\mathbf{q}_r$ indicates the querying vector of the role r. We further include a scaling factor λ, which is fixed to $\frac{1}{\sqrt{d}}$, where d is the dimension of $\mathbf{h}$ and $\mathbf{q}$, following the attention calculation in Transformer (Vaswani et al., 2017). We specify a dummy token ϵ to handle the cases where no arguments can be found for a role. This modeling scheme is flexible and allows different argument extraction strategies to be viewed *in a unified way*. In this chapter, we explore four strategies, as illustrated in Figure 7.2.

1) CLF. We start with querying based on traditional classification, which assigns to each role a *non-contextualized* vector. To allow transfer to different role names, we initialize the role vectors with average-pooled representations obtained by passing the role names individually to a pre-trained language model. We call this strategy classification-based since the role vectors can

[3]Multi-word expressions: {"fixed", "flat", "compound"}.

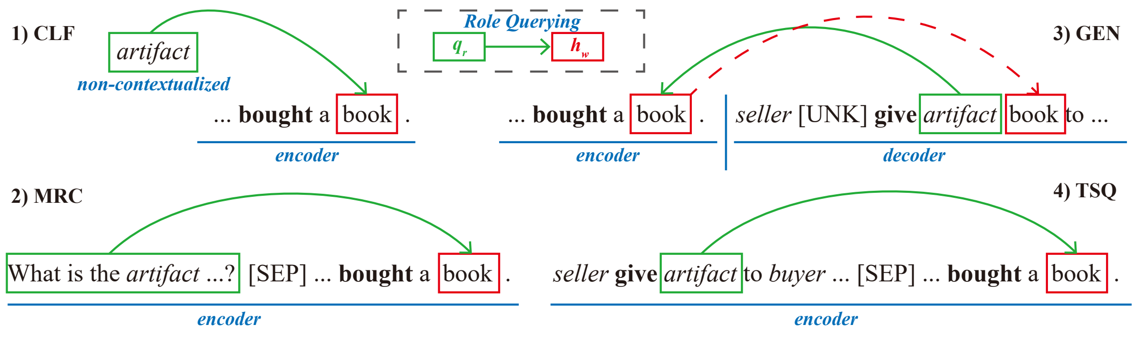

1) CLF
artifact
non-contextualized
... bought a book .
encoder
Role Querying
q_r
h_w
3) GEN
... bought a book .
encoder
seller [UNK] give artifact book to ...
decoder
2) MRC
What is the artifact ...? [SEP] ... bought a book .
encoder
4) TSQ
seller give artifact to buyer ... [SEP] ... bought a book .
encoder

Figure 7.2: Illustrations of different role querying strategies.

be viewed as weights in a linear classifier. This corresponds to more traditional argument extraction methods (Ouchi et al., 2018; Ebner et al., 2020). One shortcoming of this strategy is that the query vectors are constructed without access to input contexts, limiting their representation ability.

2) MRC. Recently, the strategy of casting NLP tasks as machine reading comprehension problems (Rajpurkar et al., 2016, 2018) has been applied to EAE (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020b; Feng et al., 2020; Lyu et al., 2021; Liu et al., 2021a). In this strategy, each role is queried with a *contextualized* question that is encoded together with the context. Unless otherwise specified, we form the role questions using the templates of Liu et al. (2021a), which can be automatically generated from the role names. Since each question queries only one role, this strategy requires a full pass through the encoder for each role, raising concerns regarding its computational efficiency, as compared to CLF.

3) GEN. More recently, many approaches extract arguments by sequence-to-sequence generation (Paolini et al., 2021; Li et al., 2021a; Hsu et al., 2022; Lu et al., 2021; Du et al., 2021b; Huang et al., 2022). Specifically, Li et al. (2021a) and Hsu et al. (2022) adopt a template-based generation strategy, which aggregates the queries of all roles for an event into one *template* sentence (or *bleached statement* (Chen et al., 2020)). This strategy is promising since the template can contain all roles and query them in one pass. Since arguments come from input contexts, we further adopt a pointer network (Vinyals et al., 2015a) for argument selection rather than generating through output vocabularies, fitting our unified querying framework. Because of the auto-regressive decoding scheme, this strategy can also suffer lower efficiency compared to CLF.

4) TSQ. We further explore a strategy that fully exploits the representational power and querying efficiency of templates. We do not fill the templates with actual words in the context but simply keep the role names as placeholders. We concatenate this template with the context, then pass the full sequence to the encoder for contextualization. Finally, the contextualized representations of the role slots in the template are adopted as role query vectors. We refer to this strategy as Template-based Slot Querying (TSQ). This approach is similar to the contemporaneous work of Ma et al. (2022). Our approach to template querying differs primarily in that: 1) We concatenate both the template and the context and feed them to the encoder, allowing for bidirectional modeling, and; 2) Our models predict argument head words rather than spans to facilitate the transfer since unlike Ma et al. (2022) our focus is transfer learning.

7.2.2 SRL Templates

We take PropBank[4] (Palmer et al., 2005), NomBank[5] (Meyers et al., 2004) and FrameNet[6] (Baker et al., 1998b) as our main SRL resources. Since many NomBank frames are derived from PropBank frames, we simply map them to the PropBank counterparts (by checking the "source" attribute in a NomBank frame) and ignore the ones that do not have such mappings. We filter event-related SRL frames by excluding the ones that do not have any verb realizations, which are judged by the POS sets provided in the frame files. Moreover, we only consider a subset of non-core or modifier roles that are related to the target EAE task, including ARGM-LOC in PropBank and {PLACE, INSTRUMENT, WEAPON, VEHICLE} in FrameNet.

To allow transfer across different schemes, we need to specify extra information required by the role querying strategies. In particular, templates are not included in SRL frame definitions and it is infeasible to manually specify them for hundreds to thousands of SRL frames. We adopt a semi-automatic method to construct the templates, with extra information collected from data statistics:

- **Role names.** We directly take the role label names of FrameNet since they are already in natural language forms. We further train[7] a role label classifier[8] with the FrameNet data and apply it to the PropBank data. Then for each frame-specific role, the most frequently predicted label will be its role name. For example, for the ARG0 role of the "buy.01" frame, its arguments in the dataset are mostly predicted to the BUYER label, which is thus assigned as its role name.
- **Role orders.** We construct a template for an SRL frame by concatenating its predicate word and role names. The main thing to specify is their ordering. We again take a statistical approach and collect each role's relative distance to the predicate. For example, in the "buy.01" frame instance of "He bought a book in a store.", ARG0 (He) gets a distance of -1, ARG1 (book) gets a +1 and ARGM-LOC (store) gets a +2. Finally, the role orders in the templates are decided by the roles' average relative distances. We aim to obtain a canonical verb-styled ordering in active voice, and thus we only consider frame instances that are realized by non-passive verbal predicates.
- **Preposition words.** When realized in natural language sentences, many roles are accompanied by prepositions. We count the frequency that a role is filled by an argument that utilizes a preposition[9] and keep the prepositions that appear more frequent than 25%. When there are

[4]https://github.com/propbank/propbank-frames/releases/tag/v3.1

[5]https://nlp.cs.nyu.edu/meyers/nombank/nombank.1.0/

[6]https://framenet.icsi.berkeley.edu/fndrupal/frameIndex

[7]Another option could be to use existing resources that connect PropBank and FrameNet, such as SemLink (Palmer, 2009; Stowe et al., 2021). Nevertheless, their coverage is still slightly lacking and we thus take a data-driven method, which could map every frame that has data.

[8]This classifier is similar to our CLF querying model except that no extraction is needed. Its accuracy on the FrameNet dev set is around 0.7. Notice that even when the classifier does not hit the most suitable label, the predicted ones may still be reasonable for our usage.

[9]The criterion is that the argument's head word has a dependency relation of "case" to a child whose POS is

Scheme	Frame	Template
PropBank	forbid.01	authority **forbid** protagonist action in place
	rent.01	at lessor lessee **rent** goods from lessor for money in place
	swim.01	from area self mover **swim** against area to goal in place
FrameNet	Abandonment	agent **abandon** theme in place
	Employing	employer **employ** employee field position task in place
	Mention	communicator **mention** specified content message in medium in place

Table 7.1: Examples of the semi-automatically generated templates.

such prepositions, we add the preposition before the role name and put them together into the slot. When there are multiple feasible prepositions, we randomly sample one in training and utilize the most frequent one in testing.

With these three types of extra information, we construct the templates by concatenating all the corresponding ordered pieces. For example, the "buy.01" PropBank frame gets a template of "buyer **buy** goods for recipient from seller for money in place". Most of the above heuristics are decided by manually checking the generated outputs for the PropBank and FrameNet frames.

Notice that this semi-automatic approach is far from perfect and there can be noises and inconsistencies, as shown in some of the examples in Table 7.1. Nevertheless, the above three pieces provide complementary information for role specification: the role names provide semantic information, the role orders include syntactic word order information, and the prepositions give further hints. In practice, we find that most of the generated templates are reasonably close to natural language. In this way, we are able to form similar queries for both SRL and EAE, tackling the label mismatch problem between different frame schemes.

7.2.3 Argument Augmentation

In addition to label mismatch, another discrepancy between SRL and EAE is that arguments in traditional SRL are syntactically constrained whereas event arguments can be extracted from any place in the context. Therefore, SRL models will have difficulties in predicting syntactically distant arguments. To mitigate this problem, we apply data augmentation (Feng et al., 2021) and knowledge distillation (Hinton et al., 2015) to augment distant arguments for SRL instances.

Firstly, we apply a simple data augmentation method by shuffling the input contexts. Since the SRL arguments are constrained by syntax, we hypothesize that by distorting syntax in some way, the model can be trained to focus more on the semantic relations between the predicates and arguments. This may allow it to predict more distant arguments. To distort syntax, we randomly chunk the input sentence with sizes randomly chosen from one to three at each time. Then these text chunks are shuffled, re-concatenated, and fed to the pre-trained model for contextualized encoding. Since our model only selects argument head words, there is no change to the later processing except for word position re-indexing. We only apply this procedure during training and simply mix vanilla unshuffled data with the shuffled ones by a 1:1 ratio.

"ADP".

Moreover, we seek signals of distant arguments from question answering (QA)[10] datasets, such as SQuAD (Rajpurkar et al., 2016, 2018). In QA annotations, the answers are not constrained by syntax and can be freely picked from the full context, providing valuable resources for distant arguments (Liu et al., 2021a). Motivated by this, we train a QA model with the MRC strategy and predict the missing arguments for SRL instances. Instead of hard predictions, we store a soft probabilistic distribution over the context words for each role and utilize these for SRL training with a standard cross-entropy objective:

$$\mathcal{L}_{\text{distill}}(r) = - \sum_{w \in \mathbf{s} \cup \{\epsilon\}} p_r^{\text{qa}}(w) \log p_r^{\text{m}}(w)$$

Here, for the querying of each role r, $p_r^{\text{qa}}(w)$ denotes the argument probabilities among the context words according to the QA model, while $p_r^{\text{m}}(w)$ indicates the current model's outputs. To avoid noise from the QA predictions, we adopt two filters. Firstly, we only apply distillation for the unfilled roles according to SRL annotations. This is intuitive since the filled roles already have gold annotations. Moreover, we apply distillation only when the prediction is confident enough. We perform calibration to the QA model by temperature scaling (Guo et al., 2017) and adopt a probability threshold of 0.5. In this way, we could borrow the signals of distant arguments from the QA datasets to enhance SRL instances with potential missing distant arguments.

7.3 Experiments

7.3.1 Settings

We conduct our main experiments with English ACE[11] (Walker et al., 2006) and ERE (LDC, 2015) event datasets. We adopt the preprocessing scripts[12] from OneIE (Lin et al., 2020). For the target event frames, we manually specify extra information such as templates, adopting those of Li et al. (2021a). Unless otherwise specified, we assume that gold event triggers are given and focus on the extraction of event arguments. We evaluate arguments by labeled F1 scores, which require both argument spans and roles to match the gold ones. We run with five random seeds and report average results.

For external data, we take PropBank, NomBank 1.0, and FrameNet 1.7 as our main SRL resources. We prepare the SRL templates by the semi-automatic process described in §7.2.2. For QA datasets, we take SQuAD 2.0 (Rajpurkar et al., 2018), QA-SRL 2.1 (FitzGerald et al., 2018), QANom (Klein et al., 2020) and QAMR (Michael et al., 2018). For the training of SRL or QA models, we simply adopt the concatenation of all the corresponding datasets. Except for those that have manual syntactic annotations, we utilize Stanza (Qi et al., 2020) to parse the texts to obtain the syntactic head words of the arguments.

We adopt pre-trained language models for initialization and fine-tune the full models during training. Specifically, we use RoBERTa$_{\text{base}}$ (Liu et al., 2019) for encoder-only models (CLF,

[10]Specifically we adopt the extractive QA-MRC data. To avoid confusion, we use "QA" when denoting data resources while using "MRC" for the querying strategy.

[11]We adopt ACE05-E^{+} (Lin et al., 2020).

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

MRC, TSQ) and BART$_{\text{base}}$ (Lewis et al., 2020a) for encoder-decoder models (GEN).

7.3.2 Main Transfer Experiments

We conduct our main experiments with English ACE and ERE datasets. Thanks to the unified querying framework, we can conduct experiments in a zero-shot setting (§7.3.2), where models trained on external data are directly evaluated on EAE. We also investigate low-resource settings where some amounts of EAE annotations are available for further fine-tuning (§7.3.2).

Zero-shot

In the zero-shot setting, we further compare with two methods in addition to SRL: 1) GPT-3 (Brown et al., 2020), where we form prompts for each role and use GPT-3 to generate the answers; 2) QA, where we train QA models[13] with the QA datasets. We also provide the fully-supervised results (Super.) as references.

Method	ACE			ERE		
	P%	R%	F1%	P%	R%	F1%
Super.	$68.93_{\pm1.07}$	$68.94_{\pm0.95}$	$68.93_{\pm0.95}$	$72.75_{\pm1.69}$	$71.80_{\pm1.29}$	$72.24_{\pm0.34}$
GPT-3	29.10	34.25	31.47	25.09	26.76	25.90
QA	$32.77_{\pm3.70}$	$47.43_{\pm1.17}$	$38.62_{\pm2.58}$	$32.68_{\pm2.78}$	$48.13_{\pm4.08}$	$38.74_{\pm2.09}$
SRL$_{\text{CLF}}$	$47.97_{\pm1.47}$	$25.37_{\pm0.86}$	$33.18_{\pm0.92}$	$50.17_{\pm1.72}$	$25.60_{\pm0.65}$	$33.89_{\pm0.88}$
SRL$_{\text{MRC}}$	$58.27_{\pm0.75}$	$39.54_{\pm1.60}$	$47.08_{\pm0.89}$	$62.02_{\pm1.15}$	$45.31_{\pm1.74}$	$52.32_{\pm0.83}$
SRL$_{\text{GEN}}$	$55.77_{\pm0.61}$	$45.31_{\pm1.26}$	$49.99_{\pm0.93}$	$58.37_{\pm0.66}$	$52.68_{\pm0.63}$	$55.38_{\pm0.62}$
SRL$_{\text{TSQ}}$	$57.74_{\pm0.95}$	$49.61_{\pm0.80}$	$53.36_{\pm0.53}$	$59.93_{\pm0.68}$	$55.84_{\pm0.78}$	$57.81_{\pm0.34}$
+shuf.	$58.36_{\pm0.53}$	$51.70_{\pm0.52}$	$54.82_{\pm0.44}$	$59.70_{\pm0.89}$	$57.42_{\pm1.26}$	$58.54_{\pm1.05}$
+distill	$54.53_{\pm0.97}$	$55.85_{\pm0.67}$	$55.17_{\pm0.42}$	$55.27_{\pm0.72}$	$60.90_{\pm0.85}$	$57.95_{\pm0.65}$
+both	$55.68_{\pm1.26}$	$57.04_{\pm0.93}$	**56.35**$_{\pm1.07}$	$56.63_{\pm0.78}$	$61.48_{\pm0.18}$	**58.96**$_{\pm0.48}$

Table 7.2: Zero-shot EAE results on event test sets.

Results The main results are shown in Table 7.2. Except for the one with the CLF strategy, SRL models perform generally better than QA and GPT-3, showing the effectiveness of utilizing SRL resources. Among the SRL models, the TSQ strategy generally performs the best, indicating the effectiveness of this contextualized querying strategy. Further improvements can be obtained with the argument augmentation techniques. Interestingly, if only using shuffling augmentation (+shuf.), precision remains roughly the same while recall increases. If only using distillation (+distill), recall increases but at the expense of precision. Finally, if both are utilized (+both), precision and recall both improve relative to the distillation-only case. This leads to the overall best F1 scores, reaching around 80% of the supervised results.

[13]Notice that we can only use the MRC strategy for QA models because of the task-specific format.

Distance Analysis As shown in Figure 7.3, we further perform breakdowns on the syntactic distances between triggers and arguments. We especially compare the QA model and the four SRL_{TSQ} models. Firstly, the QA model performs worse than SRL models except for the long distant ones ($d \geq 4$). This is due to SRL annotations mainly capturing syntactically local arguments while QA is not constrained by this. Within the SRL models, when adding shuffling (+shuf.) or distillation (+distill), the middle-ranged arguments consistently obtain improvements. One interesting pattern is that shuffling benefits $d = 1$ but hurts $d \geq 4$, while distillation seems to have the opposite effects. This may indicate that shuffling enhances more robust predictions of short- and middle-ranged arguments while distillation encourages longer-ranged ones. Finally, when combining these two techniques (+both), the model can reach a good balance, achieving the best overall results. Due to its overall better performance, we use the "SRL_{TSQ}+both" strategy for our SRL models in the remainder of this chapter.

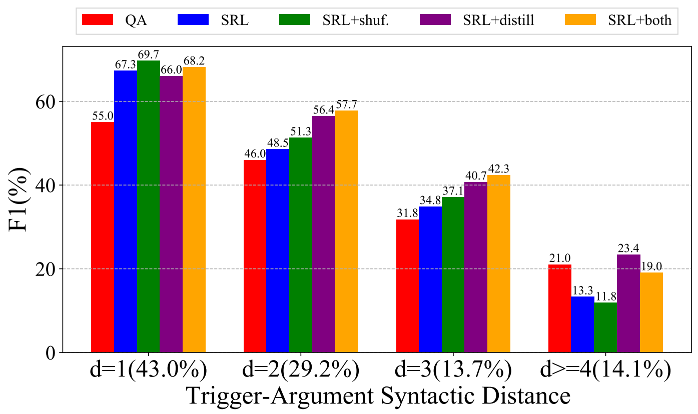


Figure 7.3: Breakdowns on trigger-argument syntactic distances.

Manual Analysis We further perform a manual error analysis to investigate what the main error types are. We randomly take 100 event frames that contain prediction errors from the ACE development set and categorize the errors. We perform this analysis for both our best zero-shot SRL model and the supervised model to examine where the main gaps are. We specify eight error categories:

- **Ambiguous** cases, where there are annotation errors or ambiguities, and the predictions could be regarded as correct in some way.
- **Coreference**, where predicted and gold arguments are co-referenced in some way.
- **Span** mismatch, where the main contents are captured with non-crucial boundary mismatches.
- **Head** mismatch, where the main contents are roughly captured but not with the exact annotated words. This happens mostly in appositions or noun modifiers with more specific content.
- **Role** misunderstanding, where the semantic meaning of a role is not correctly understood.
- **Local** inference, where correct predictions require semantic inference at the local clause.
- **Global** understanding, where correct predictions require a global understanding of the full context.
- **Others**, where the error does not fall into any of the above categories.

Category	Example	SRL	Super.
Correct	—	194 (50.52%)	238 (63.47%)
Role	Actually, they **paid**$_{\text{TransferMoney}}$ for [it]$_{\text{Beneficiary}}$.	45 (11.72%)	22 (5.87%)
Local	My$_{\text{Buyer}}$ plan is to **pay**$_{\text{TransferOwnership}}$ off my car.	40 (10.42%)	23 (6.13%)
Head	They **fired**$_{\text{Attack}}$ mortars in the [direction]$_{\text{Target}}$ of the 7th Cavalry$_{\text{Target}}$.	24 (6.25%)	8 (2.13%)
Global	"We condemned the **attack**$_{\text{Attack}}$," he said, adding that his messages to the terrorists$_{\text{Attacker}}$ is: Their efforts will not be successful.	24 (6.25%)	17 (4.53%)
Others	—	14 (3.65%)	10 (2.67%)
Ambiguous	At least four [policeman]$_{\text{Attacker}}$$_{\text{Victim}}$ were injured in **clashes**$_{\text{Attack}}$.	18 (4.69%)	20 (5.33%)
Span	The 1st [Brigade]$_{\text{Attacker}}$$_{\text{Attacker}}$ took Karbala with a minimal **fight**$_{\text{Attack}}$.	12 (3.12%)	14 (3.73%)
Coreference	He$_{\text{Defendant}}$ skipped bail during [his]$_{\text{Defendant}}$ **trial**$_{\text{Hearing}}$.	13 (3.39%)	23 (6.13%)

Table 7.3: Examples of the categories and results of the manual error analysis.

Examples of these categories and the results are shown in Table 7.3. According to the statistics, the main gaps between the SRL and supervised models are in the categories of role misunderstanding, lacking semantic inference as well as head mismatches. Head mismatches are due to the discrepancies between syntactic head and semantic core words, and might not cause severe problems. The first two are more semantic errors that are related to the essence of the EAE task. Role misunderstanding may be related to template mismatches, where roles in the SRL templates are different than those in target event ones. Lacking semantic inference is mostly upon distant arguments. Though the argument augmentation techniques recover certain distant arguments for SRL frames, this problem is still far from being solved. Notice that these semantic errors reveal the main difficulties of the EAE task, which even supervised systems have not yet fully tackled. To solve these problems, a more comprehensive semantic understanding is required.

Low-resource

We further investigate scenarios where we have some amount of target EAE annotations. With target data, we can directly train an EAE model (from pre-trained language models). We further apply a simple intermediate-training scheme (Phang et al., 2018; Wang et al., 2019a) to transfer the knowledge from SRL. We take the SRL-trained model and further fine-tune it on the target event data. A similar scheme can also be adopted with the QA model. Figure 7.4 shows the results with different amounts of training instances. Generally, SRL intermediate training is beneficial, especially for middle- and low-resource cases, again showing that SRL annotations

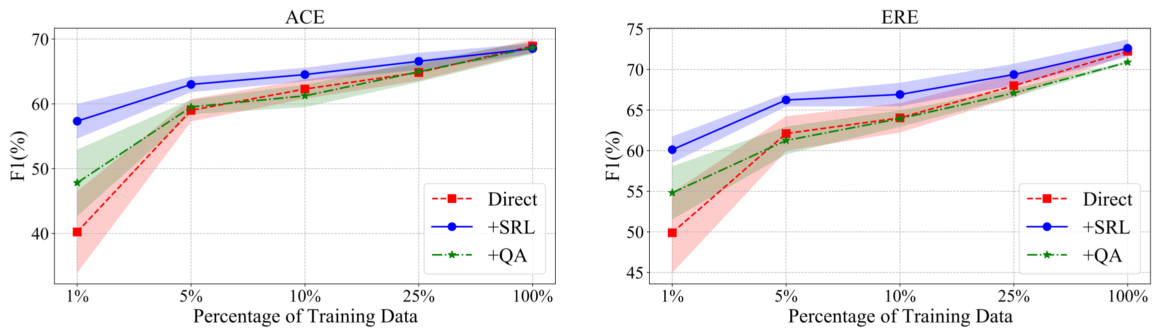


Figure 7.4: Model performance with direct or intermediate training.

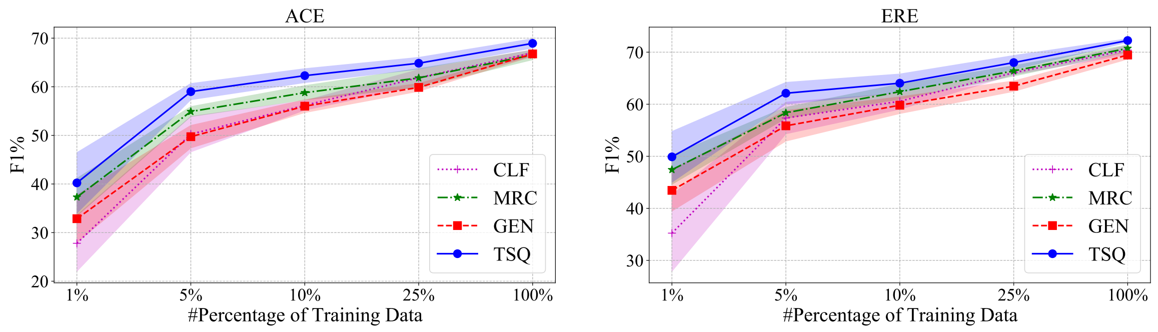


Figure 7.5: Argument F1(%) scores on ACE and ERE test sets with different amounts of training data.

can be valuable transfer sources for the extraction of event arguments. Note that when using full target data, the external SRL data is less helpful. We think this is probably because there is already enough supervision to learn most of the target patterns, and there might be less further information that SRL could provide beyond the already rich target resources.

Supervised Results

Although our main focus is on the transfer scenarios, we also conduct purely supervised experiments on the target EAE datasets. We compare the four querying strategies with different amounts of training data. The results are shown in Figure 7.5. The overall trend is similar in both datasets. In high-resource scenarios, different querying strategies could obtain similar results. In low-resource cases, the methods that capture more contextual information in the queries can generally perform better. The CLF strategy with non-contextualized queries obtains worse results than the others, while TSQ is the overall best-performing strategy. This is also consistent with the results in the zero-shot transfer scenarios.

Speed Comparisons

We also perform decoding speed comparisons to examine the efficiency of different querying strategies. The results are shown in Table 7.4. There is no surprise that the simplest CLF strategy achieves the highest decoding speed since its input sequences are the shortest and there is no further complex query encoding. TSQ is only around 10% slower, but still efficient compared with the other two methods, where MRC suffers from multiple forwarding for different role queries and GEN requires auto-regressive decoding at testing time.

Method	Single-instance	Batched
CLF	**184**	**316**
MRC	106	146
GEN	28	144
TSQ	167	281

Table 7.4: Decoding speed (instances per second) comparisons of different querying strategies.

Comparisons with ChatGPT

We also perform comparisons with the latest large language models, which have shown impressive capabilities of text understanding and generation. Specifically, we take GPT-3.5-TURBO as our studying target and compare it with our SRL model. We randomly sample 100 event frames from the ACE05 development set and reuse our previous role questions to prompt the ChatGPT model. Instead of only relying on automatic evaluation metrics based on strict matching, we conduct manual analysis to better understand the real performance of the models.

For the manual analysis, we adopt an error categorization that is similar to those in Table 7.3, but with certain adjustments in accordance with ChatGPT's characteristics. Table 7.5 lists the categories utilized in this experiment. More specifically, we have:

- **Ambiguous** cases, where there are annotation errors or ambiguities, and the predictions could be regarded as correct in some way.
- **Format** problems, where ChatGPT's generated answers can be regarded as correct, but do not exactly match the original text.
- **Coreference**, where predicted and gold arguments are co-referenced in some way. This category also includes the cases where the correct argument is a pronoun, which ChatGPT usually does not extract.
- **Span** mismatch, where the main contents are captured with non-crucial boundary mismatches. This category also includes head mismatch, where the main contents are roughly captured but not with the exact annotated words.
- **Role** misunderstanding, where the semantic meaning of a role is not correctly understood.
- **Others**, where the error does not fall into any of the above categories, indicating that the model probably needs a better contextual understanding of the inputs.

Notice that the first four categories are minor errors, which should not be penalized as the more crucial errors in the last two categories. For instance, in the "Format" error example, the LM generates "13 people" instead of "13", which leads to a failure in text matching. But the answer is correct since it can be inferred that in the original text, "13" indicates "13 people".

The detailed results of our manual analysis are shown in Table 7.6. Here, we compare our SRL-based model against the ChatGPT model (GPT-3.5-TURBO). Here, we order the error cat-

Category	Explanation	Example
Ambiguous	Ambiguous cases or annotation errors.	At least four [policeman]$_{\text{Attacker}\,\text{Victim}}$ were injured in **clashes**$_{\text{Attack}}$.
Format	Generation is correct but does not match the original text.	The attack killed 5 people and **injured**$_{\text{Injury}}$ 13$_{\text{Victim}}$. (Generation = [13 people]$_{\text{Victim}}$)
Coref.	Mismtaches related to pronouns or coreference.	He$_{\text{Defendant}}$ skipped bail during [his]$_{\text{Defendant}}$ **trial**$_{\text{Hearing}}$.
Span	Span boundary mismatches or head word errors.	The group has **pushed onward**$_{\text{Transport}}$ in the general [direction]$_{\text{Destination}}$ of the city$_{\text{Destination}}$.
Role	Role misunderstanding.	The [group]$_{\text{Artifact}\,\text{Agent}}$ has **pushed onward**$_{\text{Transport}}$ in the general direction of the city.
Others	Other errors that require contextual understanding.	My$_{\text{Buyer}}$ plan is to **pay**$_{\text{TransferOwnership}}$ off my car.

Table 7.5: Error categories and examples for the ChatGPT error analysis.

Categories	SRL		ChatGPT	
	Count	F1(%)	Count	F1(%)
Exact	200	57.47	102	37.64
Ambiguous	14	61.94	13	42.33
Format	0	61.94	22	53.70
Coref.	8	63.79	30	62.53
Span	37	74.29	20	70.25
Role	30	82.88	31	82.10
Others	59	100.00	53	100.00

Table 7.6: Error analysis results for the comparison between SRL- and ChatGPT-based models.

egories according to their severity and list error counts. We further include an F1 column, which indicates the F1 score if we relax our evaluation by viewing the instances in the corresponding error category (as well as the less severe ones) as correct. First, if only calculating by exact matches (both argument span and role predictions should be perfectly matched to the gold ones), ChatGPT performs much worse than the SRL model. Nevertheless, if further checking the specific failure cases, we can see that many errors made by ChatGPT are caused by formatting problems (correct answers not exactly matched to the original texts) and coreference/pronoun problems (no extraction of pronouns or extracting coreferenced concrete mentions). If disregarding these minor problems (many of them are actually not errors), ChatGPT can obtain very similar results to the SRL model. This indicates that we need better prompting, extraction, and evaluation methods for utilizing LLMs for this task.

Types	QA	SRL	SRL$_{\text{+self}}$	Super.
Expression	70.71	76.66	77.59	80.90
Transcription	63.64	55.63	59.72	69.46
Catabolism	62.07	66.67	66.67	74.07
Phosphorylation	75.95	78.98	83.64	89.52
Localization	53.28	66.89	67.33	69.51
– **Simple** –	68.26	73.63	75.27	79.31
Binding	39.41	34.90	35.10	50.19
Regulation	33.80	38.95	38.52	45.90
Pos. regulation	31.85	38.96	39.95	49.41
Neg. regulation	36.62	44.84	44.51	47.17
– **Complex** –	33.36	40.44	40.88	48.32
– **All** –	47.42	51.95	52.76	60.22

Table 7.7: BioNLP-11 event extraction results (F1%).

7.3.3 Further Extensions

In the previous experiments, we take ACE and ERE as the targets, which are still relatively similar to the SRL annotations. In this sub-section, we further investigate scenarios where there are larger discrepancies between the source and the target. Specifically, we examine the transfer from SRL to EAE in cross-domain (§7.3.3), multi-lingual (§7.3.3), and multi-sentence (§7.3.3) cases.

Cross-domain

We first investigate the biomedical domain, utilizing the GENIA BioNLP-11 benchmark (Kim et al., 2011). The GENIA events are quite different than general SRL frames and mainly describe detailed bio-molecule behavior (Kim et al., 2008). Still focusing on the argument extraction step, we take the event triggers predicted by the supervised system BEESL (Ramponi et al., 2020). We perform zero-shot argument extraction and evaluate the QA and SRL models, with manually compiled role questions and templates. We adopt the official evaluation metric of approximate recursive matching.

Our main comparison is between the QA and SRL models, while we also include the supervised results of BEESL as references. We further adopt a self-training approach to adapt to the target domain. Specifically, we take the texts from the original GENIA training set, ignore the original labels, predict SRL frames on these texts with our SRL model, and train a final SRL model with both these predicted structures and external SRL resources.

The results of the test set are shown in Table 7.7. SRL generally outperforms QA for most of the types. This may be due to the difficulty of asking proper questions. For example, for the "Regulation" event, we ask "What is regulated?" for the role of "Theme" and "What causes the regulation?" for "Cause". These questions may be unrelated to the actual contexts, while for the SRL models, extra hints from the query templates may be helpful. This may also explain why

QA is better on some of the types where it is relatively easy to ask questions. For example, for "Transcription", the question "What is transcribed?" would be accurate for most contexts. For the SRL models, the self-training method is beneficial overall, showing the effectiveness of utilizing unlabeled corpus from the target domain.[14] Finally, our best zero-shot model could recover more than 80% of the overall performance of the supervised model, showing that general SRL resources can still be helpful in the biomedical domain. The main gaps between the zero-shot and supervised systems are in the "Binding" and "Complex" events where there are complicated and even nested structures. One future direction is to investigate ways to better handle these complex structures.

Multi-lingual

We next explore a multi-lingual setting, taking ACE05 Arabic and Chinese datasets as our targets. We follow Huang et al. (2022) and utilize their pre-precessing scripts[15] for data preparation[16]. We further include multi-lingual external resources. For SRL, we utilize Arabic and Chinese PropBank annotations from OntoNotes (Hovy et al., 2006; Weischedel et al., 2013). For the role names in SRL frames, we again adopt a statistical approach: predicting with a FrameNet classifier based on a multilingual pre-trained encoder and adopting the mostly predicted label for each role. Due to differences in word order and usage of prepositional words in non-English languages, we exclude preposition words and simply order the roles by their ARG numbers.[17] We also include QA datasets for the target languages, adopting CMRC-2018 (Cui et al., 2019) for Chinese and the Arabic portion of TyDiQA (Clark et al., 2020) for Arabic. All our models in this experiment are based on the pre-trained XLM-R$_{\text{base}}$ (Conneau et al., 2020).

The results are shown in Table 7.8. In the first group, we compare zero-shot performance without any EAE training resources. Similar to the previous trends, SRL models are better than QA models, while including annotations in the target language could provide further benefits. In the second group, we assume access to English EAE training data. Similar to §7.3.2, we adopt an intermediate-training scheme by further fine-tuning the QA or SRL model on the English EAE data. Compared with the results of direct training in English, intermediate training with external resources could bring improvements. Again we see that models enhanced with SRL resources obtain the overall best results, which are quite promising when compared with the supervised ones.

One interesting aspect of the multi-lingual scenario is how the predictions are influenced by the word order difference between the source and target languages. We analyze the influence by measuring the performance differences in different roles. We first calculate the directional statistics for each role in each language, specifically: for a role in a language, what percentage of its arguments appear after the trigger? For example, "Attacker" appears after the trigger 26.9%

[14]We also tried a masked-language-model objective but did not find obvious improvements.

[15]`https://github.com/PlusLabNLP/X-Gear`

[16]We further re-tokenize Chinese data with CoreNLP (Manning et al., 2014) to align with segmentation in OntoNotes.

[17]The Arabic and Chinese frames adopt similar schemes as in English, specifying roles of {ARG0, ARG1, ...}. We find it reasonable by simply ordering them by the role numbers and forming templates of "ARG0 V ARG1 ARG2 ...".

Model	Arabic	Chinese
Zero-shot results without any EAE annotations.		
QA_{en}	$22.56_{\pm 1.48}$	$26.58_{\pm 2.61}$
$\text{QA}_{\text{en+tgt}}$	$23.54_{\pm 1.43}$	$27.08_{\pm 1.79}$
SRL_{en}	$37.75_{\pm 0.52}$	$39.37_{\pm 1.45}$
$\text{SRL}_{\text{en+tgt}}$	$\mathbf{40.64}_{\pm 1.49}$	$\mathbf{41.50}_{\pm 1.04}$
Multi-lingual results with English EAE annotations.		
$\text{GATE}^{\dagger}$	44.5	49.2
$\text{X-Gear}^{\dagger}$	44.8	54.0
En_{MRC}	$37.44_{\pm 3.02}$	$51.86_{\pm 0.92}$
$+\text{QA}_{\text{en}}$	$39.06_{\pm 2.86}$	$53.36_{\pm 1.06}$
$+\text{QA}_{\text{en+tgt}}$	$44.27_{\pm 1.37}$	$53.97_{\pm 1.41}$
En_{TSQ}	$37.64_{\pm 1.96}$	$53.54_{\pm 0.65}$
$+\text{SRL}_{\text{en}}$	$41.86_{\pm 0.92}$	$53.96_{\pm 0.85}$
$+\text{SRL}_{\text{en+tgt}}$	$\mathbf{51.51}_{\pm 1.32}$	$\mathbf{58.90}_{\pm 0.76}$
Supervised results with target EAE annotations.		
Super.	$58.09_{\pm 1.51}$	$65.11_{\pm 0.94}$

Table 7.8: Results (F1%) of ACE05 Arabic and Chinese.

of the time in English, while this percentage is 72.7% in Arabic. Then for each role, we have a source-target order difference metric, which is the absolute value of the frequency difference. We further calculate the performance differences between a transfer model trained with English data and a supervised model directly trained on the target language. Finally, we measure the correlation between the order differences and performance differences for the top-ten frequent roles in each language. The results for the transfer model with or without (multi-lingual) SRL intermediate training are shown in Table 7.9. Interestingly, if directly transferring from English to the target languages, there are at least moderate correlations between the order differences and performance gaps. While using SRL, the correlations decrease probably because of the extra signals about the target language order in the SRL data. This shows that order differences may be a major factor influencing the effectiveness of the cross-lingual transfer. Currently, our templates are all English-styled and it would be an interesting future direction to explore the influences of template specifications such as role orders.

Multi-sentence

Finally, we investigate multi-sentence event arguments, which are not constrained to the same sentence of the event trigger but can come from the document-level contexts. To investigate this phenomenon, we evaluate[18] on the RAMS dataset (Ebner et al., 2020), which annotates event

[18]Since our head-expanding heuristic does not cover the argument span annotation conventions of RAMS, for simplicity we only evaluate argument head words.

Language	Model	Pearson	Spearman
Arabic	w/o SRL	0.6050	0.6727
	w/ SRL	0.5157	0.1394
Chinese	w/o SRL	0.6910	0.5636
	w/ SRL	0.5025	0.2727

Table 7.9: Correlations between relative role order differences and performance gaps to supervised systems for multi-lingual EAE.

Model	Overall	Same-Sent.	Cross-Sent.
QA	$28.23_{\pm 0.74}$	$35.16_{\pm 1.42}$	$11.66_{\pm 0.69}$
SRL	$48.03_{\pm 0.30}$	$53.36_{\pm 0.30}$	$2.81_{\pm 0.78}$
SRL+pseudo	$48.00_{\pm 0.14}$	$53.50_{\pm 0.16}$	$11.17_{\pm 1.88}$
Super.	$57.38_{\pm 0.84}$	$63.45_{\pm 0.86}$	$25.52_{\pm 1.31}$

Table 7.10: Argument head F1(%) on RAMS test set.

arguments within five-sentence windows around the triggers. We similarly extend contexts to five-sentence windows if available in our training of QA and SRL models for this experiment.

The zero-shot results are shown in the first group of Table 7.10. Consistent with our previous findings, SRL performs better than QA for same-sentence arguments. Nevertheless, it predicts very few cross-sentence arguments. This is not surprising because there are no such signals in the SRL training data. Inspired by previous work on coreference and anaphora resolution (Varkel and Globerson, 2020; Konno et al., 2021), we create pseudo SRL data with cross-sentence arguments by surface-string matching. Specifically, for each nominal argument in an SRL instance, we search for words in nearby sentences that have the same lemma as the argument's head word. If there are, we delete the original true argument and add pseudo cross-sentence argument links to those matched words. Although deletion may create ungrammatical instances, we find it better than other schemes; such as replacing the original argument with a "[MASK]" token. With the additional synthetic data, the model can recover certain cross-sentence arguments while keeping similar same-sentence performance. Multi-sentence argument extraction is still a difficult task, where even the supervised system can only obtain an F1 score of around 25%. This calls for further exploration, and an investigation of how best to use auxiliary data (such as from SRL) may be a promising direction.

7.4 Related Work

Utilizing shallow semantics for event-centric information extraction tasks has been explored previously. Liu et al. (2016) leverage FrameNet frames to enhance event detection. Wang et al. (2021d) conduct contrastive pre-training with AMR structures to enhance event extraction. Previous work utilizes predicted shallow semantic structures as inputs to help low-resource event extraction (Peng et al., 2016; Huang et al., 2018; Lyu et al., 2021) and event schema induction (Huang et al., 2016b). Moreover, SRL has been utilized for implicit argument linking or implicit semantic role labeling (iSRL) in previous work (Chen et al., 2010; Laparra and Rigau, 2012, 2013; Feizabadi and Padó, 2015; O'Gorman, 2019). this chapter follows these directions and shows that SRL can be a valuable direct training resource for EAE.

For the EAE task, most previous work adopts a classification-based strategy where each role is assigned static querying parameters (Chen et al., 2015a; Nguyen et al., 2016; Wang et al., 2019b; Pouran Ben Veyseh et al., 2020; Ma et al., 2020; Ebner et al., 2020). Recently, two interesting alternative strategies have been explored to enable extraction in more flexible ways: MRC-based methods cast the problem as answering role questions (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020b; Feng et al., 2020; Lyu et al., 2021; Liu et al., 2021a), while generation-based methods adopt sequence-to-sequence generation schemes (Paolini et al., 2021; Li et al., 2021a; Hsu et al., 2022; Lu et al., 2021; Du et al., 2021b; Huang et al., 2022). We cover all these strategies within a unified role querying framework and further explore a template-based role querying strategy. This strategy is also related to prompt-based learning (Liu et al., 2021b; Schick and Schütze, 2021; Li and Liang, 2021; Petroni et al., 2019), but differs in the extraction-targeted paradigm. Concurrently, Ma et al. (2022) adopt a similar idea, while this chapter differs mainly in our focus on transfer learning.

7.5 Conclusion

In this chapter, we explore transfer learning from semantic roles to event arguments. With unified role querying strategies, we show that SRL annotations are a valuable resource for event argument extraction. The SRL model also obtains promising results when extended to new scenarios with domain and language differences.

Part III

Active Learning

Chapter 8

A Survey of Active Learning for Natural Language Processing

In previous chapters, we examine cases where the model is *passively* given a fixed training dataset, either with direct target annotations or from related corpora. Many times, although we might not be able to annotate large amounts of data for the target task, it is usually feasible to create a small amount of annotation. In this case, how to select data instances to annotate would be influential on the learning effectiveness. The idea of *active learning* is motivated in this way and aims to learn with fewer labeled data by allowing the learner to *actively* select data instances. In this chapter, we will start our investigation with a literature review of active learning, with a specific focus on its application to NLP.

8.1 Introduction

The majority of modern natural language processing (NLP) systems are based on data-driven machine learning models. The success of these models depends on the quality and quantity of the available target training data. While these models can obtain impressive performance if given enough supervision, it is usually expensive to collect large amounts of annotations, especially considering that the labeling process can be laborious and challenging for NLP tasks (§8.3.2). *Active learning* (AL), an approach that aims to achieve high accuracy with fewer training labels by allowing a model to choose the data to be annotated and used for learning, is a widely-studied approach to tackle this labeling bottleneck (Settles, 2009).

Active learning has been studied for more than twenty years (Lewis and Gale, 1994; Lewis and Catlett, 1994; Cohn et al., 1994, 1996) and there have been several literature surveys on this topic (Settles, 2009; Olsson, 2009; Fu et al., 2013; Aggarwal et al., 2014; Hino, 2020; Schröder and Niekler, 2020; Ren et al., 2021; Zhan et al., 2022). Nevertheless, there is still a lack of an AL survey for NLP that includes recent advances. Settles (2009) and Olsson (2009) provide great surveys covering AL for NLP, but these surveys are now more than a decade old. In the meantime, the field of NLP has been transformed by deep learning. While other more recent surveys cover deep active learning, they are either too specific, focused only on text classification (Schröder and Niekler, 2020), or too general, covering AI applications more broadly (Ren et al.,

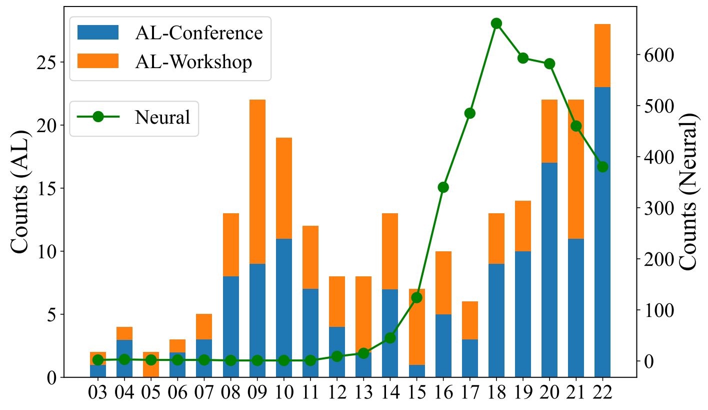


Figure 8.1: Counts of AL (left) and *"neural"* (right) papers in the ACL Anthology over the past twenty years.

2021; Zhan et al., 2022). Moreover, applying AL to NLP tasks requires specific considerations, e.g. handling complex output structures and trade-offs in text annotation cost (§8.3), which have not been thoroughly discussed.

In order to provide an NLP-specific AL survey,[1] we start by searching the ACL Anthology for AL-related papers. We simply search for the keyword *"active"* in paper titles and then perform manual filtering. We also gradually include relevant papers missed by keyword searches and papers from other venues encountered by following reference links throughout the surveying process. The distribution of AL-related papers in the ACL Anthology over the past twenty years is shown in Figure 8.1, which also includes rough counts of papers concerning neural models by searching for the word *"neural"* in titles. The overall trend is interesting. There is a peak around the years 2009 and 2010, while the counts drop and fluctuate during the mid-2010s, which corresponds to the time when neural models became prominent in NLP. We observe a renewed interest in AL research in recent years, which is primarily focused on deep active learning (Ren et al., 2021; Zhan et al., 2022).

8.1.1 Overview

We mainly examine the widely utilized pool-based scenario (Lewis and Gale, 1994), where a pool of unlabeled data is available and instances are drawn from the pool to be annotated. Algorithm 1 illustrates a typical AL procedure, which consists of a loop of instance selection with the current model and model training with updated annotations. The remainder of this survey is organized corresponding to the main steps in this procedure:

- In §8.2, we discuss the core aspect of AL: Query strategies, with a fine-grained categorization over informativeness (§8.2.1), representativeness (§8.2.2) and the combination of these two (§8.2.3).

[1]The descriptions in this survey are mostly brief to provide more comprehensive coverage in a compact way. We hope that this survey can serve as an index for related work.

Algorithm 1 A typical active learning procedure.

```
Input: An unlabeled data pool $\mathcal{U}$.
Output: The final labeled dataset $\mathcal{L}$ and trained model $\mathcal{M}$.
1: $\mathcal{L}, \mathcal{U} \leftarrow$ seed($\mathcal{U}$)                      ▷ Start (§8.5.1)
2: $\mathcal{M} \leftarrow$ train($\mathcal{L}, \mathcal{U}$)                   ▷ Model Learning (§8.4)
3: while not stop_criterion() do                            ▷ Stop (§8.5.2)
4:     $\mathcal{I} \leftarrow$ query($\mathcal{M}, \mathcal{U}$)               ▷ Query (§8.2, §8.3)
5:     $\mathcal{I}' \leftarrow$ annotate($\mathcal{I}$)                   ▷ Annotate (§8.3)
6:     $\mathcal{U} \leftarrow \mathcal{U} - \mathcal{I}$; $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{I}'$
7:     $\mathcal{M} \leftarrow$ train($\mathcal{L}, \mathcal{U}$)               ▷ Model Learning (§8.4)
8: return $\mathcal{L}, \mathcal{M}_f$
```

- In §8.3, we cover the two additional important topics of querying and annotating for NLP tasks: AL for structured prediction tasks (§8.3.1) and the cost of annotation with AL (§8.3.2).
- In §8.4, we discuss model and learning: the query-successor model mismatch scenario (§8.4.1) and AL with advanced learning techniques (§8.4.2).
- In §8.5, we examine methods for starting (§8.5.1) and stopping (§8.5.2) AL.

In §8.6, some other aspects of AL for NLP are discussed. In §8.7, we conclude with related and future directions.

8.2 Query Strategies

8.2.1 Informativeness

Informativeness-based query strategies mostly assign an informative measure to each unlabeled instance *individually*. The instance(s) with the highest measure will be selected.

Output Uncertainty

Uncertainty sampling (Lewis and Gale, 1994) is probably the simplest and the most commonly utilized query strategy. It prefers the most uncertain instances judged by the model outputs. For probabilistic models, entropy-based (Shannon, 1948), least-confidence (Culotta and McCallum, 2005) and margin-sampling (Scheffer et al., 2001; Schein and Ungar, 2007) are three typical uncertainty sampling strategies (Settles, 2009). Schröder et al. (2022) revisit some of these uncertainty-based strategies with Transformer-based models and provide empirical results for text classification. For non-probabilistic models, similar ideas can be utilized, such as selecting the instances that are close to the decision boundary in an SVM (Schohn and Cohn, 2000; Tong and Koller, 2001).

Another way to measure output uncertainty is to check the divergence of a model's predictions with respect to an instance's local region. If an instance is near the decision boundary, the model's outputs may be different within its local region. In this spirit, recent work examines different ways to check instances' **local divergence**, such as nearest-neighbor searches (Margatina et al., 2021), adversarial perturbation (Zhang et al., 2022c) and data augmentation (Jiang et al., 2020).

Disagreement

Uncertainty sampling usually considers the outputs of only one model. In contrast, **disagreement-based strategies** utilize multiple models and select the instances that are most disagreed among them. This is also a widely-adopted algorithm, of which the famous query-by-committee (QBC; Seung et al., 1992) is an example. The disagreement can be measured by vote entropy (Engelson and Dagan, 1996), KL-divergence (McCallum and Nigam, 1998) or variation ratio (Freeman, 1965).

To construct the model committee, one can train a group of distinct models. Moreover, taking a Bayesian perspective over the model parameters is also applicable (Houlsby et al., 2011). Especially with neural models, (Gal and Ghahramani, 2016) show that dropout could approximate inference and measure model uncertainty. This **deep Bayesian method** has been applied to AL for computer vision (CV) tasks (Gal et al., 2017) as well as various NLP tasks (Siddhant and Lipton, 2018; Shen et al., 2018; Shelmanov et al., 2021).

Gradient

Gradient information can be another signal for querying, with the motivation to choose the instances that would most strongly impact the model. In this strategy, informativeness is usually measured by the norm of the gradients. Since we do not know the gold labels for unlabeled instances, the loss is usually calculated as the expectation over all labels. This leads to the strategy of **expected gradient length** (EGL), introduced by Settles et al. (2007) and later applied to sequence labeling (Settles and Craven, 2008) and speech recognition (Huang et al., 2016a). Zhang et al. (2017b) explore a variation for neural networks where only the gradients of word embeddings are considered and show its effectiveness for text classification.

Performance Prediction

Predicting performance can be another indicator for querying. Ideally, the selected instances should be the ones that most **reduce future errors** if labeled and added to the training set. This motivates the **expected loss reduction** (ELR) strategy (Roy and McCallum, 2001), which chooses instances that lead to the least expected error if added to retrain a model. This strategy can be computationally costly since retraining is needed for each candidate. Along this direction, there have been several recent updates on the estimation metrics, for example, based on predicted error decay on groups Chang et al. (2020), mean objective cost of uncertainty (Zhao et al., 2021a) and strictly proper scoring rules (Tan et al., 2021).

Recently, methods have been proposed to learn another model to select instances that lead to the fewest errors, usually measured on a held-out development set. Reinforcement learning and imitation learning have been utilized to obtain the selection policy (Bachman et al., 2017; Fang et al., 2017; Liu et al., 2018b,a; Zhang et al., 2022e). This **learning-to-select strategy** may have some constraints. First, it requires labeled data (maybe from another domain) to train the policy.

To mitigate this reliance, Vu et al. (2019) use the current task model as an imperfect annotator for AL simulations. Moreover, the learning signals may be unstable for complex tasks, as Koshorek et al. (2019) show for semantic tasks.

A similar and simpler idea is to select the most erroneous or ambiguous instances with regard to the current task model, which can also be done with another **performance-prediction model**. Yoo and Kweon (2019) directly train a smaller model to predict the instance losses for CV tasks, which have been also adopted for NLP (Cai et al., 2021; Shen et al., 2021). In a similar spirit, Wang et al. (2017) employ a neural model to judge the correctness of the model prediction for SRL and Brantley et al. (2020) learn a policy to decide whether expert querying is required for each state in sequence labeling. Inspired by data maps (Swayamdipta et al., 2020), Zhang and Plank (2021) train a model to select ambiguous instances whose average correctness over the training iterations is close to a pre-defined threshold. For machine translation (MT), special techniques can be utilized to seek erroneous instances, such as using a backward translator to check round-trip translations (Haffari et al., 2009; Zeng et al., 2019) or quality estimation (Logacheva and Specia, 2014a,b; Chimoto and Bassett, 2022).

8.2.2 Representativeness

Only considering the informativeness of individual instances may have the drawback of sampling bias Dasgupta (2011); Prabhu et al. (2019) and the selection of outliers (Roy and McCallum, 2001; Karamcheti et al., 2021). Therefore, representativeness, which measures how instances correlate with each other, is another major factor to consider when designing AL query strategies.

Density

With the motivation to avoid outliers, density-based strategies prefer instances that are more **representative of the unlabeled set**. Selecting by n-gram or word counts (Ambati et al., 2010a; Zhao et al., 2020b) can be regarded as a simple way of density measurement. Generally, the common measurement is an instance's average similarity to all other instances (McCallum and Nigam, 1998; Settles and Craven, 2008). While it may be costly to calculate similarities of all instance pairs, considering only k-nearest neighbor instances has been proposed as an alternative option (Zhu et al., 2008c, 2009).

Discriminative

Another direction[2] is to select instances that are **different from already labeled instances**. Again, for NLP tasks, simple feature-based metrics can be utilized for this purpose by preferring instances with more unseen n-grams or out-of-vocabulary words (Eck et al., 2005; Bloodgood and Callison-Burch, 2010; Erdmann et al., 2019). Generally, similarity scores can also be utilized to select the instances that are less similar to the labeled set (Kim et al., 2006; Zhang et al., 2018; Zeng et al., 2019). Another interesting idea is to train a model to discriminate the labeled and unlabeled sets. Gissin and Shalev-Shwartz (2019) directly train a classifier for this purpose,

[2]Some work also uses the word "diversity,", however, we specifically preserve this word for batch-diversity in §8.2.2.

while naturally adversarial training can be also adopted (Sinha et al., 2019; Deng et al., 2018). In domain adaptation scenarios, the same motivation leads to the usage of a domain separator to filter instances (Rai et al., 2010).

Batch Diversity

Ideally, only one most useful instance would be selected in each iteration. However, it is more efficient and practical to adopt **batch-mode AL** (Settles, 2009), where each time a batch of instances is selected. In this case, we need to consider the dissimilarities not only between selected instances and labeled ones but also within the selected batch.

To select a batch of diverse instances, there are two common approaches. 1) **Iterative selection** collects the batch in an iterative greedy way (Brinker, 2003; Shen et al., 2004). In each iteration, an instance is selected by comparing it with previously chosen instances to avoid redundancy. Some more advanced diversity-based criteria, like coreset (Geifman and El-Yaniv, 2017; Sener and Savarese, 2018) and determinantal point processes (Shi et al., 2021), can also be approximated in a similar way. 2) **Clustering-based** methods partition the unlabeled data into clusters and select instances among them (Tang et al., 2002; Xu et al., 2003; Shen et al., 2004; Nguyen and Smeulders, 2004; Zhdanov, 2019; Yu et al., 2022; Maekawa et al., 2022). Since the chosen instances come from different clusters, diversity can be achieved to some extent.

To calculate similarity, in addition to comparing the input features or intermediate representations, other methods are also investigated, such as utilizing model-based similarity (Hazra et al., 2021), gradients (Ash et al., 2020; Kim, 2020), and masked LM surprisal embeddings (Yuan et al., 2020).

8.2.3 Hybrid

There is no surprise that informativeness and representativeness can be combined for instance querying, leading to hybrid strategies. A **simple combination** can be used to merge multiple criteria into one. This can be achieved by a weighted sum (Kim et al., 2006; Chen et al., 2011) or multiplication (Settles and Craven, 2008; Zhu et al., 2008c).

There are several strategies to **naturally integrate** multiple criteria. Examples include (uncertainty) weighted clustering (Zhdanov, 2019), diverse gradient selection (Ash et al., 2020; Kim, 2020) where the gradients themselves contain uncertainty information (§8.2.1) and determinantal point processes (DPP) with quality-diversity decomposition (Shi et al., 2021).

Moreover, **multi-step querying**, which applies multiple criteria in series, is another natural hybrid method. For example, one can consider first filtering certain highly uncertain instances and then performing clustering to select a diverse batch from them (Xu et al., 2003; Shen et al., 2004; Mirroshandel et al., 2011). An alternative strategy of selecting the most uncertain instances per cluster has also been utilized (Tang et al., 2002).

Instead of statically merging into one query strategy, **dynamic combination** may better fit the AL learning process, since different strategies may excel at different AL phases. For example, at the start of AL, uncertainty sampling may be unreliable due to little labeled data, and representativeness-based methods could be preferable, whereas in later stages where we have enough data and target finer-grained decision boundaries, uncertainty may be a suitable strategy.

DUAL (Donmez et al., 2007) is such a dynamic strategy that can switch from a density-based selector to an uncertainty-based one. Ambati et al. (2011b) further propose GraDUAL, which gradually switches strategies within a switching range. Wu et al. (2017) adopt a similar idea with a pre-defined monotonic function to control the combination weights.

8.3 Query and Annotation

8.3.1 AL for Structured Prediction

AL has been widely studied for classification tasks, while in NLP, many tasks involve *structured prediction*. In these tasks, the system needs to output a structured object consisting of a group of inter-dependent variables (Smith, 2011), such as a label sequence or a parse tree. Special care needs to be taken when querying and annotating for these more complex tasks (Thompson et al., 1999). One main decision is whether to annotate full structures for input instances (§8.3.1) or allow the annotation of only partial structures (§8.3.1).

Full-structure AL

First, if we regard the full output structure of an instance as a whole and perform query and annotation at the full-instance level, then AL for structured prediction tasks is not very different than for simpler classification tasks. Nevertheless, considering that the output space is usually exponentially large and infeasible to explicitly enumerate, querying may require further inspection.

Some **uncertainty sampling** strategies, such as entropy, need to consider the full output space. Instead of the infeasible explicit enumeration, dynamic-programming algorithms that are similar to the ones in decoding and inference processes can be utilized, such as algorithms for tree-entropy (Hwa, 2000, 2004) and sequence-entropy (Mann and McCallum, 2007; Settles and Craven, 2008).

Instead of considering the full output space, **top-k approximation** is a simpler alternative that takes k-best predicted structures as a proxy. This is also a frequently utilized method (Tang et al., 2002; Kim et al., 2006; Rocha and Sanchez, 2013).

For disagreement-based strategies, the measurement of **partial disagreement** may be required since full-match can be too strict for structured objects. Fine-grained evaluation scores can be reasonable choices for this purpose, such as the F1 score for sequence labeling (Ngai and Yarowsky, 2000).

Since longer instances usually have larger uncertainties and might be preferred, **length normalization** is a commonly-used heuristic to avoid this bias (Tang et al., 2002; Hwa, 2000, 2004; Shen et al., 2018). Yet, Settles and Craven (2008) argue that longer sequences should not be discouraged and may contain more information.

Instead of directly specifying the full utility of an instance, **aggregation** is also often utilized by gathering utilities of its sub-structures, usually along the factorization of the structured modeling. For example, the sequence uncertainty can be obtained by summing or averaging the uncertainties of all the tokens (Settles and Craven, 2008). Other aggregation methods are also applicable, such as weighted sum by word frequency (Ringger et al., 2007) or using only the most uncertain (least probable) one (Myers and Palmer, 2021; Liu et al., 2022c).

Partial-structure AL

A structured object can be decomposed into smaller sub-structures with different training utilities. For example, in a dependency tree, functional relations are usually easier to judge while prepositional attachment links may be more informative for the learning purpose. This naturally leads to AL with partial structures, where querying and annotating can be performed at the sub-structure level.

Factorizing full structures into the **finest-grained sub-structures** and regarding them as the annotation units could be a natural choice. Typical examples include individual tokens for sequence labeling (Marcheggiani and Artières, 2014), word boundaries for segmentation (Neubig et al., 2011; Li et al., 2012b), syntactic-unit pairs for dependency parsing (Sassano and Kurohashi, 2010) and mention pairs for coreference (Gasperin, 2009; Miller et al., 2012; Sachan et al., 2015). The querying strategy for the sub-structures can be similar to the classification cases, though inferences are usually needed to calculate marginal probabilities. Moreover, if full structures are desired as annotation outputs, semi-supervised techniques such as self-training (§8.4.2) could be utilized to assign pseudo labels to the unannotated parts (Tomanek and Hahn, 2009a; Majidi and Crane, 2013).

At many times, choosing **larger sub-structures** is preferable, since partial annotation still needs the understanding of larger contexts and frequently jumping among different contexts may require more reading time (§8.3.2). Moreover, increasing the sampling granularity may mitigate the missed class effect, where certain classes may be overlooked (Tomanek et al., 2009). Typical examples of larger sub-structures include sub-sequences for sequence labeling (Shen et al., 2004; Chaudhary et al., 2019; Radmard et al., 2021), word-wise head edges for dependency parsing (Flannery and Mori, 2015; Li et al., 2016), neighborhood pools (Laws et al., 2012) or mention-wise anaphoric links (Li et al., 2020a; Espeland et al., 2020) for coreference, and phrases for MT (Bloodgood and Callison-Burch, 2010; Miura et al., 2016; Hu and Neubig, 2021). In addition to increasing granularity, **grouping queries** can also help to make annotation easier, such as adopting a two-stage selection of choosing uncertain tokens from uncertain sentences (Mirroshandel and Nasr, 2011; Flannery and Mori, 2015) and selecting nearby instances in a row (Miller et al., 2012).

For AL with partial structures, **output modeling** is of particular interest since the model needs to learn from partial annotations. If directly using local discriminative models where each sub-structure is decided independently, learning with partial annotations is straightforward since the annotations are already complete to the models (Neubig et al., 2011; Flannery and Mori, 2015). For more complex models that consider interactions among output sub-structures, such as global models, special algorithms are required to learn from incomplete annotations (Scheffer et al., 2001; Wanvarie et al., 2011; Marcheggiani and Artières, 2014; Li et al., 2016). One advantage of these more complex models is the interaction of the partial labels and the remaining parts. For example, considering the **output constraints** for structured prediction tasks, combining the annotated parts and the constraints may reduce the output space of other parts and thus lower their uncertainties, leading to better queries (Roth and Small, 2006; Sassano and

Kurohashi, 2010; Mirroshandel and Nasr, 2011). More generally, the annotation of one label can intermediately influence others with cheap re-inference, which can help batch-mode selection (Marcheggiani and Artières, 2014) and interactive correction (Culotta and McCallum, 2005).

In addition to classical structured-prediction tasks, classification tasks can also be cast as structured predictions with partial labeling. **Partial feedback** is an example that is adopted to make the annotating of classification tasks simpler, especially when there are a large number of target labels. For example, annotators may find it much easier to answer yes/no questions (Hu et al., 2019) or rule out negative classes (Lippincott and Van Durme, 2021) than to identify the correct one.

8.3.2 Annotation Cost

AL mainly aims to reduce real annotation costs and we discuss several important topics on this point.

Cost Measurement

Most AL work adopts simple measurements of unit cost, that is, assuming that annotating each instance requires the same cost. Nevertheless, the annotation efforts for different instances may vary (Settles et al., 2008). For example, longer sentences may cost more to annotate than shorter ones. Because of this, many papers assume unit costs to tokens instead of sequences, which may still be inaccurate. Especially, AL tends to select difficult and ambiguous instances, which may require more annotation efforts (Hachey et al., 2005; Lynn et al., 2012). It is important to **properly measure annotation cost** since the measurement directly affects the evaluation of AL algorithms. The comparisons of query strategies may vary if adopting different cost measurements (Haertel et al., 2008a; Bloodgood and Callison-Burch, 2010; Chen et al., 2015b).

Probably the best cost measurement is the actual **annotation time** (Baldridge and Palmer, 2009). Especially, when the cost comparisons are not that straightforward, such as comparing annotating data against writing rules (Ngai and Yarowsky, 2000) or partial against full annotations (§8.3.1; Flannery and Mori, 2015; Li et al., 2016, 2020a), time-based evaluation is an ideal choice. This requires actual annotating exercises rather than simulations.

Since cost measurement can also be used for querying (§8.3.2), it would be helpful to be able to **predict the real cost** before annotating. This can be cast as a regression problem, for which some previous work learns a linear cost model based on input features (Settles et al., 2008; Ringger et al., 2008; Haertel et al., 2008a; Arora et al., 2009).

Cost-sensitive Querying

Given the goal of reducing actual cost, the querying strategies should also take it into consideration. That is, we want to select not only high-utility instances but also low-cost ones. A natural cost-sensitive querying strategy is **return-on-investment** (ROI; Haertel et al., 2008b; Settles et al., 2008; Donmez and Carbonell, 2008). In this strategy, instances with higher net benefit per unit cost are preferred, which is equivalent to dividing the original querying utility by cost measure. Tomanek and Hahn (2010) evaluate the effectiveness of ROI together with two other strategies, including constraining maximal cost budget per instance and weighted rank combination. Haertel et al. (2015) provide further analytic and empirical evaluation, showing that ROI can reduce total cost.

In real AL scenarios, things can be much more complex. For example, there can be multiple annotators with different expertise (Baldridge and Palmer, 2009; Huang et al., 2017; Cai et al., 2020), and the annotators may refuse to answer or make mistakes (Donmez and Carbonell, 2008). Being aware of these scenarios, Donmez and Carbonell (2008) propose **proactive learning** to jointly select the optimal oracle and instance. Li et al. (2017) further extend proactive learning to NER tasks.

Directly Reducing Cost

In addition to better query strategies, there are other ways of directly reducing annotation cost, such as computer-assisted annotation. In AL, models and annotators usually interact in an indirect way where models only query the instances to present to the annotators, while there could be closer interactions.

Pre-annotation is such an idea, where not only the raw data instances but also the model's best or top-k predictions are sent to the annotators to help them make decisions. If the model's predictions are reasonable, the annotators can simply select or make a few corrections to obtain the gold annotations rather than create them from scratch. This method has been shown effective when combined with AL (Baldridge and Osborne, 2004; Vlachos, 2006; Ringger et al., 2008; Skeppstedt, 2013; Cañizares-Díaz et al., 2021). Post-editing for MT is also a typical example (Dara et al., 2014).

Moreover, the models could provide help at **real annotating time**. For example, Culotta and McCallum (2005) present an interactive AL system where the user's corrections can propagate to the model, which generates new predictions for the user to further refine. Interactive machine translation (IMT) adopts a similar idea, where the annotator corrects the first erroneous character, based on which the model reproduces the prediction. AL has also been combined with IMT to further reduce manual efforts (González-Rubio et al., 2012; Peris and Casacuberta, 2018; Gupta et al., 2021a).

Crowdsourcing is another way to reduce annotation costs and can be combined with AL. We provide more discussions on this in §8.6.

Wait Time

In AL iterations, the annotators may need to wait for the training and querying steps (Line 7 and 4 in Algorithm 1). This wait time may bring some hidden costs, thus more efficient querying and training would be preferable for faster turnarounds.

To speed up **querying**, sub-sampling is a simple method to deal with large unlabeled pools (Roy and McCallum, 2001; Ertekin et al., 2007; Tsvigun et al., 2022; Maekawa et al., 2022). For some querying strategies, pre-calculating and caching unchanging information can also help to speed up (Ashrafi Asli et al., 2020; Citovsky et al., 2021). In addition, approximation with k-nearest neighbors can also be utilized to calculate density (Zhu et al., 2009) or search for instances after adversarial attacks (Ru et al., 2020).

To reduce **training** time, a seemingly reasonable strategy is to apply incremental training across AL iterations, that is, continuing training previous models on the new instances. However, Ash and Adams (2020) show that this type of warm-start may lead to sub-optimal performance for neural models and recent AL work usually trains models from scratch (Hu et al., 2019; Ein-Dor et al., 2020). Another method is to use an efficient model for querying and a more powerful model for final training. However, this might lead to sub-optimal results, which will be discussed in §8.4.1.

Another idea to reduce wait time is to simply allow querying with **stale information**. Actually, batch-mode AL (§8.2.2) is such an example where instances in the same batch are queried with the same model. Haertel et al. (2010) propose parallel AL, which maintains separate loops of annotating, training, and scoring, and allows dynamic and parameterless instance selection at any time.

8.4 Model and Learning

8.4.1 Model Mismatch

While it is natural to adopt the same best-performing model throughout the AL process, there are cases where the query and final (successor) models can mismatch (Lewis and Catlett, 1994). Firstly, more efficient models are preferable for querying to reduce wait time (§8.3.2). Moreover, since data usually outlive models, re-using AL-base data to train another model would be desired (Baldridge and Osborne, 2004; Tomanek et al., 2007). Previous work shows that model mismatch may make the gains from AL negligible or even negative (Baldridge and Osborne, 2004; Lowell et al., 2019; Shelmanov et al., 2021), which raises concerns about the utilization of AL in practice.

For efficiency purposes, distillation can be utilized to improve querying efficiency while keeping reasonable AL performance. Shelmanov et al. (2021) show that using a smaller distilled version of a pre-trained model for querying does not lead to too much performance drop. Tsvigun et al. (2022) combine this idea with pseudo-labeling and sub-sampling to further reduce computational cost. Similarly, Nguyen et al. (2022) keep a smaller proxy model for query and synchronize the proxy with the main model by distillation.

8.4.2 Learning

AL can be combined with other advanced learning techniques to further reduce required annotations.

Semi-supervised learning. Since AL usually assumes an unlabeled pool, semi-supervised learning can be a natural fit. Combining these two is not a new idea: (McCallum and Nigam, 1998) adopt the EM algorithm to estimate the outputs of unlabeled data and utilize them for learning. This type of self-training or pseudo-labeling technique is often utilized in AL (Tomanek and Hahn, 2009a; Majidi and Crane, 2013; Yu et al., 2022). With similar motivation, (Dasgupta and Ng, 2009) use an unsupervised algorithm to identify the unambiguous instances to train an active learner. For the task of word alignment, which can be learned in an unsupervised manner, incorporating supervision with AL can bring further improvements in a data-efficient way (Ambati et al., 2010b,c).

Transfer learning. AL can be easily combined with transfer learning, another technique to reduce required annotations. Utilizing pre-trained models is already a good example (Ein-Dor et al., 2020; Yuan et al., 2020; Tamkin et al., 2022) and continual training (Gururangan et al., 2020) can also be applied (Hua and Wang, 2022; Margatina et al., 2022). Moreover, transductive learning is commonly combined with AL by transferring learning signals from different domains (Chan and Ng, 2007; Shi et al., 2008; Rai et al., 2010; Saha et al., 2011; Wu et al., 2017; Kasai et al., 2019; Yuan et al., 2022) or languages (Qian et al., 2014; Fang and Cohn, 2017; Fang et al., 2017; Chaudhary et al., 2019, 2021; Moniz et al., 2022). Initializing the model via meta-learning has also been found helpful (Zhu et al., 2022). In addition to the task model, the model-based query policy (§8.2.1) is also often obtained with transfer learning.

Weak supervision. AL can also be combined with weakly supervised learning. Examples include learning from execution results for semantic parsing (Ni et al., 2020), labeling based on structure vectors for entity representations (Qian et al., 2020), learning from gazetteers for sequence labeling (Brantley et al., 2020) and interactively discovering labeling rules (Zhang et al., 2022b).

Data augmentation. Augmentation is also applicable in AL and has been explored with iterative back-translation (Zhao et al., 2020b), mixup for sequence labeling (Zhang et al., 2020a) and phrase-to-sentence augmentation for MT (Hu and Neubig, 2021). As discussed in §8.2.1, augmentation can also be helpful for instance querying (Jiang et al., 2020; Zhang et al., 2022c). Another interesting scenario involving augmentation and AL is query synthesis, which directly generates instances to be annotated instead of selecting existing unlabeled ones. Though synthesizing texts is still a hard problem generally, there have been successful applications for simple classification tasks (Schumann and Rehbein, 2019; Quteineh et al., 2020).

8.5 Starting and Stopping AL

8.5.1 Starting AL

While there are cases where there are already enough labeled data to train a reasonable model and AL is utilized to provide further improvements (Bloodgood and Callison-Burch, 2010; Geifman and El-Yaniv, 2017), at many times we are facing the cold-start problem, where instances need to be selected without a reasonable model. Especially, how to select the seed data to start the AL process is an interesting question, which may greatly influence the performance in initial AL stages (Tomanek et al., 2009; Horbach and Palmer, 2016).

Random sampling is probably the most commonly utilized strategy, which is reasonable since it preserves the original data distribution. Some representativeness-based querying strategies (§8.2.2) can also be utilized, for example, selecting points near the clustering centroids is a way to obtain representative and diverse seeds (Kang et al., 2004; Zhu et al., 2008c; Hu et al., 2010). Moreover, some advanced learning techniques (§8.4.2) can also be helpful here, such as transfer learning (Wu et al., 2017) and unsupervised methods (Vlachos, 2006; Dasgupta and Ng, 2009). In addition, the language model can be a useful tool, with which Dligach and Palmer (2011) select low-probability words in the context of word sense disambiguation and Yuan et al. (2020) choose cluster centers with surprisal embeddings by pre-trained contextualized LMs.

8.5.2 Stopping AL

When adopting AL in practice, it would be desirable to know the time to stop AL when the model performance is already near the upper limits, before running out of all the budgets. For this purpose, a stopping criterion is needed, which checks certain metrics satisfying certain conditions. There can be simple heuristics. For example, AL can be stopped when all unlabeled instances are no closer than any of the support vectors with an SVM (Schohn and Cohn, 2000; Ertekin et al., 2007) or no new n-grams remain in the unlabeled set for MT (Bloodgood and Callison-Burch, 2010). Nevertheless, these are not generic solutions. For the design of a general stopping criterion, there are three main aspects to consider: *metric*, *dataset*, and *condition*.

For the **metric**, measuring performance on a development set seems a natural option. However, the results would be unstable if this set is too small and it would be impractical to assume a large development set. Cross-validation on the training set also has problems since the labeled data by AL is usually biased. In this case, metrics from the query strategies can be utilized. Examples include uncertainty or confidence (Zhu and Hovy, 2007; Vlachos, 2008), disagreement (Tomanek et al., 2007; Tomanek and Hahn, 2008; Olsson and Tomanek, 2009), estimated performance (Laws and Schütze, 2008), expected error (Zhu et al., 2008a), confidence variation (Ghayoomi, 2010), as well as actual performance on the selected instances (Zhu and Hovy, 2007). Moreover, comparing the predictions between consecutive AL iterations is another reasonable option (Zhu et al., 2008b; Bloodgood and Vijay-Shanker, 2009a).

The **dataset** to calculate the stopping metric requires careful choosing. The results could be unstable if not adopting a proper set (Tomanek and Hahn, 2008). Many papers suggest that a separate unlabeled dataset should be utilized (Tomanek and Hahn, 2008; Vlachos, 2008; Bloodgood and Vijay-Shanker, 2009a; Beatty et al., 2019; Kurlandski and Bloodgood, 2022). Since the stopping metrics usually do not rely on gold labels, this dataset could potentially be very large to provide more stable results, though wait time would be another factor to consider in this case (§8.3.2).

The **condition** to stop AL is usually comparing the metrics to a pre-defined threshold. Earlier work only looks at the metric at the current iteration, for example, stopping if the uncertainty or the error is less than the threshold (Zhu and Hovy, 2007). In this case, the threshold is hard to specify since it relies on the model and the task. (Zhu et al., 2008b) cascade multiple stopping criteria to mitigate this reliance. A more stable option is to track the change of the metrics over several AL iterations, such as stopping when the confidence consistently drops (Vlachos, 2008), the changing rate flattens (Laws and Schütze, 2008) or the predictions stabilize across iterations (Bloodgood and Vijay-Shanker, 2009a; Bloodgood and Grothendieck, 2013).

Pullar-Strecker et al. (2021) provide an empirical comparison over common stopping criteria and would be a nice reference. Moreover, stopping AL can be closely related to performance prediction and early stopping. Especially, the latter can be of particular interest since learning in early AL stages is a low-resource problem, and how to perform early stopping may also require careful considerations.

8.6 Other Aspects

We describe some other aspects that are frequently seen when applying AL to NLP.

Crowdsourcing and Noise. Crowdsourcing is another way to reduce annotation costs by including non-expert annotations (Snow et al., 2008). Naturally, AL and crowdsourcing may also be combined with the hope to further reduce cost (Ambati et al., 2010a; Laws et al., 2011; Yan et al., 2011; Fang et al., 2014; Zhao et al., 2020c). One specific factor to consider in this case is the noises in the crowdsourced data, since noisy data may have a negative impact on the effectiveness of AL (Rehbein and Ruppenhofer, 2011). Cost-sensitive querying strategies (§8.3.2) can be utilized to select both annotators and instances by estimating labelers' reliability (Yan et al., 2011; Fang et al., 2014). Requiring multiple annotations per instance and then consolidating is also applicable (Laws et al., 2011). Lin et al. (2019a) provide a framework that enables automatic crowd consolidation for AL on the tasks of sequence labeling.

Multiple Targets. In many cases, we may want to consider multiple targets rather than only one, for example, annotating instances in multiple domains (Xiao and Guo, 2013; He et al., 2021; Longpre et al., 2022) or multiple languages (Haffari and Sarkar, 2009; Qian et al., 2014; Moniz et al., 2022). Moreover, there may be multiple target tasks, where multi-task learning (MTL) can interact with AL (Reichart et al., 2008; Ambati et al., 2011a; Rocha and Sanchez, 2013; Ikhwantri et al., 2018; Zhu et al., 2020; Rotman and Reichart, 2022). In these scenarios with multiple targets, naturally, strategies that consider all the targets are usually more preferable. Reichart et al. (2008) show that a query strategy that considers all target tasks obtains the overall best performance for MTL. Moniz et al. (2022) suggest that joint learning across multiple languages using a single model outperforms other strategies such as equally dividing budgets or allocating only for a high-resource language and then performing the transfer.

Data Imbalance. Imbalance is a frequently occurring phenomenon in NLP and AL can have interesting interactions with it. On the one hand, as in plain learning scenarios, AL should take data imbalance into consideration, with modifications to the model (Bloodgood and Vijay-Shanker, 2009b), learning algorithm (Zhu and Hovy, 2007) and query strategies (Tomanek et al., 2009; Escudeiro and Jorge, 2010; Li et al., 2012a). On the other hand, AL can be utilized to address the data imbalance problem and build better data (Ertekin et al., 2007; Tomanek and Hahn, 2009b; Attenberg and Ertekin, 2013; Mottaghi et al., 2020; Mussmann et al., 2020).

8.7 Related Topics and Future Directions

8.7.1 Related Topics

There are many related topics that could be explored together with AL. Other data-efficient learning methods such as semi-supervised and transfer learning are naturally compatible with AL (§8.4.2). Curriculum learning (Bengio et al., 2009), which arranges training instances in a meaningful order, may also be integrated with AL (Platanios et al., 2019; Zhao et al., 2020a; Jafarpour et al., 2021). Uncertainty (Gawlikowski et al., 2021), outlier detection (Hodge and Austin, 2004), and performance prediction (Xia et al., 2020) can be related to instance querying. Crowdsourcing can be adopted to further reduce annotation cost (§8.6). Model efficiency (Menghani, 2021) would be crucial to reduce wait time (§8.3.2). AL is a typical type of human-in-the-loop framework (Wang et al., 2021c), and it will be interesting to explore more human-computer interaction techniques in AL.

8.7.2 Future Directions

Complex tasks. AL is mostly adopted for simple classification, while there are many more complex tasks in NLP. For example, except for MT, generation tasks have been much less thoroughly explored with AL. Tasks with more complex inputs such as NLI and QA also require extra care when using AL; obtaining unlabeled data is already non-trivial. Nevertheless, preliminary work has shown that AL can be helpful for data collection for such tasks (Mussmann et al., 2020).

Beyond direct target labeling. In addition to directly annotating target labels, AL can also be utilized in other ways to help the target task, such as labeling features or rationales (Melville and Sindhwani, 2009; Druck et al., 2009; Sharma et al., 2015), annotating explanations (Liang et al., 2020), evaluation (Mohankumar and Khapra, 2022) and rule discovery (Zhang et al., 2022b).

AL in practice. Most AL work simulates annotations on an existing labeled dataset. Though this method is convenient for algorithm development, it ignores several challenges of applying AL in practice. As discussed in this survey, real annotation cost (§8.3.2), efficiency and wait time (§8.3.2), data reuse (§8.4.1) and starting and stopping (§8.5) are all important practical aspects which may not emerge in simulation. Moreover, since the AL process usually cannot be repeated multiple times, how to select the query strategy and other hyper-parameters remains a great challenge. It will be critical to address these issues to bring AL into practical use (Rehbein et al., 2010; Attenberg and Provost, 2011; Settles, 2011; Lowell et al., 2019) and make it more widely utilized (Tomanek and Olsson, 2009).

Chapter 9

Data-efficient Active Learning for Structured Prediction with Partial Annotation and Self-Training

As discussed in the previous chapter, active learning is a promising way to battle the lacking of annotation budgets and learn models with fewer labeled data instances. In this chapter, we explore more efficient active learning for structure prediction in natural language processing. We show that the combination of partial annotation and self-training using an adaptive selection ratio reduces annotation cost over strong full annotation baselines under a fair comparison scheme that takes reading time into consideration.

9.1 Introduction

Structured prediction (Smith, 2011) is a fundamental problem in NLP, wherein the label space consists of complex structured outputs with groups of interdependent variables. It covers a wide range of NLP tasks, including sequence labeling, syntactic parsing, and information extraction (IE). Modern structured predictors are developed in a data-driven way, by training statistical models with suitable annotated data. Recent developments in neural models and especially pre-trained language models (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019) have greatly improved system performance on these tasks. Nevertheless, the success of these models still relies on the availability of sufficient manually annotated data, which is often expensive and time-consuming to obtain.

To mitigate such data bottlenecks, active learning (AL), which allows the model to select the most informative data instances to annotate, has been demonstrated to achieve good model accuracy while requiring fewer labels (Settles, 2009). When applying AL to structured prediction, one natural strategy is to perform full annotation (FA) for the output structures, for example, annotating a full sequence of labels or a full syntax tree. Due to its simplicity, FA has been widely adopted in AL approaches for structured prediction tasks (Hwa, 2004; Settles and Craven, 2008; Shen et al., 2018). Nevertheless, a structured object can usually be decomposed into smaller sub-structures having non-uniform difficulty and informativeness. For example, as shown in

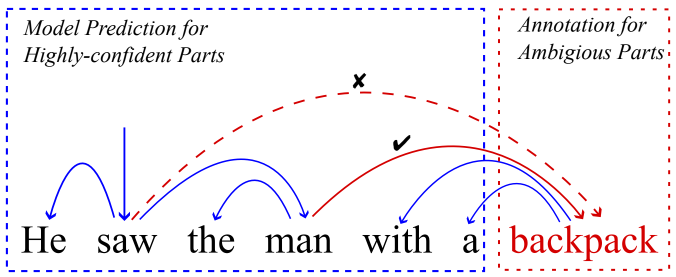

Model Prediction for Highly-confident Parts
Annotation for Ambigious Parts
✘
✔
He saw the man with a backpack

Figure 9.1: Example partial annotation of a dependency tree.

Figure 9.1, in a dependency tree, edges such as functional relations are relatively easy to learn, requiring fewer manual annotations, while prepositional attachment links may be more informative and thus more worthwhile to annotate.

The non-uniform distribution of informative sub-structures naturally suggests AL with partial annotation (PA), where the annotation budgets can be preserved by only choosing a portion of informative sub-structures to annotate rather than laboriously labeling entire sentence structures. This idea has been explored in previous work, covering typical structured prediction tasks such as sequence labeling (Shen et al., 2004; Marcheggiani and Artières, 2014; Chaudhary et al., 2019; Radmard et al., 2021) and dependency parsing (Sassano and Kurohashi, 2010; Mirroshandel and Nasr, 2011; Flannery and Mori, 2015; Li et al., 2016). Our work follows this direction and investigates the central question in AL with PA of how to decide which sub-structures to select. Most previous work uses a pre-defined fixed selection criterion, such as a threshold or ratio, which may be hard to decide in practice. In this chapter, we adopt a performance predictor to estimate the error rate of the queried instances and decide the ratio of partial selection accordingly. In this way, our approach can automatically and adaptively adjust the amount of partial selection throughout the AL process.

Another interesting question for AL is how to better leverage unlabeled data. In this chapter, we investigate a simple semi-supervised method, self-training (Yarowsky, 1995), which adopts the model's automatic predictions on the unlabeled data as extra training signals. Self-training naturally complements AL in the typical pool-based setting where we assume access to a pool of unlabeled data (Settles, 2009). It is particularly compatible with PA-based AL since the unselected sub-structures are typically also highly-confident under the current model and likely to be predicted correctly without requiring additional annotation. We revisit this idea from previous work (Tomanek and Hahn, 2009a; Majidi and Crane, 2013) and investigate its applicability with modern neural models and our adaptive partial selection approach.

We perform a comprehensive empirical investigation on the effectiveness of different AL strategies for typical structured prediction tasks. We perform fair comparisons that account for the hidden cost of reading time by keeping the context size the same for all the strategies in each AL cycle. With evaluations on four benchmark tasks for structured prediction (named entity recognition, dependency parsing, event extraction, and relation extraction), we show that PA can obtain roughly the same benefits as FA with the same reading cost but less sub-structure labeling cost, leading to better data efficiency. We also demonstrate that the adaptive partial selection scheme and self-training play crucial and complementary roles.

Algorithm 2 AL Procedure.

Input: Seed dataset $\mathcal{L}_0$, dev dataset $\mathcal{D}$, unlabeled pool $\mathcal{U}$, total budget t, batch selection size b, annotation $strategy$.

Output: Final labeled dataset $\mathcal{L}$, trained model $\mathcal{M}$.

```
1: $\mathcal{L} \leftarrow \mathcal{L}_0$   # Initialize
2: while $t > 0$ do   # Until out of budget
3:     $\mathcal{M} \leftarrow$ train($\mathcal{L}$, $\mathcal{U}$)   # Model training
4:     $\mathcal{S} \leftarrow$ sentence-query($\mathcal{M}$, $\mathcal{U}$)   # Sentence selection
5:     if $strategy$ == "partial" then
6:         $r \leftarrow$ auto-ratio($\mathcal{S}, \mathcal{D}$)   # Decide adaptive ratio
7:         partial-annotate($\mathcal{S}, r$)   # Partial annotation
8:     else
9:         full-annotate($\mathcal{S}$)   # Full annotation
10:    $\mathcal{U} \leftarrow \mathcal{U} - \mathcal{S}$;  $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{S}$;  $t \leftarrow t - b$
11: $\mathcal{M} \leftarrow$ train($\mathcal{L}$, $\mathcal{U}$)   # Final model training
12: return $\mathcal{L}$, $\mathcal{M}$
```

9.2 Method

9.2.1 AL for Structured Prediction

We adopt the common pool-based AL setting, which iteratively selects and annotates instances from an unlabeled pool. Please refer to Settles (2009) for the basics and details of AL; our main illustration focuses more specifically on different AL strategies for structured prediction.

Algorithm 2 illustrates the overall AL process. We focus on sentence-level tasks. In FA, each sentence is annotated with a full structured object (for example, a label sequence or a syntax tree). In PA, annotation granularity is at the sub-structure level (for example, a sub-sequence of labels or a partial tree). We adopt a two-step selection approach for all the strategies by first choosing a batch of sentences and then annotating within this batch. This approach is natural for FA since the original aim is to label full sentences, and it is also commonly adopted in previous PA work (Mirroshandel and Nasr, 2011; Flannery and Mori, 2015; Li et al., 2016). Moreover, this approach makes it easier to control the reading context size for fair comparisons of different strategies as described in §9.3.2.

Without loss of generality, we take sequence labeling as an example here and illustrate several key points in the AL process. While there are multiple options for each component or step in AL, we focus on the comparison of AL strategies and use commonly adopted settings for other aspects.

- **Model**. We adopt a standard BERT-based model with a CRF output layer for structured output modeling (Lafferty et al., 2001), together with the BIO tagging scheme. With this model, we can obtain the marginal probability distributions over the labels for each input token.
- **Querying Strategy**. We utilize the query-by-uncertainty strategy with the margin-based metric, which has been shown effective in AL for structured prediction (Marcheggiani and Artières, 2014; Li et al., 2016). Specifically, each token obtains an uncertainty score with the difference

between the (marginal) probabilities of the most and second most likely label. We also tried several other strategies in preliminary experiments, such as maximum probability or entropy, but did not find obvious benefits.

- **Sentence selection**. For both FA and PA, selecting a batch of uncertain sentences is the first querying step. We use the number of total tokens to measure batch size since sentences may have variant lengths. The sentence-level uncertainty is obtained by averaging the token-level ones. This length normalization heuristic is commonly adopted to avoid biases towards longer sentences (Hwa, 2004; Shen et al., 2018).
- **Token selection**. In PA, a subset of highly uncertain tokens is further chosen for annotation. One important question is how many tokens to select. Instead of using a pre-defined fixed selection criterion, we develop an adaptive strategy to decide the amount, as will be described in §9.2.2.
- **Annotation**. Sequence labeling is usually adopted for tasks involving mention extraction, where annotations are over spans rather than individual tokens. Previous work explores sub-sequence querying (Chaudhary et al., 2019; Radmard et al., 2021), which brings further complexities. Since we mainly explore tasks with short mention spans, we adopt a simple annotation protocol: Labeling the full spans where any inside token is queried. Note that for annotation cost measurement, we also include the extra labeled tokens in addition to the queried ones.
- **Model learning**. For FA, we adopt the standard log-likelihood as the training loss. For PA, we follow previous work (Scheffer et al., 2001; Wanvarie et al., 2011; Marcheggiani and Artières, 2014) and adopt marginalized likelihood to learn from incomplete annotations (Tsuboi et al., 2008; Greenberg et al., 2018).

9.2.2 Adaptive Partial Selection

PA adopts a second selection stage to choose highly uncertain sub-structures within the selected sentences. One crucial question here is how many sub-structures to select. Typical solutions in previous work include setting an uncertainty threshold (Tomanek and Hahn, 2009a) or specifying a selection ratio (Li et al., 2016). The threshold or ratio is usually pre-defined with a fixed hyper-parameter.

This fixed selection scheme might not be an ideal one. First, it is usually hard to specify such fixed values in practice. If too many sub-structures are selected, there will be little difference between FA and PA, whereas if too few, the annotation amount is insufficient to train good models. Moreover, this scheme is not adaptive to the model. As the model is trained with more data throughout the AL process, the informative sub-structures become less dense as the model improves. Thus, the number of selected sub-structures should be adjusted accordingly. To mitigate these shortcomings, we develop a dynamic strategy that can decide the selection in an automatic and adaptive way.

We adopt the ratio-based strategy which enables straightforward control of the selected amount. Specifically, we rank the sub-structures by the uncertainty score and choose those scoring highest by the ratio. Our decision on the selecting ratio is based on the hypothesis that a reasonable ratio should roughly correspond to the current model's error rate on all the candidates. The intuition is that incorrectly predicted sub-structures are the most informative ones that can help to correct the model's mistakes.

Since the queried instances come from the unlabeled pool without annotations, the error rate cannot be directly obtained, requiring estimation. We adopt a simple one-dimensional logistic regression model for this purpose. The input to the model is the uncertainty score[1] and the output is a binary prediction of whether its prediction is confidently correct[2] or not. The estimator is trained using all the sub-structures together with their correctness on the development set[3] and then applied to the queried candidates. For each candidate sub-structure s, the estimator will give it a correctness probability. We estimate the overall error rate as one minus the average correctness probability over all the candidates in the query set $\mathcal{Q}$, and set the selection ratio r as this error rate:

$$r = 1 - \frac{1}{n} \sum_{s \in \mathcal{Q}} p(correct = 1|s)$$

In this way, the selection ratio can be set adaptively according to the current model's performance. If the model is weak and makes many mistakes, we will have a larger ratio which can lead to more dense annotations and richer training signals. As the model is trained with more data and makes fewer errors, the ratio will be tuned down correspondingly to avoid wasting annotation budgets on already-correctly-predicted sub-structures. As we will see in later experiments, this adaptive scheme is suitable for AL (§9.3.3).

9.2.3 Self-training

Better utilization of unlabeled data is a promising direction to further enhance model training in AL since unlabeled data are usually freely available from the unlabeled pool. In this chapter, we adopt self-training (Yarowsky, 1995) for this purpose.

The main idea of self-training is to enhance the model training with pseudo labels that are predicted by the current model on the unlabeled data. It has been shown effective for various NLP tasks (Yarowsky, 1995; McClosky et al., 2006; He et al., 2020; Du et al., 2021a). For the training of AL models, self-training can be seamlessly incorporated. For FA, the application of self-training is no different than that in the conventional scenarios by applying the current model to all the unannotated instances in the unlabeled pool. The more interesting case is on the partially annotated instances in the PA regime. The same motivation from the adaptive ratio scheme (§9.2.2) also applies here: We select the highly-uncertain sub-structures that are error-prone and the remaining un-selected parts are likely to be correctly predicted; therefore we can trust the predictions on the un-selected sub-structures and include them for training. One more enhancement to apply here is that we could further perform re-inference by incorporating the updated annotations over the selected sub-structures, which can enhance the predictions of unannotated sub-structures through output dependencies.

[1] We transform the input with a logarithm, which leads to better estimation according to preliminary experiments.

[2] The specific criterion is that the $\arg\max$ prediction matches the gold one and its margin is greater than 0.5. Since neural models are usually over-confident, it is hard to decide on a confidence threshold. Nevertheless, we find 0.5 a reasonable value for the ratio decision here.

[3] We re-use the development set for the task model training.

In this chapter, we adopt a soft version of self-training through knowledge distillation (KD; Hinton et al., 2015). This choice is because we want to avoid the potential negative influences of ambiguous predictions (mostly in completely unlabeled instances). One way to mitigate this is to set an uncertainty threshold and only utilize the highly-confident sub-structures. However, it is unclear how to set a proper value, similar to the scenarios in query selection. Therefore, we take the model's full output predictions as the training targets without further processing.

Specifically, our self-training objective function is the cross-entropy between the output distributions predicted by the previous model m' before training and the current model m being trained:

$$\mathcal{L} = -\sum_{y \in \mathcal{Y}} p_{m'}(y|x) \log p_m(y|x)$$

Several points are notable here: 1) The previous model is kept unchanged, and we can simply cache its predictions before training; 2) Over the instances that have partial annotations, the predictions should reflect these annotations by incorporating corresponding constraints at inference time; 3) For tasks with CRF based models, the output space $\mathcal{Y}$ is usually exponentially large and infeasible to explicitly enumerate; we utilize special algorithms (Wang et al., 2021b) to deal with this.

Finally, we find it beneficial to include both the pseudo labels and the real annotated gold labels for the model training. With the gold data, the original training loss is adopted, while the KD objective is utilized with the pseudo labels. We simply mix these two types of data with a ratio of 1:1 in the training process, which we find works well.

9.3 Experiments

9.3.1 Main Settings

Tasks and data. Our experiments are conducted over four English tasks. The first two are named entity recognition (NER) and dependency parsing (DPAR), which are representative structured prediction tasks for predicting sequence and tree structures. We adopt the CoNLL-2003 English dataset (Tjong Kim Sang and De Meulder, 2003) for NER and the English Web Treebank (EWT) from Universal Dependencies v2.10 (Nivre et al., 2020) for DPAR. Moreover, we explore two more complex IE tasks: Event extraction and relation extraction. Each task involves two pipelined sub-tasks: The first aims to extract the event trigger and/or entity mentions, and the second predicts links between these mentions as event arguments or entity relations. We utilize the ACE05 dataset (Walker et al., 2006) for these IE tasks.

AL. For the AL procedure, we adopt settings following conventional practices. We use the original training set as the unlabeled data pool to select instances. Unless otherwise noted, we set the AL batch size (for sentence selection) to 4K tokens, which roughly corresponds to 2% of the total pool size for most of the datasets we use. The initial seed training set and the development set are randomly sampled (with FA) using this batch size. Unless otherwise noted, we run 14 AL cycles for each experiment. Following most AL work, annotation is simulated by checking and assigning the labels from the original dataset. In FA, we annotate all the sub-structures for the selected sentences. In PA, we first decide the selection ratio and apply it to the selected sentences. We further adopt a heuristic that selects the union of sentence-wise uncertain sub-structures as well as global ones since both may contain informative sub-structures. Finally, all the presented results are averaged over five runs with different random seeds.

Model and training. For the models, we adopt standard architectures by stacking task-specific structured predictors over pre-trained RoBERTa$_{\text{base}}$ (Liu et al., 2019) and the full models are fine-tuned at each training iteration. After obtaining new annotations in each AL cycle, we first train a model based on all the available full or partial annotations. When using self-training, we further apply this newly trained model to assign pseudo soft labels to all unannotated instances and combine them with the existing annotations to train another model. Compared to using the old model from the last AL cycle, this strategy can give more accurate pseudo labels since the newly updated model usually performs better by learning from more annotations. For PA, pseudo soft labels are assigned to both un-selected sentences and the unannotated sub-structures in the selected sentences.

9.3.2 Comparison Scheme

Since FA and PA annotate at different granularities, we need a common cost measurement to compare their effectiveness properly. A reasonable metric is the number of the labeled sub-structures; for instance, the number of labeled tokens for sequence labeling or edges for dependency parsing. This metric is commonly adopted in previous PA work (Tomanek and Hahn, 2009a; Flannery and Mori, 2015; Li et al., 2016; Radmard et al., 2021).

Nevertheless, evaluating only by sub-structures ignores a crucial hidden cost: The reading time of the contexts. For example, in sequence labeling with PA, although not every token in the sentence needs to be tagged, the annotator may still need to read the whole sentence to understand its meaning. Therefore, if performing comparisons only by the amount of annotated sub-structures, it will be unfair for the FA baseline because more contexts must be read to carry out PA.

In this chapter, we adopt a simple two-facet comparison scheme that considers both reading and labeling costs. We first control the reading cost by choosing the same size of contexts in the sentence selection step of each AL cycle (Line 4 in Algorithm 2). Then, we further compare the sub-structure labeling cost, measured by the sub-structure annotation cost. If PA can roughly reach the FA performance with the same reading cost but fewer sub-structures annotated, it would be fair to say that PA can help reduce cost over FA. A better comparison scheme should evaluate against a unified estimation of the real annotation costs (Settles et al., 2008). This usually requires actual annotation exercises rather than simulations, which we leave to future work.

9.3.3 NER and DPAR

Settings. We compare primarily three strategies: FA, PA, and a baseline where randomly selected sentences are fully annotated (Rand). We also include a supervised result (Super.) which is obtained from a model trained with the full original training set. We measure reading cost by

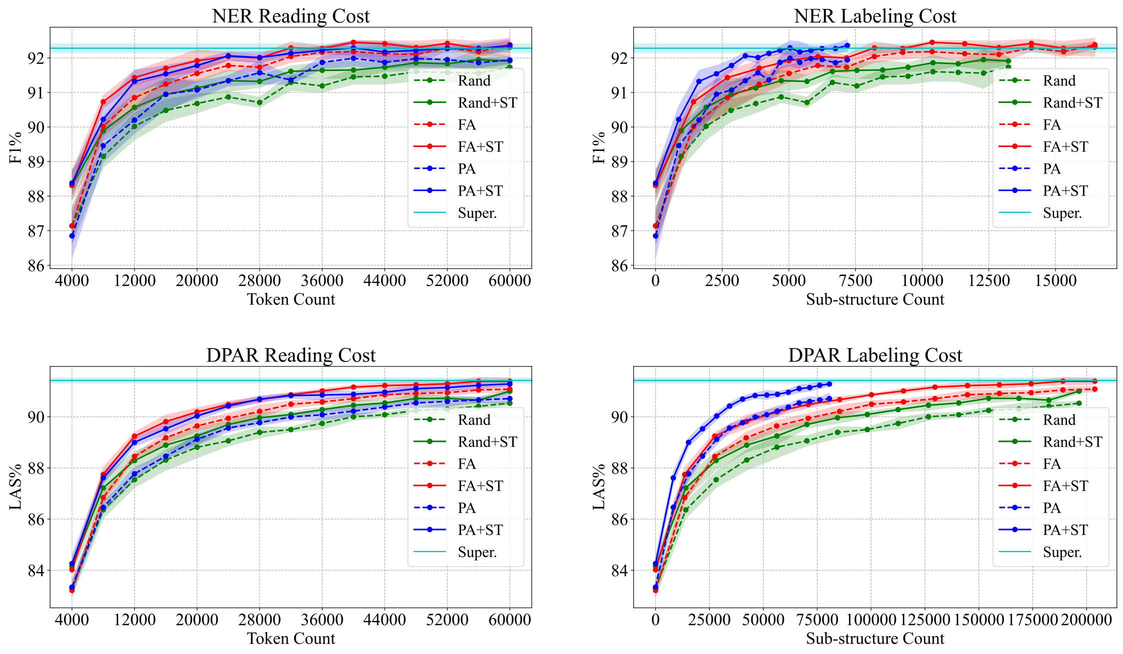


Figure 9.2: Comparisons according to reading and labeling cost.

the total number of tokens in the selected sentences. For labeling cost, we further adopt metrics with practical considerations. In NER, lots of tokens, such as functional words, can be easily judged as the 'O' (non-entity) tag. To avoid overestimating the costs of such easy tokens for FA, we filter tokens by their part-of-speech (POS) tags and only count the ones that are likely to be inside an entity mention.[4] For PA, we still count every queried token. For the task of DPAR, similarly, different dependency links can have variant annotation difficulties. We utilize the surface distance between the head and modifier of the dependency edge as the measure of labeling cost, considering that the decision of longer dependencies is usually harder.

Main Results. The main test results are shown in Figure 9.2, where the patterns on both tasks are similar. First, AL brings clear improvements over the random baseline and can roughly reach the fully supervised performance with only a small portion of data annotated (around 18% for CoNLL-2003 and 30% for UD-EWT). Moreover, self-training (+ST) is helpful for all the strategies, boosting performance without the need for extra manual annotations. Finally, with the help of self-training, the PA strategy can roughly match the performance of FA with the same amount of reading cost (according to the left figures) while labeling fewer sub-structures (according to the right figures). This indicates that PA can help to further reduce annotation costs over the strong FA baselines.

[4]The POS tags are assigned by Stanza (Qi et al., 2020). For CoNLL-2003, we filter by PROPN and ADJ, which cover more than 95% of the entity tokens.

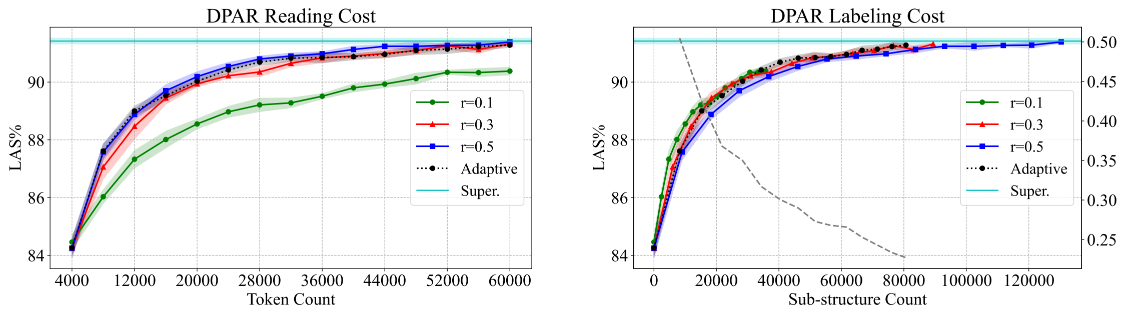


Figure 9.3: Comparisons of different strategies to decide the partial ratio.

Ratio Analysis. We further analyze the effectiveness of our adaptive ratio scheme with DPAR as the case study. We compare the adaptive scheme to schemes with fixed ratio r, and the results[5] are shown in Figure 9.3. For the fixed-ratio schemes, if the value is too small (such as 0.1), although its improving speed is the fastest at the beginning, its performance lags behind others with the same reading contexts due to fewer sub-structures annotated. If the value is too large (such as 0.5), it grows slowly, probably because too many uninformative sub-structures are annotated. The fixed scheme with $r = 0.3$ seems a good choice; however, it is unclear how to find this sweet spot since we usually do not wish to tune such hyperparameters in realistic AL processes. The adaptive scheme provides a reasonable solution by automatically deciding the ratio according to the model performance.

Error and Uncertainty Analysis. We further analyze the error rates and uncertainties of the queried sub-structures. We still take DPAR as a case study and Figure 9.4 shows the results along the AL cycles in PA mode. Here, 'pred' denotes the predicted error rate, 'error' denotes the actual error rate, and 'margin' denotes the uncertainty (margin) scores. For the suffixes, '(S)' indicates partially selected sub-structures, and '(N)' indicates non-selected ones. 'Margin(N)' is omitted since it is always close to 1. First, though adopting a simple model, the performance predictor can give reasonable estimations for the overall error rates. Moreover, by further breaking down the error rates into selected (S) and non-selected (N) groups, we can see that the selected ones contain many errors, indicating the need for manual corrections. On the other hand, the error rates on the non-selected sub-structures are much lower, verifying the effectiveness of using model-predicted pseudo labels on them in self-training. Finally, the overall margin of the selected sentences keeps increasing towards 1, indicating that there are many non-ambiguous sub-structures even in highly-uncertain sentences. The margins of the selected sub-structures are much lower, suggesting that annotating them could provide more informative signals for model training.

Domain-transfer Experiments. Previous experiments adopt a setting where we assume a single data resource and start AL on it from scratch. In practice, there may be existing related data resources that can be helpful by using transfer learning. We investigate a domain-transfer sce-

[5]Here, we use self-training (+ST) for all the strategies.

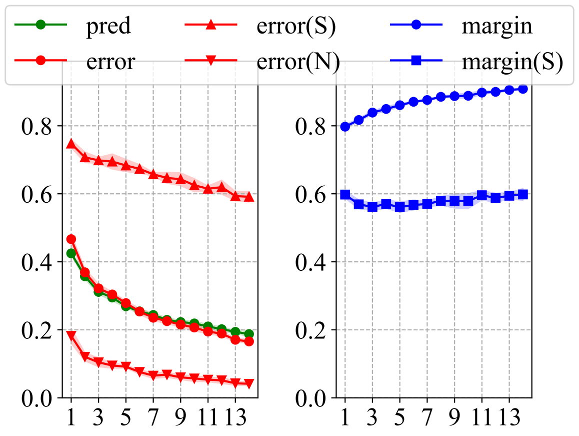


Figure 9.4: Analyses of error rates and uncertainties (margins) of the DPAR sub-structures in the queried sentences along the AL cycles (x-axis).

nario: in addition to unlabeled in-domain data, we assume abundant out-of-domain annotated data and perform AL on the target domain. We adopt tweet texts as the target domain, using Broad Twitter Corpus (BTC; Derczynski et al., 2016) for NER and Tweebank (Liu et al., 2018d) for DPAR. We assume we have models trained from a richly-annotated source domain and continue performing AL on the target domain. The source domains are the datasets that we utilize in our main experiments: CoNLL03 for NER and UD-EWT for DPAR. We adopt a simple model-transfer approach by initializing the model from the one trained with the source data and further fine-tuning it with the target data. Since the target data size is small, we reduce the AL batch sizes for BTC and Tweebank to 2000 and 1000 tokens, respectively. The results for these experiments are shown in Figure 9.5. In these experiments, we also include the no-transfer results, adopting the "FA+ST" but without model transfer. For NER, without transfer learning, the results are generally worse, especially in early AL stages, where there is a small amount of annotated data to provide training signals. In these cases, knowledge learned from the source domain can provide extra information to boost the results. For DPAR, we can see even larger benefits of using transfer learning; there are still clear gaps between transfer and no-transfer strategies when the former already reaches the supervised performance. These results indicate that the benefits of AL and transfer learning can be orthogonal, and combining them can lead to promising results.

9.3.4 Information Extraction

We further explore more complex IE tasks that involve multiple types of output. Specifically, we investigate event extraction and relation extraction. We adopt a classical pipelined approach, which splits the full task into two sub-tasks: the first performs mention extraction, while the second examines mention pairs and predicts relations. While previous work has investigated multi-task AL with FA (Reichart et al., 2008; Zhu et al., 2020; Rotman and Reichart, 2022), this chapter is the first to explore PA in this challenging setting.

We extend our PA scheme to this multi-task scenario with several modifications. First, for the sentence-selection stage, we obtain a sentence-wise uncertainty score UNC(x) with a weighted

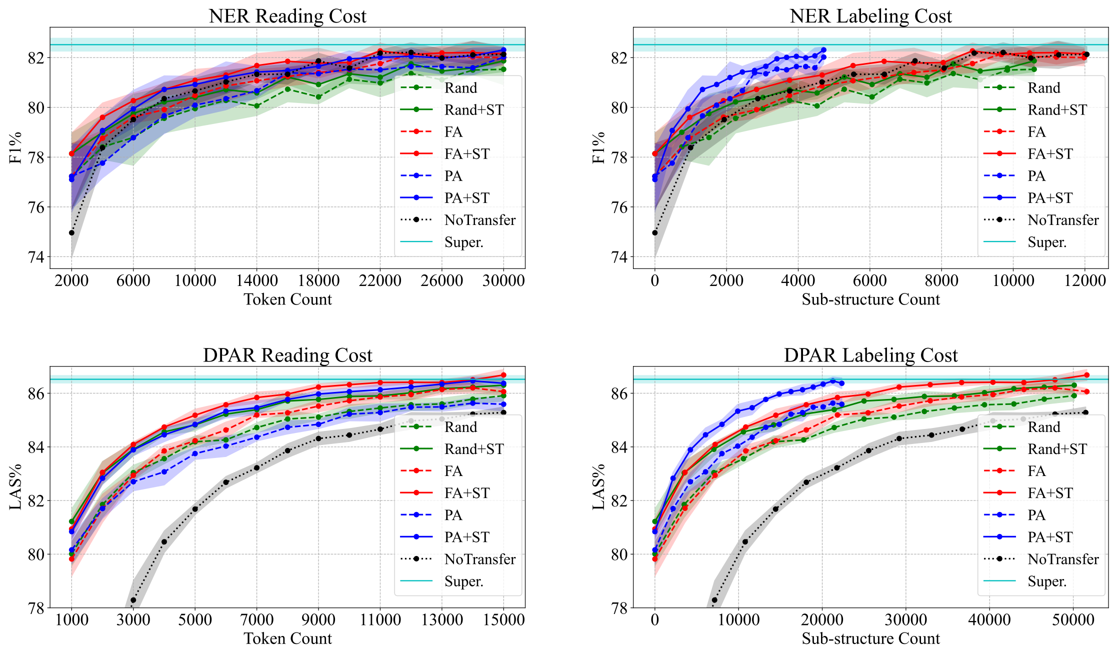


Figure 9.5: AL results in domain-transfer settings (CoNLL03 $\rightarrow$ BTC for NER and UD-EWT $\rightarrow$ Tweebank for DPAR).

combination of the two sub-tasks' uncertainty scores:

$$\begin{aligned}\text{UNC}(x) = \beta \cdot \text{UNC-Mention}(x) \\ + (1-\beta) \cdot \text{UNC-Relation}(x)\end{aligned}$$

Following Rotman and Reichart (2022), we set β to a relatively large value (0.9), which is found to be helpful for the second relational sub-task.

Moreover, for partial selection, we separately select sub-structures for the two sub-tasks according to the adaptive selection scheme. Since the second relational sub-task depends on the mentions extracted from the first sub-task, we utilize predicted mentions and view each feasible mention pair as a querying candidate. A special annotation protocol is adopted to deal with the incorrectly predicted mentions. For each queried relation, we first examine its mentions and perform corrections if there are mention errors that can be fixed by matching the gold ones. If neither of the two mentions can be corrected, we discard[6] this query.

Finally, to compensate for the influences of errors in mention extraction, we adopt further heuristics of increasing the partial ratio by the estimated percentage of queries with incorrect mentions, as well as including a second annotation stage with queries over newly annotated mentions.

We show the results in Figure 9.6, where the overall trends are similar to those in NER and DPAR. Here, labeling cost is simply measured as the number of candidate argument/relation links. Overall, self-training is helpful for all AL strategies, indicating the benefits of making

[6]In preliminary experiments, we also tried including such items as negative examples but did not find benefits.

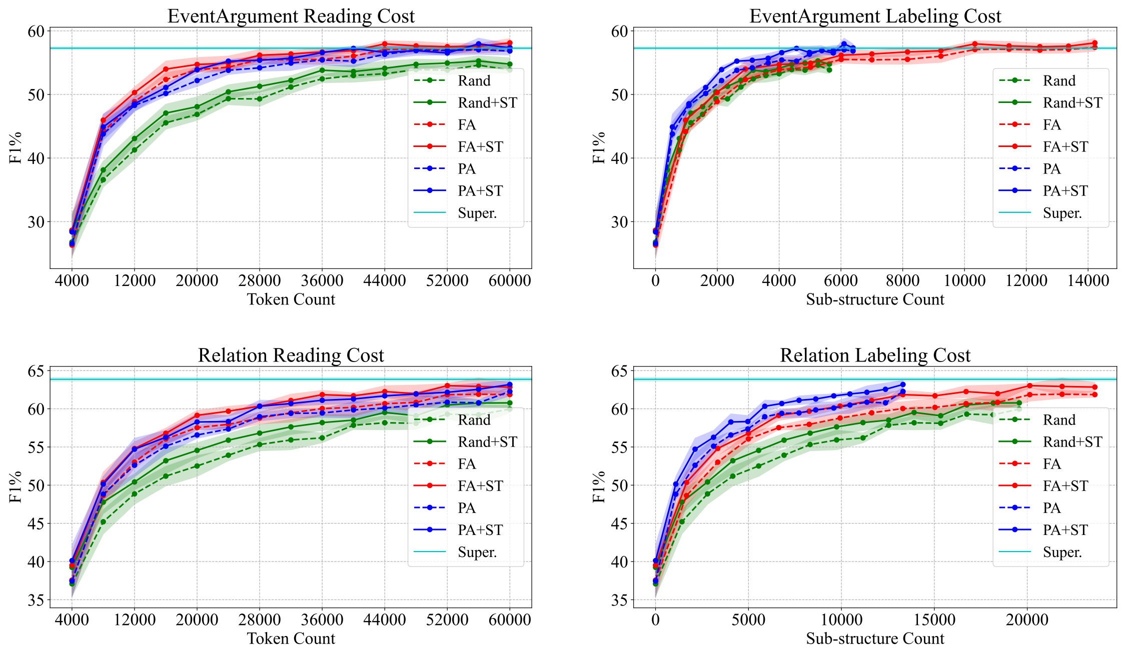


Figure 9.6: Results of event argument extraction and relation extraction on ACE05.

better use of unlabeled data. If measured by the labeling cost, PA learns the fastest and costs only around half of the annotated arguments of FA to reach the supervised result. On the other hand, PA is also competitive concerning reading cost and can generally match the FA results in later AL stages.

9.4 Explorations on the Utilization of Syntactic Information

In previous sections, we mainly treat structures as the predicting targets. It will be interesting to explore whether structures, such as syntactic information, can be utilized to further help downstream tasks. In this section, we conduct two preliminary experiments in such settings, exploring the helpfulness of syntactic path information for the target task of event argument extraction.

9.4.1 AL selection with Syntactic Diversity

In our main experiments, we perform AL selection based on uncertainty, which can help to choose the instances where the model is most ambiguous. To allow practical model learning, we usually adopt **batch-mode AL** (Settles, 2009), where each time a batch of instances is selected. In this case, there can still be informative but redundant instances inside one selected batch. Therefore, one promising direction for further improvements is to consider the diversity criterion to mitigate redundancies.

We explore diversity-enhanced selection for the task of event argument extraction (on ACE05) with a simplified setting. Specifically, we assume given trigger and entity mentions and adopt a fixed batch of 100 trigger-argument pairs in each AL cycle. These simplifications are adopted to more clearly show the effects of the selecting criteria, excluding influences by other factors such as error propagation brought by the automatic mention extraction process.

In this experiment, we adopt a simple signature-based diversity-enhanced selection procedure. First, for each trigger-argument candidate pair, we assign it a signature. Then, we rank all the candidates according to their uncertainty scores and perform selection according to this order until running out of the overall budget. The main difference from plain uncertainty selection is that we further specify a signature-based budget: at each selection step, we ignore the current instance if there have been enough previously-selected instances that have the same signature. In this way, we can obtain a set of informative instances that are also diverse according to the signatures.

The main remaining question is how to specify the signature for the candidate instances. In this experiment, we explore two types of signatures: syntactic path and event type. The former captures syntactic information with the labeled dependency path between the event trigger and the argument, while the latter captures semantic information for the event frame. For example, considering a trigger-argument pair "went"-"store" in the following sentence: "He left the *store* in the downtown area and **went** back home." Here, the syntactic path signature is "[↑conj, ↓obj]", that is, "went" is coordinated by "left", which further governs the object "store". The event type feature is TRANSPORT triggered by the predicate "went". Notice that these two types of signatures capture different aspects of the argument candidate: the syntactic one reflects the surface form structure, which can usually constrain the range of the possible argument roles, while the event type contains semantic information which can provide important hints and constraints, directly specifying the target roles for the event frames. Considering the orthogonality of the information that they can provide, we further adopt a combined signature of the syntactic path and event type to capture more information.

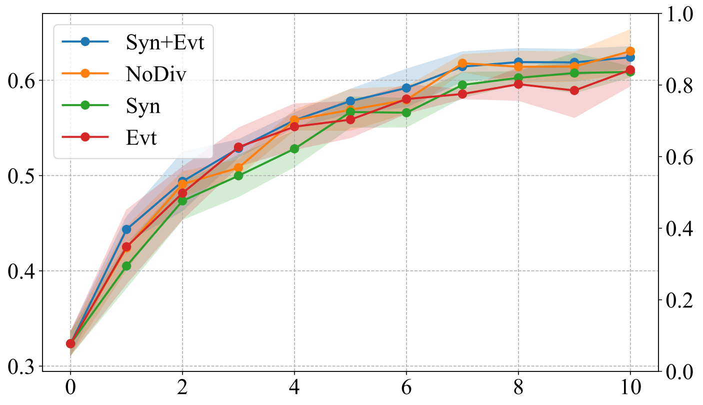


Figure 9.7: AL results (F1%) with different selecting strategies.

The main results are shown in Figure 9.7. The baseline without any diversity enhancement ("NoDiv") turns out to be a strong selecting strategy for this task. Encouraging diversity by using only one type of information, either syntactic path ("Syn") or event type ("Evt"), performs worse than the uncertainty baseline. This indicates that for event argument extraction, uncertainty is still the most important criterion to consider for AL instance selection. By combining syntactic information and event type, we can mitigate the gaps and obtain some slight improvements, especially in the early AL iterations.

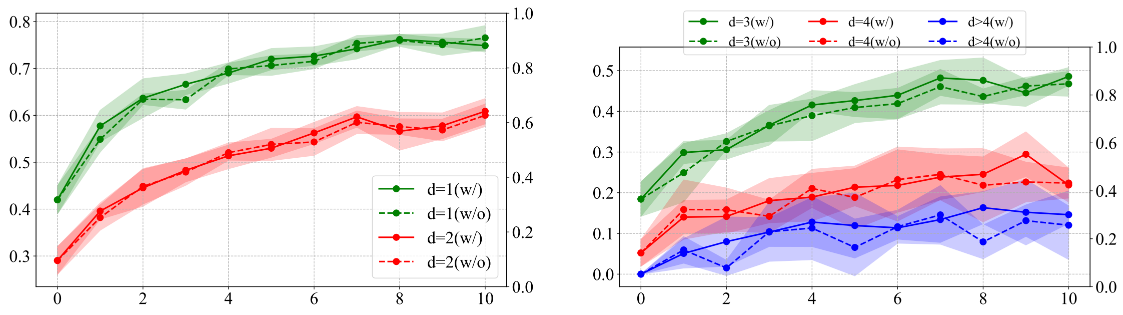


Figure 9.8: Performance breakdowns on trigger-argument syntactic distances.

To further investigate the differences between strategies with (w/) or without (w/o) diversity enhancement, we provide a breakdown analysis on the syntactic distance between the trigger and the argument. The results are shown in Figure 9.8. Here, we adopt the “Syn+Evt” signature for the diversity-enhanced strategy because of its better performance. The results show that for the pairs with short syntactic distances (equaling one or two as shown in the left sub-figure), the two strategies perform nearly the same. This also explains the similarity of their overall performance since most of the valid event arguments are short-distant ones. If further checking the longer-distant ones (larger than two in the right sub-figure), we can see that the diversity-enhanced strategy can obtain slightly better and more consistent results.

9.4.2 Prompt Formulation with Syntactic Information

Recently, large language models (LLMs) have shown great capabilities in solving a wide range of NLP tasks using simple prompting methods. In particular, LLMs show abilities of in-context learning (Brown et al., 2020; Dong et al., 2022), that is, learning to map inputs to outputs with a few in-context demonstration examples without any parameter updating. More specifically, for a classification task that predicts a label y from an input x, we assume that we have a demonstration set C, which contains a few pairs of (x, y) examples. This set can be viewed as the training set in the traditional supervised learning paradigm. Nevertheless, in in-context learning, rather than tuning model parameters with the demonstration examples, we simply form a prompt using the labeled examples and the querying instance and directly feed the prompt to the LM to select the highest-scored target y. Although clean and straightforward, this prompting-based method relies much on the prompts and the performance can be greatly influenced by the prompts' formats (Zhao et al., 2021b; Min et al., 2022b). Therefore, prompt formulation is a key designing factor for the effectiveness of in-context learning.

In this experiment, we explore how to form better prompts for an event argument classification task by using syntactic information. We focus on simpler classification-based tasks since full extraction is still a challenging task for in-context learning (Jimenez Gutierrez et al., 2022). We consider the top-three frequent events from ACE05: ATTACK, TRANSPORT, DIE. For each event type, we further take the top-four frequent roles as the role prediction targets: {Attacker, Target, Instrument, Place} for ATTACK, {Artifact, Vehicle, Origin, Destination} for TRANSPORT, {Agent, Victim, Place, Instrument} for DIE. Specifically, the input x for this task is a context sentence together with a pair of an event trigger and an entity mention, while the output y is the corresponding argument role of the entity mention.

Different from plain classification tasks, here, the input x contains extra specific structures, that is, there are marked spans of trigger and entity mentions that we need to encode in the prompts. We adopt a simple prompt template to include such information: ROLE is the role that the entity "ENTITY" plays in the event "EVENT" for the following sentence: SENT. Here, the slots of ROLE, ENTITY, EVENT and SENT can be filled in by the actual information of the instance. Taking our previous example again, for the instance of "He left the *store* in the downtown area and **went** back home." (here "store" is the ORIGIN in the TRANSPORT event triggered by "went"), the filled template will be: "Origin is the role that the entity "store" plays in the event "went" for the following sentence: He left the store in the downtown area and went back home." Our template starts with the target roles since we adopt the channel scheme (Min et al., 2022a), which estimates $p_{LM}(x|y)$ instead of directly $p_{LM}(y|x)$. In preliminary experiments, we consistently found better and more stable results by adopting the channel prompting method.

In this sub-section, we explore how to better form the prompts for in-context learning on the argument classification tasks. We mainly investigate two aspects: 1) demonstration selection, which explores how to dynamically select demonstration examples according to the querying instance; 2) context representation, which explores how to better represent the input context for the target task. Although the experiments in this sub-section do not directly involve AL, there are close connections between the two explored aspects and AL, with inspirations from representative-based AL strategies:

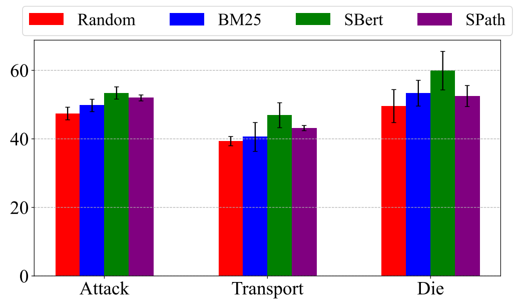
- **Demonstration Selection.** Due to the sequence length limitation of LMs, we can only put a few demonstrations in the in-context learning prompts. In many cases, we may have a training set that contains more instances than the LM context limitations; therefore, we need to perform instance selections to formulate the prompts. Liu et al. (2022a) show that selecting demonstration examples that are similar to the querying instance can bring improvements. Specifically, for each testing example, we use a similarity measurement to search its k-nearest neighbors in the demonstration pool. In this approach, the key factor is the similarity metric, for which we explore four methods in this experiment, including Random, BM25 (Robertson et al., 2009), sentence-bert (SBert) (Reimers and Gurevych, 2019) and syntactic path (SPath). For SBert, we simply adopt cosine similarity on sentence embeddings, while for SPath, we utilize the negative of the edit distance between the trigger-entity syntactic paths (the same as the ones specified in the previous experiment).
- **Context Representations.** Another interesting aspect is how to represent the input context, especially considering that the inputs consist of two anchoring mentions in addition to the context sentence. In this experiment, we explore various ways to represent the input contexts, which are illustrated in Table 9.1 with our previous example sentence. Here, "Full" denotes the original full sentence, "Mentions" only includes words in the event trigger and the argument entity, and "Between" further includes all the words between the two mentions. We further explore two syntax-based methods: "SPath" indicates that we only

include the words on the syntactic path between the trigger and the argument; to make the context representations more familiar to LMs, "SPath+" further adds the functional words that are dependants of the "SPath" words.

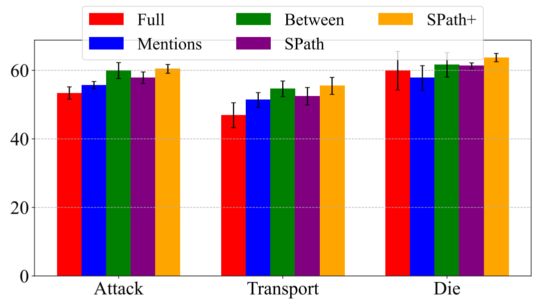
Method	Text
Full	He *left* the store in the downtown area and **went** back home.
Mentions	*left* **went**
Between	*left* the store in the downtown area and **went**
SPath	*left* store **went**
SPath+	*left* the store and **went**

Table 9.1: Illustrations of the example sentence for context representations.

Main Results. The main results are shown in Figure 9.9. For these results, we utilize gpt2-xl as the underlying LM and set the demonstration pool size and in-context example number to 100 and 16, respectively. We randomly select the demonstration pool from the original ACE05 training set and test on all corresponding instances from the test set. For the evaluation metric, we report the Macro F1 scores over the argument roles since role distributions are skewed. All the results are averaged over five runs with different random seeds. The left figure first shows the effects of different demonstration selection methods. Compared with the random baseline, all similarity-based methods can bring improvements. The syntax-based strategy can obtain similar results to the BM25-based retrieval method, while "SBert" obtains the best results, probably because the neural sentence representations can capture more semantic meanings. In the right figure, we further show the influences of using different context representations (using the "SBert"-based selection method). Compared with the full context baseline, other methods with more concise representations can bring benefits. The reason might be that the more concise representations can better present the models the positions of the trigger and entity mentions. Encouragingly, the syntax-based "SPath+" method obtains the overall best results, showing that syntax can help to better represent the essential information for this task.



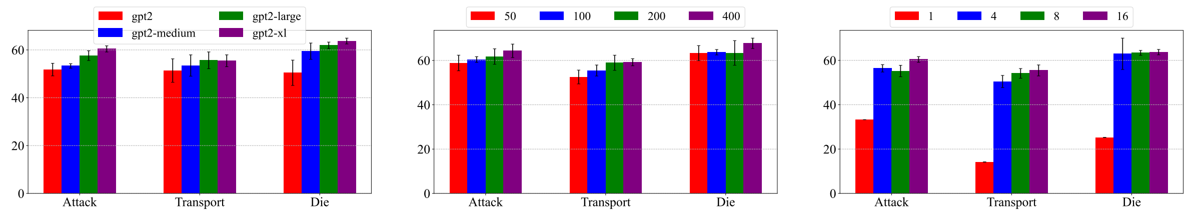
(a) Demonstration Selection



(b) Context Representations

Figure 9.9: In-context learning results (F1%) with different methods.

More Results. Figure 9.10 shows more results with variations in the underlying model, pool size, and the number of demonstration examples. Overall, there are not too many surprises. We can generally observe better results with larger models, larger pool sizes, and more demonstration examples. Nevertheless, there seem to be diminishing benefits of further scaling after certain thresholds. For example, on the demonstration number, results with 4 examples can be very close results to those with more examples, while further additions are restricted by the maximum sequence length that the underlying LM can accept. This is in contrast to the common phenomenon that in low-resource scenarios more training data usually bring obviously better results. It will be interesting to explore how to bring more improvements for in-context learning with the scaling of the number of training instances, where performing inference with k-nearest neighbors retrieval can be a promising direction (Xu et al., 2023).



(a) Model (b) Pool Size (c) Demonstration Number

Figure 9.10: More results with variations on model, pool size, and demonstration number.

9.5 Related Work

Self-training. Self-training is a commonly utilized semi-supervised method to incorporate unlabeled data. It has been shown effective for a variety of NLP tasks, including word sense disambiguation (Yarowsky, 1995), parsing (McClosky et al., 2006), named entity recognition (Meng et al., 2021; Huang et al., 2021), text generation (He et al., 2020) as well as natural language understanding (Du et al., 2021a). Moreover, self-training can be especially helpful for low-resource scenarios, such as in few-shot learning (Vu et al., 2021; Chen et al., 2021). Notice that the early stages of AL also correspond to such scenarios, where self-training can be especially effective.

PA. Learning from incomplete annotations has been well-explored for structured prediction. For CRF models, taking the marginal likelihood as the objective function has been one of the most utilized techniques (Tsuboi et al., 2008; Täckström et al., 2013a; Yang and Vozila, 2014; Greenberg et al., 2018). There are also other methods to deal with incomplete annotations, such as adopting local models (Neubig and Mori, 2010; Flannery et al., 2011), max-margin objective (Fernandes and Brefeld, 2011), learning with constraints (Ning et al., 2018, 2019; Mayhew et al., 2019) and negative sampling (Li et al., 2022).

AL for structured prediction. AL has been investigated for various structured prediction tasks in NLP, such as sequence labeling (Settles and Craven, 2008; Shen et al., 2018), parsing (Hwa, 2004), semantic role labeling (Wang et al., 2017; Myers and Palmer, 2021) and machine translation (Haffari et al., 2009; Zeng et al., 2019). While most previous work adopts FA, that is, annotating full structured objects for the inputs, PA can help to further reduce the annotation cost. Typical examples of PA sub-structures include tokens and subsequences for tagging (Marcheggiani and Artières, 2014; Chaudhary et al., 2019; Radmard et al., 2021), word-wise head edges for dependency parsing (Flannery and Mori, 2015; Li et al., 2016) and mention links for coreference resolution (Li et al., 2020a; Espeland et al., 2020).

9.6 Conclusion

In this chapter, we investigate better AL strategies for structured prediction problems in NLP. We adopt a performance estimator to automatically decide suitable ratios for partial sub-structure selection. We further utilize self-training to make better use of the available unlabeled data pool. With comprehensive experiments on various tasks, we show that the combination of PA and self-training can be more data-efficient than strong full AL baselines.

Chapter 10

Conclusions and Future Directions

In this thesis, we investigate three directions to tackle the challenge of data resource limitation for structured prediction in NLP. In Part I, we investigate the effectiveness of structured output modeling and plan to further study its interactions with the amount of data resources, especially in resource-limited scenarios. Moreover, in Part II, we study transfer learning with related data resources from other languages (cross-lingual learning) and related tasks (multi-task learning). We show that proper utilization of these related data resources could lead to performance gains with proper model designs. Finally, in Part III, we plan to explore more efficient ways for data creation with active learning. The implementations of all these studies are and will be publicly available on `https://github.com/zzsfornlp/zmsp`. We hope that our explorations could shed some light on the line of research toward building more effective and data-efficient structured predictors. In addition to the topics discussed in this thesis, there are various *future directions* that would be interesting to explore.

More complex tasks. In this thesis, we are mainly investigating traditional structured prediction tasks such as parsing and semantic role labeling, while it would be interesting to explore more complex ones. Especially, text generation tasks, such as machine translation, paraphrasing, dialogue generation, and so on, are complex and important structured prediction applications. Although recent work usually addresses generation tasks using sequence-to-sequence models with auto-regressive local normalizations, it will still be important to investigate more on the output structures for generation (Ranzato et al., 2015; Wiseman and Rush, 2016; Edunov et al., 2018). Moreover, data availability is also a bottleneck for these tasks, and our studies may still provide helpful methods to reduce the required annotations to achieve reasonable performance.

Modeling input structures. For NLP tasks, the inputs usually consist of sequences of tokens, which are highly structured. In addition to explicit output modeling for structured prediction tasks, how to better capture the structures in inputs may be an interesting question for a larger range of NLP tasks. Recent neural-network-based models, especially the pre-trained ones, are good at capturing inter-dependencies in the inputs. It would be interesting to explore how to further enhance the input modeling, where one potential direction would be the infusion of explicit structural information (Zhou et al., 2020; Bai et al., 2021; Li et al., 2021b; Wu et al., 2021).

Model Efficiency. While recent neural models have shown strong performance, it usually brings larger computational and thus environmental costs (Strubell et al., 2019), which calls for research on more efficient models (Menghani, 2021). While there are various ways to improve the efficiency of NLP models, considering structures might be a potential direction. In some way, structured pruning can be an example, which removes groups of consecutive parameters and has been shown effective to obtain compact NLP models (Wang et al., 2020b; Xia et al., 2022).

Model Interpretability. In addition, the neural models have been questioned for their opaque nature (Lipton, 2018). While traditional feature-based linear models could be straightforward to interpret, neural models hide their reasoning process inside the hidden layer computations. Therefore, recent work has been devoted to the analysis and interpretation of neural models (Belinkov and Glass, 2019; Belinkov et al., 2020). Interpretability can have interesting connections to language structures. For example, predicting linguistic structures have been utilized as target tasks to understand neural models, where probing is a typical example to query structured linguistic information encoded in the neural representations (Conneau et al., 2018; Tenney et al., 2019; Hewitt and Manning, 2019). In addition, it would be interesting to explore whether models that consider more explicit structures can lead to better interpretability.

Better Data Usage. In addition to transfer learning and active learning, there are other ways to make better use of the available data resources, such as data augmentation by creating synthesis data instances (Feng et al., 2021) or semi-supervised learning by incorporating useful information from unlabeled data (Zhu, 2005). Exploring how to better combine these techniques would be an interesting direction. Moreover, a better understanding of the data will also be a helpful way to enhance model learning. For example, data maps derived from training dynamics can be utilized to identify ambiguous data (Swayamdipta et al., 2020). An interesting direction will be exploring the usage of data maps with unlabeled data. One challenge is that the creation of data maps requires target labels, where using self-training-styled pseudo labels can be a feasible option. This may be especially helpful in limited-resource scenarios, where better data understanding can lead to effective ways to find a minimum collection of required resources to improve model performance.

Representation Learning. Neural models capture meaning with implicit representations, while the inputs and outputs of NLP tasks as well as most linguistic representations, such as labels and trees, are explicitly represented as discrete items. Better connections between these two types of representations can bring further benefits. One promising recent research direction to better align them is prompting (Liu et al., 2023), which formalizes the target task as a template-filling problem and solves it with pre-trained language models. My previous work (Zhang et al., 2022f) also uses this idea by forming argument queries with templates. One open question for this direction is how to design better templates in an automatic way, where analyzing the influences of the templates' structural properties will be an interesting idea. Another exciting research direction to connect implicit and explicit representations is to inject knowledge into neural models with external memory networks (Lewis et al., 2020b). Previous researches usually store all the instances in the memory in a flattened way, while exploring the structured organization of the memory storage will be helpful for more effective and efficient instance querying.

Connections to Large Language Models. Recently, the field of NLP has been revolutionized by the development of large language models (LLMs) (Brown et al., 2020; Chowdhery et al., 2022; Scao et al., 2022; Zhang et al., 2022d; OpenAI, 2023; Touvron et al., 2023). Various studies have shown that LLMs have the capability to handle a wide range of NLP tasks in a prompting and text generation way (Jiao et al., 2023; Qin et al., 2023; Bang et al., 2023; Wei et al., 2023; Bubeck et al., 2023). Moreover, the scaling of LLMs has surprisingly resulted in new emergent capabilities (Wei et al., 2022a), such as in-context learning (Brown et al., 2020; Dong et al., 2022), instruction following (Ouyang et al., 2022) and multi-step reasoning (Wei et al., 2022b; Qiao et al., 2022). It will be interesting to explore the connections between LLMs and language structures. On the one hand, LLMs can be utilized to build better-structured predictors. Recently, there has been a trend casting NLP tasks as a unified text-to-text transformation problem (Raffel et al., 2020), structured prediction tasks can be also tackled in a similar way (Vinyals et al., 2015b; Paolini et al., 2021; He and Choi, 2023). The main question in this direction is how to represent the target structured output, more specifically, how to linearize the structured outputs in a text sequence format. While the LMs can directly predict a structured format, considering that they are originally trained on language texts, representing structured outputs that are closer to natural languages might reduce the discrepancies between LM training and target-task inference, leading to easier adaptation and better results. The template-based method explored in Chapter 7 can be a promising method for this purpose. On the other hand, it will also be interesting to explore the utilization of language structures for LM prompting. Although LLMs have shown surprising abilities in NLP tasks, they are known to be sensitive to the utilized prompts (Zhao et al., 2021b; Min et al., 2022b). Therefore, how to construct and organize better prompts would be an important point, and exploring the structures of the prompts could be a promising direction to better elicit the full power of LMs. Moreover, LLMs can be good few-shot or zero-shot learners (Brown et al., 2020; Kojima et al., 2022). This is also related to the main topic of this thesis exploring model improvements in resource-limited scenarios, and techniques that are explored in this thesis, such as active learning and self-training, can have interesting interactions with LLMs and may bring further enhancement.

We hope that with explorations in these directions, good structured representations can be obtained without too many manual annotations, and they can lead to further advances in automatic language understanding.

Bibliography

Charu C Aggarwal, Xiangnan Kong, Quanquan Gu, Jiawei Han, and S Yu Philip. Active learning: A survey. In *Data Classification*, pages 599–634. Chapman and Hall/CRC, 2014. 8.1

Željko Agić, Jörg Tiedemann, Danijela Merkler, Simon Krek, Kaja Dobrovoljc, and Sara Može. Cross-lingual dependency parsing of related languages with rich morphosyntactic tagsets. In *Proceedings of the EMNLP'2014 Workshop on Language Technology for Closely Related Languages and Language Variants*, pages 13–24, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4203. URL https://aclanthology.org/W14-4203. 5.5

Roee Aharoni and Yoav Goldberg. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.692. URL https://www.aclweb.org/anthology/2020.acl-main.692. 6.3.5

David Ahn. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia, July 2006. Association for Computational Linguistics. URL https://aclanthology.org/W06-0901. 4.2.3, 7.1

Alfred V. Aho and Jeffrey D. Ullman. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ, 1972. 2.2

Alan Akbik, Laura Chiticariu, Marina Danilevsky, Yunyao Li, Shivakumar Vaithyanathan, and Huaiyu Zhu. Generating high quality proposition Banks for multilingual semantic role labeling. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 397–407, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1039. URL https://www.aclweb.org/anthology/P15-1039. 6.3.1, 6.4

Alan Akbik, Xinyu Guan, and Yunyao Li. Multilingual aliasing for auto-generating proposition Banks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3466–3474, Osaka, Japan, December 2016a. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1327. 6.3.1

Alan Akbik, Vishwajeet Kumar, and Yunyao Li. Towards semi-automatic generation of proposition Banks for low-resource languages. In *Proceedings of the 2016 Conference on Em-*

pirical Methods in Natural Language Processing, pages 993–998, Austin, Texas, November 2016b. Association for Computational Linguistics. doi: 10.18653/v1/D16-1102. URL https://www.aclweb.org/anthology/D16-1102. 6.3.1

Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. Active learning and crowd-sourcing for machine translation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010a. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2010/pdf/244_Paper.pdf. 8.2.2, 8.6

Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. Active learning-based elicitation for semi-supervised word alignment. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 365–370, Uppsala, Sweden, July 2010b. Association for Computational Linguistics. URL https://aclanthology.org/P10-2067. 8.4.2

Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. Active semi-supervised learning for improving word alignment. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 10–17, Los Angeles, California, June 2010c. Association for Computational Linguistics. URL https://aclanthology.org/W10-0102. 8.4.2

Vamshi Ambati, Sanjika Hewavitharana, Stephan Vogel, and Jaime Carbonell. Active learning with multiple annotations for comparable data classification task. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 69–77, Portland, Oregon, June 2011a. Association for Computational Linguistics. URL https://aclanthology.org/W11-1210. 8.6

Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. Multi-strategy approaches to active learning for statistical machine translation. In *Proceedings of Machine Translation Summit XIII: Papers*, Xiamen, China, September 19-23 2011b. URL https://aclanthology.org/2011.mtsummit-papers.12. 8.2.3

Maryam Aminian, Mohammad Sadegh Rasooli, and Mona Diab. Cross-lingual transfer of semantic roles: From raw text to semantic roles. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 200–210, Gothenburg, Sweden, May 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-0417. URL https://www.aclweb.org/anthology/W19-0417. 6.4

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4: 431–444, 2016. doi: 10.1162/tacl_a_00109. URL https://aclanthology.org/Q16-1031. 5.1, 5.5

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1231. URL https://aclanthology.org/P16-1231. 2.2, 3.1

Shilpa Arora, Eric Nyberg, and Carolyn P. Rosé. Estimating annotation cost for active learning in a multi-annotator environment. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 18–26, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1903. 8.3.2

Jordan Ash and Ryan P Adams. On warm-starting neural network training. *Advances in Neural Information Processing Systems*, 33:3884–3894, 2020. 8.3.2

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryghZJBKPS. 8.2.2, 8.2.3

Seyed Arad Ashrafi Asli, Behnam Sabeti, Zahra Majdabadi, Preni Golazizian, Reza Fahmi, and Omid Momenzadeh. Optimizing annotation effort using active learning strategies: A sentiment analysis case study in Persian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2855–2861, Marseille, France, May 2020. European Language Resources Association. URL https://aclanthology.org/2020.lrec-1.348. 8.3.2

Josh Attenberg and Şeyda Ertekin. Class imbalance and active learning. *Imbalanced Learning: Foundations, Algorithms, and Applications*, pages 101–149, 2013. 8.6

Josh Attenberg and Foster Provost. Inactive learning? difficulties employing active learning in practice. *ACM SIGKDD Explorations Newsletter*, 12(2):36–41, 2011. 8.7.2

Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. Zero-resource dependency parsing: Boosting delexicalized cross-lingual transfer with linguistic knowledge. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 119–130, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1012. 5.1, 5.5

Philip Bachman, Alessandro Sordoni, and Adam Trischler. Learning algorithms for active learning. In *International Conference on Machine Learning*, pages 301–310. PMLR, 2017. 8.2.1

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. Syntax-BERT: Improving pre-trained transformers with syntax trees. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3011–3020, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.262. URL https://aclanthology.org/2021.eacl-main.262. 10

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, 1998a. URL https://aclanthology.org/C98-1013. 2.2, 2.3

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada, August 1998b. Association for Computational Linguistics. doi: 10.3115/980845.980860. URL https://aclanthology.org/P98-1013. 7.1, 7.2.2

Gökhan BakIr, Thomas Hofmann, Alexander J Smola, Bernhard Schölkopf, and Ben Taskar. *Predicting structured data*. MIT press, 2007. 1, 2.1

Jason Baldridge and Miles Osborne. Active learning and the total cost of annotation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 9–16, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-3202. 8.3.2, 8.4.1

Jason Baldridge and Alexis Palmer. How well does active learning *actually* work? Time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 296–305, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/D09-1031. 8.3.2, 8.3.2

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*, pages 178–186, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-2322. 2.3

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*, 2023. 1.1, 10

Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, 41(1):164–171, 1970. 1.2, 2.2, 4.4

Garrett Beatty, Ethan Kochis, and Michael Bloodgood. The use of unlabeled data versus labeled data for stopping active learning for text classification. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pages 287–294. IEEE, 2019. 8.5.2

Yonatan Belinkov and James Glass. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019. doi: 10.1162/tacl_a_00254. URL https://aclanthology.org/Q19-1004. 1.1, 10

Yonatan Belinkov, Sebastian Gehrmann, and Ellie Pavlick. Interpretability and analysis in neural NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 1–5, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-tutorials.1. URL https://aclanthology.org/2020.acl-tutorials.1. 1.1, 10

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb):1137–1155, 2003. 1.1

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. 8.7.1

Michael Bloodgood and Chris Callison-Burch. Bucking the trend: Large-scale cost-focused active learning for statistical machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 854–864, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL https://aclanthology.org/P10-1088. 8.2.2, 8.3.1, 8.3.2, 8.5.1, 8.5.2

Michael Bloodgood and John Grothendieck. Analysis of stopping active learning based on stabilizing predictions. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 10–19, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-3502. 8.5.2

Michael Bloodgood and K. Vijay-Shanker. A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 39–47, Boulder, Colorado, June 2009a. Association for Computational Linguistics. URL https://aclanthology.org/W09-1107. 8.5.2

Michael Bloodgood and K. Vijay-Shanker. Taking into account the differences between actively and passively acquired data: The case of active learning with support vector machines for imbalanced datasets. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 137–140, Boulder, Colorado, June 2009b. Association for Computational Linguistics. URL https://aclanthology.org/N09-2035. 8.6

Bernd Bohnet, Ryan McDonald, Emily Pitler, and Ji Ma. Generalized transition-based dependency parsing via control parameters. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 150–160, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1015. URL https://aclanthology.org/P16-1015. 2.2

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: 10.1162/tacl_a_00051. URL https://aclanthology.org/Q17-1010. 1.1, 5.4.1

Claire Bonial, Julia Bonn, Kathryn Conger, Jena D. Hwang, and Martha Palmer. PropBank: Semantics of new predicate types. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3013–3019, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/1012_Paper.pdf. 2.3

Kianté Brantley, Amr Sharaf, and Hal Daumé III. Active imitation learning with noisy guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2093–2105, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.189. URL https://aclanthology.org/2020.acl-main.189. 8.2.1, 8.4.2

Klaus Brinker. Incorporating diversity in active learning with support vector machines. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 59–66,

2003. 8.2.2

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 7.3.2, 9.4.2, 10

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023. 1.1, 10

Sabine Buchholz and Erwin Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, June 2006. Association for Computational Linguistics. URL https://aclanthology.org/W06-2920. 2.3

David Burkett and Dan Klein. Variational inference for structured NLP models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Tutorials)*, pages 9–10, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/P13-5006. 4.4

Jan Buys and Jan A. Botha. Cross-lingual morphological tagging for low-resource languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1954–1964, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1184. URL https://aclanthology.org/P16-1184. 5.5

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2753–2765, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1233. 3.2.3, 6.4

Rui Cai and Mirella Lapata. Syntax-aware semantic role labeling without parsing. *Transactions of the Association for Computational Linguistics*, 7:343–356, March 2019. doi: 10.1162/tacl_a_00272. URL https://www.aclweb.org/anthology/Q19-1022. 6.1, 6.4

Rui Cai and Mirella Lapata. Alignment-free cross-lingual semantic role labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3883–3894, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.319. URL https://www.aclweb.org/anthology/2020.emnlp-main.319. 6.4

Tingting Cai, Yangming Zhou, and Hong Zheng. Cost-quality adaptive active learning for chinese clinical named entity recognition. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 528–533. IEEE, 2020. 8.3.2

Tingting Cai, Zhiyuan Ma, Hong Zheng, and Yangming Zhou. Ne–lp: normalized entropy- and loss prediction-based sampling for active learning in chinese word segmentation on ehrs. *Neural Computing and Applications*, 33(19):12535–12549, 2021. 8.2.1

Hian Cañizares-Díaz, Alejandro Piad-Morffis, Suilan Estevez-Velarde, Yoan Gutiérrez, Yudivián Almeida Cruz, Andres Montoyo, and Rafael Muñoz-Guillena. Active learning for assisted corpus construction: A case study in knowledge discovery from biomedical text. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 216–225, Held Online, September 2021. INCOMA Ltd. URL https://aclanthology.org/2021.ranlp-1.26. 8.3.2

Rémi Cardon, Adrien Bibal, Rodrigo Wilkens, David Alfter, Magali Norré, Adeline Müller, Watrin Patrick, and Thomas François. Linguistic corpus annotation for automatic text simplification evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1842–1866, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.121. 1.1

Xavier Carreras. Experiments with a higher-order projective dependency parser. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 957–961, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1101. 2.2, 3.1.1

Xavier Carreras and Lluís Màrquez. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-2412. 2.3

Xavier Carreras and Lluís Màrquez. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0620. 2.3, 3.2.2

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997. 1.2, 6.1, 6.4

Yee Seng Chan and Hwee Tou Ng. Domain adaptation with active learning for word sense disambiguation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 49–56, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/P07-1007. 8.4.2

Haw-Shiuan Chang, Shankar Vembu, Sunil Mohan, Rheeya Uppaal, and Andrew McCallum. Using error decay prediction to overcome practical issues of deep active learning for named entity recognition. *Machine Learning*, 109(9):1749–1778, 2020. 8.2.1

Ming-Wei Chang, Lev-Arie Ratinov, Nicholas Rizzolo, and Dan Roth. Learning and inference with constraints. In *AAAI*, pages 1513–1518, 2008. 4.4

Aditi Chaudhary, Jiateng Xie, Zaid Sheikh, Graham Neubig, and Jaime Carbonell. A little annotation does a lot of good: A study in bootstrapping low-resource named entity recognizers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5164–5174, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1520. URL https://aclanthology.org/D19-1520. 8.3.1, 8.4.2, 9.1, 9.2.1, 9.5

Aditi Chaudhary, Antonios Anastasopoulos, Zaid Sheikh, and Graham Neubig. Reducing confusion in active learning for part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 9:1–16, 2021. doi: 10.1162/tacl_a_00350. URL https://aclanthology.org/2021.tacl-1.1. 8.4.2

Chenhua Chen, Alexis Palmer, and Caroline Sporleder. Enhancing active learning for semantic role labeling via compressed dependency trees. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 183–191, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I11-1021. 8.2.3

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1082. URL https://aclanthology.org/D14-1082. 1.1, 2.2, 3.1

Desai Chen, Nathan Schneider, Dipanjan Das, and Noah A. Smith. SEMAFOR: Frame argument resolution with log-linear models. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 264–267, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL https://aclanthology.org/S10-1059. 7.4

Yiming Chen, Yan Zhang, Chen Zhang, Grandee Lee, Ran Cheng, and Haizhou Li. Revisiting self-training for few-shot learning of language model. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9125–9135, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.718. URL https://aclanthology.org/2021.emnlp-main.718. 9.5

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China, July 2015a. Association for Computational Linguistics. doi: 10.3115/v1/P15-1017. URL https://aclanthology.org/P15-1017. 7.4

Yukun Chen, Thomas A Lasko, Qiaozhu Mei, Joshua C Denny, and Hua Xu. A study of active learning methods for named entity recognition in clinical text. *Journal of biomedical informatics*, 58:11–18, 2015b. 8.3.2

Yunmo Chen, Tongfei Chen, Seth Ebner, Aaron Steven White, and Benjamin Van Durme. Reading the manual: Event extraction as definition comprehension. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 74–83, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.spnlp-1.9. URL https://aclanthology.org/2020.spnlp-1.9. 7.2.1

Ethan A. Chi, John Hewitt, and Christopher D. Manning. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.493. URL `https://aclanthology.org/2020.acl-main.493`. 1.1

Everlyn Chimoto and Bruce Bassett. COMET-QE and active learning for low-resource machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4735–4740, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.findings-emnlp.348`. 8.2.1

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 10

Y.J. Chu and T.H. Liu. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14: 1396–1400, 1965. 2.2, 3.1.1, 4.4

Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34, 2021. 8.3.2

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020. doi: 10.1162/tacl_a_00317. URL `https://aclanthology.org/2020.tacl-1.30`. 7.3.3

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4828. URL `https://www.aclweb.org/anthology/W19-4828`. 3.2.3

James Clarke and Mirella Lapata. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429, 2008. 4.4

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994. 8.1

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996. 8.1

Michael Collins. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637, 2003. doi: 10.1162/089120103322753356. URL `https://aclanthology.org/J03-4003`. 2.3

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011. 1.1

Simone Conia and Roberto Navigli. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1396–1410, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.120. URL `https://aclanthology.org/2020.coling-main.120`. 6.2.1, 6.4

Simone Conia and Roberto Navigli. Probing for predicate argument structures in pretrained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4622–4632, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.316. URL `https://aclanthology.org/2022.acl-long.316`. 1.1

Simone Conia, Andrea Bacciu, and Roberto Navigli. Unifying cross-lingual semantic role labeling with heterogeneous linguistic resources. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 338–351, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.31. URL `https://aclanthology.org/2021.naacl-main.31`. 6.4

Alexis Conneau, German 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 (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1198. URL `https://aclanthology.org/P18-1198`. 1.1, 10

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*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL `https://www.aclweb.org/anthology/2020.acl-main.747`. 6.1, 6.3.1, 6.4, 7.3.3

Ryan Cotterell and Kevin Duh. Low-resource named entity recognition with cross-lingual, character-level neural conditional random fields. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 91–96, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. URL `https://aclanthology.org/I17-2016`. 5.5

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. A span-extraction dataset for Chinese machine reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5883–5889, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1600. URL `https://aclanthology.org/D19-1600`. 7.3.3

Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *AAAI*, volume 5, pages 746–751, 2005. 8.2.1, 8.3.1, 8.3.2

Aswarth Abhilash Dara, Josef van Genabith, Qun Liu, John Judge, and Antonio Toral. Active learning for post-editing based incrementally retrained MT. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 185–189, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. doi: 10.3115/v1/E14-4036. URL https://aclanthology.org/E14-4036. 8.3.2

Sajib Dasgupta and Vincent Ng. Mine the easy, classify the hard: A semi-supervised approach to automatic sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 701–709, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/P09-1079. 8.4.2, 8.5.1

Sanjoy Dasgupta. Two faces of active learning. *Theoretical computer science*, 412(19):1767–1781, 2011. 8.2.2

Angel Daza and Anette Frank. Translate and label! an encoder-decoder approach for cross-lingual semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 603–615, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1056. URL https://www.aclweb.org/anthology/D19-1056. 6.4

Angel Daza and Anette Frank. X-SRL: A parallel cross-lingual semantic role labeling dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3904–3914, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.321. URL https://www.aclweb.org/anthology/2020.emnlp-main.321. 6.4

Marie-Catherine De Marneffe and Christopher D Manning. Stanford typed dependencies manual. Technical report, Technical report, Stanford University, 2008. 2.3

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, May 2006. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2006/pdf/440_pdf.pdf. 2.3

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/1062_Paper.pdf. 2.3

Yue Deng, KaWai Chen, Yilin Shen, and Hongxia Jin. Adversarial active learning for sequences labeling and generation. In *IJCAI*, pages 4012–4018, 2018. 8.2.2

Pascal Denis and Jason Baldridge. Joint determination of anaphoricity and coreference resolution using integer programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 236–243, Rochester, New York, April 2007. Association for Computational Linguistics. URL https://aclanthology.org/N07-1030. 4.4

Leon Derczynski, Kalina Bontcheva, and Ian Roberts. Broad Twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1111. 9.3.3

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423. 1.1, 1.2, 2.2, 3.2, 3.2.1, 4.1, 6.1, 6.2.3, 6.3.1, 6.3.2, 6.4, 9.1

Dmitriy Dligach and Martha Palmer. Good seed makes a good crop: Accelerating active learning using language modeling. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 6–10, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-2002. 8.5.1

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*, 2022. 9.4.2, 10

Pinar Donmez and Jaime G Carbonell. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 619–628, 2008. 8.3.2

Pinar Donmez, Jaime G Carbonell, and Paul N Bennett. Dual strategy active learning. In *European Conference on Machine Learning*, pages 116–127. Springer, 2007. 8.2.3

David Dowty. Thematic proto-roles and argument selection. *language*, 67(3):547–619, 1991. 2.2

Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. In *ICLR*, 2017. 1.1, 2.2, 3.1, 3.1.1, 3.1.1, 3.1.2, 4.1, 4.2.2, 5.3.1, 5.3.2, 5.4.1, 6.2.2

Timothy Dozat and Christopher D. Manning. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2077. URL https://aclanthology.org/P18-2077. 3.1.1, 6.2.2

Timothy Dozat, Peng Qi, and Christopher D Manning. Stanford's graph-based neural dependency parser at the conll 2017 shared task. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, 2017. 3.1

Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 81–90, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/D09-1009. 8.7.2

Matthew S Dryer. Word order. *Language typology and syntactic description*, 1:61–131, 2007. 1.1, 5.2

Matthew S. Dryer and Martin Haspelmath, editors. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig, 2013. URL https://wals.info/. 5.2

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Veselin Stoyanov, and Alexis Conneau. Self-training improves pre-training for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5408–5418, Online, June 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.426. URL https://aclanthology.org/2021.naacl-main.426. 1.2, 9.2.3, 9.5

Xinya Du and Claire Cardie. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.49. URL https://aclanthology.org/2020.emnlp-main.49. 7.1, 7.2.1, 7.4

Xinya Du, Alexander 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: Main Volume*, pages 634–644, Online, April 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.52. URL https://aclanthology.org/2021.eacl-main.52. 7.2.1, 7.4

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1033. URL https://aclanthology.org/P15-1033. 2.2, 3.1

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.718. URL https://aclanthology.org/2020.acl-main.718. 7.1, 7.2.1, 7.3.3, 7.4

Matthias Eck, Stephan Vogel, and Alex Waibel. Low cost portability for statistical machine translation based on n-gram frequency and TF-IDF. In *Proceedings of the Second International Workshop on Spoken Language Translation*, Pittsburgh, Pennsylvania, USA, October 24-25 2005. URL https://aclanthology.org/2005.iwslt-1.7. 8.2.2

Jack Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards,* *B*, 71:233–240, 1967. 2.2, 3.1.1, 4.4

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 355–364, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1033. URL https://aclanthology.org/N18-1033. 1, 10

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.638. URL https://aclanthology.org/2020.emnlp-main.638. 8.3.2, 8.4.2

Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996. URL https://aclanthology.org/C96-1058. 1.2, 2.2, 2.2, 3.1.1, 3.1.1, 3.1.1, 4.2.2, 5.3.2

Sean P. Engelson and Ido Dagan. Minimizing manual annotation cost in supervised training from corpora. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 319–326, Santa Cruz, California, USA, June 1996. Association for Computational Linguistics. doi: 10.3115/981863.981905. URL https://aclanthology.org/P96-1042. 8.2.1

Alexander Erdmann, David Joseph Wrisley, Benjamin Allen, Christopher Brown, Sophie Cohen-Bodénès, Micha Elsner, Yukun Feng, Brian Joseph, Béatrice Joyeux-Prunel, and Marie-Catherine de Marneffe. Practical, efficient, and customizable active learning for named entity recognition in the digital humanities. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2223–2234, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1231. URL https://aclanthology.org/N19-1231. 8.2.2

Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 127–136, 2007. 8.3.2, 8.5.2, 8.6

Nuno Escudeiro and Alípio Jorge. D-confidence: An active learning strategy which efficiently identifies small classes. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 18–26, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/W10-0103. 8.6

Vebjørn Espeland, Beatrice Alex, and Benjamin Bach. Enhanced labelling in active learning for coreference resolution. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 111–121, Barcelona, Spain (online), December

2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.crac-1.12. 8.3.1, 9.5

Agnieszka Falenska and Jonas Kuhn. The (non-)utility of structural features in BiLSTM-based dependency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 117–128, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1012. URL https://aclanthology.org/P19-1012. 3.1.3

Meng Fang and Trevor Cohn. Model transfer for tagging low-resource languages using a bilingual dictionary. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 587–593, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-2093. URL https://aclanthology.org/P17-2093. 8.4.2

Meng Fang, Jie Yin, and Dacheng Tao. Active learning for crowdsourcing using knowledge transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014. 8.6

Meng Fang, Yuan Li, and Trevor Cohn. Learning how to active learn: A deep reinforcement learning approach. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1063. URL https://aclanthology.org/D17-1063. 8.2.1, 8.4.2

Hao Fei, Meishan Zhang, and Donghong Ji. Cross-lingual semantic role labeling with high-quality translated training corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7026, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.627. URL https://www.aclweb.org/anthology/2020.acl-main.627. 6.3.2, 6.3.2, 6.4

Hao Fei, Meishan Zhang, Fei Li, and Donghong Ji. Cross-lingual semantic role labeling with model transfer. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28: 2427–2437, 2020b. 6.1, 6.4

Hao Fei, Shengqiong Wu, Yafeng Ren, Fei Li, and Donghong Ji. Better combine them together! integrating syntactic constituency and dependency representations for semantic role labeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 549–559, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.49. URL https://aclanthology.org/2021.findings-acl.49. 6.4

Parvin Sadat Feizabadi and Sebastian Padó. Combining seemingly incompatible corpora for implicit semantic role labeling. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 40–50, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.18653/v1/S15-1005. URL https://aclanthology.org/S15-1005. 7.4

Rui Feng, Jie Yuan, and Chao Zhang. Probing and fine-tuning reading comprehension models for few-shot event extraction. *arXiv preprint arXiv:2010.11325*, 2020. 7.1, 7.2.1, 7.4

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.84. URL https://aclanthology.org/2021.findings-acl.84. 7.2.3, 10

Eraldo R Fernandes and Ulf Brefeld. Learning from partially annotated sequences. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 407–422. Springer, 2011. 9.5

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1191. URL https://aclanthology.org/P18-1191. 7.3.1

Daniel Flannery and Shinsuke Mori. Combining active learning and partial annotation for domain adaptation of a Japanese dependency parser. In *Proceedings of the 14th International Conference on Parsing Technologies*, pages 11–19, Bilbao, Spain, July 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-2202. URL https://aclanthology.org/W15-2202. 8.3.1, 8.3.2, 9.1, 9.2.1, 9.3.2, 9.5

Daniel Flannery, Yusuke Miayo, Graham Neubig, and Shinsuke Mori. Training dependency parsers from partially annotated corpora. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 776–784, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I11-1087. 9.5

Erick Fonseca and Sandra Aluísio. A deep architecture for non-projective dependency parsing. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 56–61, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.3115/v1/W15-1508. URL https://aclanthology.org/W15-1508. 3.1.1

Erick Fonseca and André F. T. Martins. Revisiting higher-order dependency parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8795–8800, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.776. URL https://aclanthology.org/2020.acl-main.776. 3.1.3, 4.4

Linton C Freeman. *Elementary applied statistics: for students in behavioral science*. New York: Wiley, 1965. 8.2.1

Yifan Fu, Xingquan Zhu, and Bin Li. A survey on instance selection for active learning. *Knowledge and information systems*, 35(2):249–283, 2013. 8.1

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016. 8.2.1

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

8.2.1

Pablo Gamallo, Marcos Garcia, and Santiago Fernández-Lanza. Dependency-based open information extraction. In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 10–18, Avignon, France, April 2012. Association for Computational Linguistics. URL https://aclanthology.org/W12-0702. 5.1

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-2501. URL https://aclanthology.org/W18-2501. 1

Caroline Gasperin. Active learning for anaphora resolution. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 1–8, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1901. 8.3.1

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021. 8.7.1

Yonatan Geifman and Ran El-Yaniv. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941*, 2017. 8.2.2, 8.5.1

Masood Ghayoomi. Using variance as a stopping criterion for active learning of frame assignment. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 1–9, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/W10-0101. 8.5.2

Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288, 2002. doi: 10.1162/089120102760275983. URL https://aclanthology.org/J02-3001. 1, 2.2, 3.2, 6.1, 7.1

Daniel Gildea and Martha Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 239–246, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073124. URL https://www.aclweb.org/anthology/P02-1031. 3.2.3, 6.1

Daniel Gissin and Shai Shalev-Shwartz. Discriminative active learning. *arXiv preprint arXiv:1907.06347*, 2019. 8.2.2

Yoav Goldberg. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*, 2019. 3.2.3

Yoav Goldberg and Michael Elhadad. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 742–750, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/N10-1115. 2.2

Carlos Gómez-Rodríguez, Michalina Strzyz, and David Vilares. A unifying theory of transition-based and sequence labeling parsing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3776–3793, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.336. URL https://aclanthology.org/2020.coling-main.336. 2.2

Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. Active learning for interactive machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 245–254, Avignon, France, April 2012. Association for Computational Linguistics. URL https://aclanthology.org/E12-1025. 8.3.2

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. 1.1, 2.1

Matthew R. Gormley and Jason Eisner. Structured belief propagation for NLP. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing: Tutorial Abstracts*, pages 5–6, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-5002. URL https://aclanthology.org/P15-5002. 4.4

Nathan Greenberg, Trapit Bansal, Patrick Verga, and Andrew McCallum. Marginal likelihood training of BiLSTM-CRF for biomedical named entity recognition from disjoint label sets. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2824–2829, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1306. URL https://aclanthology.org/D18-1306. 9.2.1, 9.5

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/guo17a.html. 7.2.3

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. Cross-lingual dependency parsing based on distributed representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1234–1244, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1119. URL https://aclanthology.org/P15-1119. 5.1, 5.4.1, 5.4.2, 5.5

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. A representation learning framework for multi-source transfer parsing. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2734–2740, 2016. 5.1, 5.5

Kamal Gupta, Dhanvanth Boppana, Rejwanul Haque, Asif Ekbal, and Pushpak Bhattacharyya. Investigating active learning in interactive neural machine translation. In *Proceedings of Machine Translation Summit XVIII: Research Track*, pages 10–22, Virtual, August 2021a. Association for Machine Translation in the Americas. URL https://aclanthology.org/

2021.mtsummit-research.2. 8.3.2

Prakhar Gupta, Jeffrey Bigham, Yulia Tsvetkov, and Amy Pavel. Controlling dialogue generation with semantic exemplars. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3018–3029, Online, June 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.240. URL https://aclanthology.org/2021.naacl-main.240. 1.1

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740. URL https://aclanthology.org/2020.acl-main.740. 8.4.2

Ben Hachey, Beatrice Alex, and Markus Becker. Investigating the effects of selective sampling on the annotation task. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 144–151, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0619. 8.3.2

Robbie Haertel, Eric Ringger, Kevin Seppi, James Carroll, and Peter McClanahan. Assessing the costs of sampling methods in active learning for annotation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 65–68, Columbus, Ohio, June 2008a. Association for Computational Linguistics. URL https://aclanthology.org/P08-2017. 8.3.2

Robbie Haertel, Paul Felt, Eric K. Ringger, and Kevin Seppi. Parallel active learning: Eliminating wait time with minimal staleness. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 33–41, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/W10-0105. 8.3.2

Robbie Haertel, Eric Ringger, Kevin Seppi, and Paul Felt. An analytic and empirical evaluation of return-on-investment-based active learning. In *Proceedings of The 9th Linguistic Annotation Workshop*, pages 11–20, Denver, Colorado, USA, June 2015. Association for Computational Linguistics. doi: 10.3115/v1/W15-1602. URL https://aclanthology.org/W15-1602. 8.3.2

Robbie A Haertel, Kevin D Seppi, Eric K Ringger, and James L Carroll. Return on investment for active learning. In *Proceedings of the NIPS workshop on cost-sensitive learning*, volume 72, 2008b. 8.3.2

Gholamreza Haffari and Anoop Sarkar. Active learning for multilingual statistical machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 181–189, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/P09-1021. 8.6

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. Active learning for statistical phrase-based machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 415–423, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/N09-1047. 8.2.1, 9.5

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W09-1201. 2.2, 2.3, 3.2, 3.2.1, 3.2.3, 6.3.1

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1206. URL https://aclanthology.org/D17-1206. 5.1

Katri Haverinen, Jenna Kanerva, Samuel Kohonen, Anna Missilä, Stina Ojala, Timo Viljanen, Veronika Laippala, and Filip Ginter. The finnish proposition bank. *Language Resources and Evaluation*, 49(4):907–926, 2015. 2.3, 6.3.1

Rishi Hazra, Parag Dutta, Shubham Gupta, Mohammed Abdul Qaathir, and Ambedkar Dukkipati. Active^2 learning: Actively reducing redundancies in active learning methods for sequence tagging and machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1982–1995, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.159. URL https://aclanthology.org/2021.naacl-main.159. 8.2.2

Han He and Jinho D Choi. Unleashing the true potential of sequence-to-sequence models for sequence tagging and structure parsing. *arXiv preprint arXiv:2302.02275*, 2023. 10

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. Revisiting self-training for neural sequence generation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgdnAVKDH. 1.2, 9.2.3, 9.5

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1044. URL https://aclanthology.org/P17-1044. 1.1, 2.2, 3.2, 3.2, 3.2.3, 6.1, 6.2.3

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369, Melbourne, Australia, July 2018a. Association for Computational Linguistics. doi: 10.18653/v1/P18-2058. URL https://www.aclweb.org/anthology/P18-2058.

3.2, 3.2.1, 3.2.3

Rui He, Shan He, and Ke Tang. Multi-domain active learning: A comparative study. *arXiv preprint arXiv:2106.13516*, 2021. 8.6

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071, Melbourne, Australia, July 2018b. Association for Computational Linguistics. doi: 10.18653/v1/P18-1192. URL https://www.aclweb.org/anthology/P18-1192. 3.2.3, 6.1, 6.4

Shexia He, Zuchao Li, and Hai Zhao. Syntax-aware multilingual semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5350–5359, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1538. URL https://aclanthology.org/D19-1538. 6.2.1, 6.4

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.201. URL https://aclanthology.org/2021.naacl-main.201. 1.1

John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1419. URL https://aclanthology.org/N19-1419. 1.1, 3.2.3, 10

Hideitsu Hino. Active learning: Problem settings and recent developments. *arXiv preprint arXiv:2012.04225*, 2020. 8.1

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 7.2.3, 9.2.3

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997. 2.2, 5.3.1

Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004. 8.7.1

Andrea Horbach and Alexis Palmer. Investigating active learning for short-answer scoring. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 301–311, San Diego, CA, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-0535. URL https://aclanthology.org/W16-0535. 8.5.1

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011. 8.2.1

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA, June 2006. Association for Computational Linguistics. URL https://aclanthology.org/N06-2015. 2.3, 6.3.1, 7.3.3

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*, pages 1890–1908, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.138. URL https://aclanthology.org/2022.naacl-main.138. 7.1, 7.2.1, 7.4

Junjie Hu and Graham Neubig. Phrase-level active learning for neural machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 1087–1099, Online, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.wmt-1.117. 8.3.1, 8.4.2

Peiyun Hu, Zack Lipton, Anima Anandkumar, and Deva Ramanan. Active learning with partial feedback. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJfSEnRqKQ. 8.3.1, 8.3.2

Rong Hu, Brian Mac Namee, and Sarah Jane Delany. Off to a good start: Using clustering to select the initial training set in active learning. In *Twenty-Third International FLAIRS Conference*, 2010. 8.5.1

Xinyu Hua and Lu Wang. Efficient argument structure extraction with transfer learning and active learning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 423–437, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.36. URL https://aclanthology.org/2022.findings-acl.36. 8.4.2

Jiaji Huang, Rewon Child, Vinay Rao, Hairong Liu, Sanjeev Satheesh, and Adam Coates. Active learning for speech recognition: the power of gradients. *arXiv preprint arXiv:1612.03226*, 2016a. 8.2.1

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. Few-shot named entity recognition: An empirical baseline study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10408–10423, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.813. URL https://aclanthology.org/2021.emnlp-main.813. 9.5

Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7304–7308. IEEE,

2013. 5.5

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: Main Volume*, pages 1022–1033, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.88. URL https://aclanthology.org/2021.eacl-main.88. 1.1

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 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4633–4646, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.317. URL https://aclanthology.org/2022.acl-long.317. 7.2.1, 7.3.3, 7.4

Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, and Avirup Sil. Liberal event extraction and event schema induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 258–268, Berlin, Germany, August 2016b. Association for Computational Linguistics. doi: 10.18653/v1/P16-1025. URL https://aclanthology.org/P16-1025. 7.4

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1201. URL https://aclanthology.org/P18-1201. 7.4

Sheng-Jun Huang, Jia-Lve Chen, Xin Mu, and Zhi-Hua Zhou. Cost-effective active learning from diverse labelers. In *IJCAI*, pages 1879–1885, 2017. 8.3.2

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015. 2.2

Rebecca Hwa. Sample selection for statistical grammar induction. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 45–52, Hong Kong, China, October 2000. Association for Computational Linguistics. doi: 10.3115/1117794.1117800. URL https://aclanthology.org/W00-1306. 8.3.1

Rebecca Hwa. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276, 2004. doi: 10.1162/0891201041850894. URL https://aclanthology.org/J04-3001. 8.3.1, 9.1, 9.2.1, 9.5

Fariz Ikhwantri, Samuel Louvan, Kemal Kurniawan, Bagas Abisena, Valdi Rachman, Alfan Farizki Wicaksono, and Rahmad Mahendra. Multi-task active learning for neural semantic role labeling on low resource conversational corpus. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 43–50, Melbourne, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-3406. URL https://aclanthology.org/W18-3406. 8.6

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Confer-*

ence of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1875–1885, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1170. URL https://aclanthology.org/N18-1170. 1.1

Borna Jafarpour, Dawn Sepehr, and Nick Pogrebnyakov. Active curriculum learning. In *Proceedings of the First Workshop on Interactive Learning for Natural Language Processing*, pages 40–45, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.internlp-1.6. URL https://aclanthology.org/2021.internlp-1.6. 8.7.1

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514, 2021. 1.1

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023. 1.1

Zhuoren Jiang, Zhe Gao, Yu Duan, Yangyang Kang, Changlong Sun, Qiong Zhang, and Xiaozhong Liu. Camouflaged Chinese spam content detection with semi-supervised generative active learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3080–3085, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.279. URL https://aclanthology.org/2020.acl-main.279. 8.2.1, 8.4.2

Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Xing Wang, and Zhaopeng Tu. Is chatgpt a good translator? a preliminary study. *arXiv preprint arXiv:2301.08745*, 2023. 1.1, 10

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.372. URL https://aclanthology.org/2020.findings-emnlp.372. 4.1

Zhanming Jie, Aldrian Obaja Muis, and Wei Lu. Efficient dependency-guided named entity recognition. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3457–3465, 2017. 5.1

Bernal Jimenez Gutierrez, Nikolas McNeal, Clayton Washington, You Chen, Lang Li, Huan Sun, and Yu Su. Thinking about GPT-3 in-context learning for biomedical IE? think again. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4497–4512, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.329. 9.4.2

Ishan Jindal, Yunyao Li, Siddhartha Brahma, and Huaiyu Zhu. CLAR: A cross-lingual argument regularizer for semantic role labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3113–3125, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.279. URL https://www.aclweb.org/anthology/2020.findings-emnlp.279. 6.4

Richard Johansson and Pierre Nugues. Extended constituent-to-dependency conversion for English. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA 2007)*, pages 105–112, Tartu, Estonia, May 2007. University of Tartu, Estonia. URL https://aclanthology.org/W07-2416. 2.3

Shafiq Joty, Preslav Nakov, Lluís Màrquez, and Israa Jaradat. Cross-language learning with adversarial neural networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 226–237, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1024. URL https://aclanthology.org/K17-1024. 5.5

Dan Jurafsky. *Speech & language processing*. Pearson, 2000. 2.1

Jaeho Kang, Kwang Ryel Ryu, and Hyuk-Chul Kwon. Using cluster-based sampling to select initial training set for active learning in text classification. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 384–388. Springer, 2004. 8.5.1

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. One-shot neural cross-lingual transfer for paradigm completion. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1993–2003, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1182. URL https://aclanthology.org/P17-1182. 5.5

Siddharth Karamcheti, Ranjay Krishna, Li Fei-Fei, and Christopher Manning. Mind your outliers! investigating the negative impact of outliers on active learning for visual question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7265–7281, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.564. URL https://aclanthology.org/2021.acl-long.564. 8.2.2

Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource deep entity resolution with transfer and active learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5851–5861, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1586. URL https://aclanthology.org/P19-1586. 8.4.2

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. Corpus annotation for mining biomedical events from literature. *BMC bioinformatics*, 9(1):1–25, 2008. 7.1, 7.3.3

Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. Overview of Genia event task in BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 7–15, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/W11-1802. 7.3.3

Joo-Kyung Kim, Young-Bum Kim, Ruhi Sarikaya, and Eric Fosler-Lussier. Cross-lingual transfer learning for POS tagging without cross-lingual resources. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2832–2838, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1302. URL https://aclanthology.org/D17-1302. 5.1, 5.5

Seokhwan Kim, Yu Song, Kyungduk Kim, Jeong-Won Cha, and Gary Geunbae Lee. MMRbased active machine learning for bio named entity recognition. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 69–72, New York City, USA, June 2006. Association for Computational Linguistics. URL `https://aclanthology.org/N06-2018`. 8.2.2, 8.2.3, 8.3.1

Yekyung Kim. Deep active learning for sequence labeling based on diversity and uncertainty in gradient. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 1–8, Suzhou, China, December 2020. Association for Computational Linguistics. URL `https://aclanthology.org/2020.lifelongnlp-1.1`. 8.2.2, 8.2.3

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL `https://aclanthology.org/D16-1139`. 4.1

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3.2.2, 4.3.1, 6.3.1

Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327, 2016. doi: 10.1162/tacl_a_00101. URL `https://aclanthology.org/Q16-1023`. 1.1, 2.2, 3.1, 3.1.1, 5.3.1

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017. 6.3.2

Nikita Kitaev and Dan Klein. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1249. URL `https://aclanthology.org/P18-1249`. 5.1

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.274. URL `https://aclanthology.org/2020.coling-main.274`. 7.3.1

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=e2TBb5y0yFf`. 10

Alexander Koller, Stephan Oepen, and Weiwei Sun. Graph-based meaning representations: Design and processing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 6–11, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-4002. URL `https://aclanthology.org/P19-4002`. 1

Ryuto Konno, Shun Kiyono, Yuichiroh Matsubayashi, Hiroki Ouchi, and Kentaro Inui. Pseudo zero pronoun resolution improves zero anaphora resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3790–3806, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.308. URL https://aclanthology.org/2021.emnlp-main.308. 7.3.3

Terry Koo and Michael Collins. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL https://aclanthology.org/P10-1001. 1.2, 2.2, 3.1.1, 3.1.1

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. Structured prediction models via the matrix-tree theorem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1015. 2.2, 3.1.1, 4.4

Omri Koshorek, Gabriel Stanovsky, Yichu Zhou, Vivek Srikumar, and Jonathan Berant. On the limits of learning to actively learn semantic representations. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 452–462, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1042. URL https://aclanthology.org/K19-1042. 8.2.1

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 (Volume 1: Long Papers)*, pages 1190–1200, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P13-1117. 6.1, 6.4

Mikhail Kozhevnikov and Ivan Titov. Cross-lingual model transfer using feature representation projection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–585, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2095. URL https://www.aclweb.org/anthology/P14-2095. 6.1, 6.4

Sandra Kübler, Ryan McDonald, and Joakim Nivre. Dependency parsing. *Synthesis lectures on human language technologies*, 1(1):1–127, 2009. 1, 2.2, 2.2, 3.1.1, 4.2.2

Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. Dynamic programming algorithms for transition-based dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 673–682, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1068. 2.2

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1277. URL `https://aclanthology.org/D19-1277`. 1.1, 2.2, 4.1

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, and Noah A. Smith. Distilling an ensemble of greedy dependency parsers into one MST parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1744–1753, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1180. URL `https://aclanthology.org/D16-1180`. 3.1

Gourab Kundu, Avi Sil, Radu Florian, and Wael Hamza. Neural cross-lingual coreference resolution and its application to entity linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 395–400, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2063. URL `https://aclanthology.org/P18-2063`. 5.5

Luke Kurlandski and Michael Bloodgood. Impact of stop sets on stopping active learning for text classification. *arXiv preprint arXiv:2201.05460*, 2022. 8.5.2

John D Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, 2001. 1, 1.2, 2.2, 3.2.1, 4.2.1, 4.3.1, 6.3.5, 9.2.1

Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019. 6.1

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1030. URL `https://aclanthology.org/N16-1030`. 2.2

Egoitz Laparra and German Rigau. Exploiting explicit annotations and semantic types for implicit argument resolution. In *2012 IEEE Sixth International Conference on Semantic Computing*, pages 75–78. IEEE, 2012. 7.4

Egoitz Laparra and German Rigau. ImpAr: A deterministic algorithm for implicit semantic role labelling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1180–1189, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://aclanthology.org/P13-1116`. 7.4

Florian Laws and Hinrich Schütze. Stopping criteria for active learning of named entity recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 465–472, Manchester, UK, August 2008. Coling 2008 Organizing Committee. URL `https://aclanthology.org/C08-1059`. 8.5.2

Florian Laws, Christian Scheible, and Hinrich Schütze. Active learning with Amazon Mechanical Turk. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1546–1556, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics. URL https://aclanthology.org/D11-1143. 8.6

Florian Laws, Florian Heimerl, and Hinrich Schütze. Active learning for coreference resolution. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 508–512, Montréal, Canada, June 2012. Association for Computational Linguistics. URL https://aclanthology.org/N12-1055. 8.3.1

LDC. ACE (automatic content extraction) english annotation guidelines for events version 5.4.3. *Linguistic Data Consortium*, 2005. 4.2.3, 7.1

LDC. Deft Rich ERE annotation guidelines: Events version 3.0. *Linguistic Data Consortium*, 2015. 7.3.1

Fei-Tzin Lee, Miguel Ballesteros, Feng Nan, and Kathleen McKeown. Using structured content plans for fine-grained syntactic control in pretrained language model generation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5882–5895, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics. URL https://aclanthology.org/2022.coling-1.514. 1.1

David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994. 8.1, 8.4.1

David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer, 1994. 1.2, 8.1, 8.1.1, 8.2.1

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*, pages 7871–7880, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703. 7.3.1

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *NeurIPS*, 2020b. 10

Belinda Z. Li, Gabriel Stanovsky, and Luke Zettlemoyer. Active learning for coreference resolution using discrete annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8320–8331, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.738. URL https://aclanthology.org/2020.acl-main.738. 8.3.1, 8.3.2, 9.5

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online, November 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.73. URL https://aclanthology.org/2020.findings-emnlp.73. 7.1, 7.2.1, 7.4

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70, 2020c.

4.1

Maolin Li, Nhung Nguyen, and Sophia Ananiadou. Proactive learning for named entity recognition. In *BioNLP 2017*, pages 117–125, Vancouver, Canada,, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2314. URL https://aclanthology.org/W17-2314. 8.3.2

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*, pages 894–908, Online, June 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.69. URL https://aclanthology.org/2021.naacl-main.69. 7.1, 7.1, 7.2.1, 7.3.1, 7.4

Shoushan Li, Shengfeng Ju, Guodong Zhou, and Xiaojun Li. Active learning for imbalanced sentiment classification. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 139–148, Jeju Island, Korea, July 2012a. Association for Computational Linguistics. URL https://aclanthology.org/D12-1013. 8.6

Shoushan Li, Guodong Zhou, and Chu-Ren Huang. Active learning for Chinese word segmentation. In *Proceedings of COLING 2012: Posters*, pages 683–692, Mumbai, India, December 2012b. The COLING 2012 Organizing Committee. URL https://aclanthology.org/C12-2067. 8.3.1

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online, July 2020d. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.744. URL https://www.aclweb.org/anthology/2020.acl-main.744. 3.2.3, 4.4

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353. 7.4

Yangming Li, Lemao Liu, and Shuming Shi. Rethinking negative sampling for handling missing entity annotations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7188–7197, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.497. URL https://aclanthology.org/2022.acl-long.497. 9.5

Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. Active learning for dependency parsing with partial annotation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 344–354, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1033. URL https://aclanthology.org/P16-1033. 8.3.1, 8.3.2, 9.1, 9.2.1, 9.2.2, 9.3.2, 9.5

Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. Improving BERT with syntax-aware local attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 645–653, Online, August 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.57. URL https://aclanthology.org/2021.findings-acl.57. 10

Zuchao Li, Shexia He, Jiaxun Cai, Zhuosheng Zhang, Hai Zhao, Gongshen Liu, Linlin Li, and Luo Si. A unified syntax-aware framework for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2401–2411, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1262. URL https://www.aclweb.org/anthology/D18-1262. 3.2.3

Weixin Liang, James Zou, and Zhou Yu. ALICE: Active learning with contrastive natural language explanations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4380–4391, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.355. URL https://aclanthology.org/2020.emnlp-main.355. 8.7.2

Bill Yuchen Lin, Dong-Ho Lee, Frank F. Xu, Ouyu Lan, and Xiang Ren. AlpacaTag: An active learning-based crowd annotation framework for sequence tagging. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 58–63, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-3010. URL https://aclanthology.org/P19-3010. 8.6

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5182–5192, Florence, Italy, July 2019b. Association for Computational Linguistics. doi: 10.18653/v1/P19-1511. URL https://www.aclweb.org/anthology/P19-1511. 3.2, 3.2.1, 3.2.3

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*, pages 7999–8009, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.713. URL https://aclanthology.org/2020.acl-main.713. 1, 4.3.1, 4.3.1, 7.3.1, 11

Thomas Lippincott and Ben Van Durme. Active learning and negative evidence for language identification. In *Proceedings of the Second Workshop on Data Science with Human in the Loop: Language Advances*, pages 47–51, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.dash-1.8. URL https://aclanthology.org/2021.dash-1.8. 8.3.1

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018. 1.1, 10

Robert Litschko, Goran Glavaš, Simone Paolo Ponzetto, and Ivan Vulić. Unsupervised cross-lingual information retrieval using monolingual data only. In *The 41st International ACM*

SIGIR Conference on Research & Development in Information Retrieval, pages 1253–1256, 2018. 5.5

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori 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: Volume 2, Short Papers*, pages 8–14, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-2002. 5.2

Haitao Liu. Dependency direction as a means of word-order typology: A method based on dependency treebanks. *Lingua*, 120(6):1567–1578, 2010. 5.2

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online, May 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.deelio-1.10. URL https://aclanthology.org/2022.deelio-1.10. 9.4.2

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.128. URL https://aclanthology.org/2020.emnlp-main.128. 7.1, 7.2.1, 7.4

Jian Liu, Yufeng Chen, and Jinan Xu. Machine reading comprehension as data augmentation: A case study on implicit event argument extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2725, Online and Punta Cana, Dominican Republic, November 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.214. URL https://aclanthology.org/2021.emnlp-main.214. 7.1, 7.2.1, 7.2.3, 7.4

Jin Liu, Chongfeng Fan, Zhou Fengyu, and Huijuan Xu. Syntax controlled knowledge graph-to-text generation with order and semantic consistency. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1278–1291, Seattle, United States, July 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.95. URL https://aclanthology.org/2022.findings-naacl.95. 1.1

Ming Liu, Wray Buntine, and Gholamreza Haffari. Learning to actively learn neural machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 334–344, Brussels, Belgium, October 2018a. Association for Computational Linguistics. doi: 10.18653/v1/K18-1033. URL https://aclanthology.org/K18-1033. 8.2.1

Ming Liu, Wray Buntine, and Gholamreza Haffari. Learning how to actively learn: A deep imitation learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1883, Melbourne, Australia, July 2018b. Association for Computational Linguistics. doi: 10.18653/v1/P18-1174. URL https://aclanthology.org/P18-1174. 8.2.1

Mingyi Liu, Zhiying Tu, Tong Zhang, Tonghua Su, Xiaofei Xu, and Zhongjie Wang. Ltp: A new active learning strategy for crf-based named entity recognition. *Neural Processing Letters*, pages 1–22, 2022c. 8.3.1

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. *arXiv preprint arXiv:2107.13586*, 2021b. 7.4

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 Computing Surveys*, 2023. 1.1, 1.1, 10

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. *Internation Conference on Learning Representations*, 2018c. 5.1

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. Leveraging FrameNet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2134–2143, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1201. URL https://aclanthology.org/P16-1201. 7.4

Yijia Liu, Yi Zhu, Wanxiang Che, Bing Qin, Nathan Schneider, and Noah A. Smith. Parsing tweets into Universal Dependencies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 965–975, New Orleans, Louisiana, June 2018d. Association for Computational Linguistics. doi: 10.18653/v1/N18-1088. URL https://aclanthology.org/N18-1088. 9.3.3

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 arXiv:1907.11692*, 2019. 1.1, 6.1, 7.3.1, 9.1, 9.3.1

Varvara Logacheva and Lucia Specia. Confidence-based active learning methods for machine translation. In *Proceedings of the EACL 2014 Workshop on Humans and Computer-assisted Translation*, pages 78–83, Gothenburg, Sweden, April 2014a. Association for Computational Linguistics. doi: 10.3115/v1/W14-0312. URL https://aclanthology.org/W14-0312. 8.2.1

Varvara Logacheva and Lucia Specia. A quality-based active sample selection strategy for statistical machine translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2690–2695, Reykjavik, Iceland, May 2014b. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/658_Paper.pdf. 8.2.1

Shayne Longpre, Julia Reisler, Edward Greg Huang, Yi Lu, Andrew Frank, Nikhil Ramesh, and Chris DuBois. Active learning over multiple domains in natural language tasks. *arXiv preprint arXiv:2202.00254*, 2022. 8.6

David Lowell, Zachary C. Lipton, and Byron C. Wallace. Practical obstacles to deploying active learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 21–30, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1003. URL https://aclanthology.org/D19-1003. 8.4.1, 8.7.2

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 (Volume 1: Long Papers)*, pages 2795–2806, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.217. URL https://aclanthology.org/2021.acl-long.217. 7.1, 7.2.1, 7.4

Teresa Lynn, Jennifer Foster, Mark Dras, and Elaine Uí Dhonnchadha. Active learning and the Irish treebank. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, pages 23–32, Dunedin, New Zealand, December 2012. URL https://aclanthology.org/U12-1005. 8.3.2

Qing Lyu, Hongming Zhang, Elior Sulem, and Dan Roth. Zero-shot event extraction via transfer learning: Challenges and insights. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 322–332, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-short.42. URL https://aclanthology.org/2021.acl-short.42. 7.1, 7.2.1, 7.4

Jie Ma, Shuai Wang, Rishita Anubhai, Miguel Ballesteros, and Yaser Al-Onaizan. Resource-enhanced neural model for event argument extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3554–3559, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.318. URL https://aclanthology.org/2020.findings-emnlp.318. 7.1, 7.4

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1101. URL https://aclanthology.org/P16-1101. 1.1, 2.2, 3.1

Xuezhe Ma and Eduard Hovy. Neural probabilistic model for non-projective MST parsing. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 59–69, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I17-1007. 3.1, 3.1.1

Xuezhe Ma and Fei Xia. Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1337–1348, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/

v1/P14-1126. URL https://aclanthology.org/P14-1126. 5.5

Xuezhe Ma and Hai Zhao. Fourth-order dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 785–796, Mumbai, India, December 2012. The COLING 2012 Organizing Committee. URL https://aclanthology.org/C12-2077. 2.2, 3.1.1

Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1403–1414, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1130. URL https://aclanthology.org/P18-1130. 3.1, 3.1.2, 5.3.2, 5.4.1

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.466. URL https://aclanthology.org/2022.acl-long.466. 7.2.1, 7.4

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. The penn arabic treebank: Building a large-scale annotated arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, 2004. 2.3

Seiji Maekawa, Dan Zhang, Hannah Kim, Sajjadur Rahman, and Estevam Hruschka. Low-resource interactive active labeling for fine-tuning language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3230–3242, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.235. 8.2.2, 8.3.2

Saeed Majidi and Gregory Crane. Active learning for dependency parsing by a committee of parsers. In *Proceedings of the 13th International Conference on Parsing Technologies (IWPT 2013)*, pages 98–105, Nara, Japan, November 2013. Assocation for Computational Linguistics. URL https://aclanthology.org/W13-5711. 8.3.1, 8.4.2, 9.1

Gideon Mann and Andrew McCallum. Efficient computation of entropy gradient for semi-supervised conditional random fields. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 109–112, Rochester, New York, April 2007. Association for Computational Linguistics. URL https://aclanthology.org/N07-2028. 8.3.1

Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing*. MIT press, 1999. 2.1

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-5010. URL https://aclanthology.org/P14-5010. 1, 3.2.2, 6.3.1, 16

Diego Marcheggiani and Thierry Artières. An experimental comparison of active learning strategies for partially labeled sequences. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 898–906, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1097. URL https://aclanthology.org/D14-1097. 8.3.1, 9.1, 9.2.1, 9.5

Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1159. URL https://aclanthology.org/D17-1159. 1.2, 3.2.3, 6.1, 6.3.2, 6.4

Diego Marcheggiani and Ivan Titov. Graph convolutions over constituent trees for syntax-aware semantic role labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3915–3928, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.322. URL https://www.aclweb.org/anthology/2020.emnlp-main.322. 3.2.3

Diego Marcheggiani, Anton Frolov, and Ivan Titov. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 411–420, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1041. URL https://www.aclweb.org/anthology/K17-1041. 3.2.3, 6.4

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. URL https://aclanthology.org/J93-2004. 1.1, 2.3

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.51. URL https://aclanthology.org/2021.emnlp-main.51. 8.2.1

Katerina Margatina, Loic Barrault, and Nikolaos Aletras. On the importance of effectively adapting pretrained language models for active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 825–836, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.93. URL https://aclanthology.org/2022.acl-short.93. 8.4.2

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. Special issue introduction: Semantic role labeling: An introduction to the special issue. *Computational Linguistics*, 34(2):145–159, 2008. doi: 10.1162/coli.2008.34.2.145. URL https://www.aclweb.org/anthology/J08-2001. 2.2, 3.2.3

André FT Martins, Mário AT Figueiredo, Pedro MQ Aguiar, Noah A Smith, and Eric P Xing. Ad3: Alternating directions dual decomposition for map inference in graphical models. *The Journal of Machine Learning Research*, 16(1):495–545, 2015. 4.4

Stephen Mayhew, Snigdha Chaturvedi, Chen-Tse Tsai, and Dan Roth. Named entity recognition with partially annotated training data. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 645–655, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1060. URL `https://aclanthology.org/K19-1060`. 9.5

Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 188–191, 2003. URL `https://aclanthology.org/W03-0430`. 1

Andrew McCallum and Kamal Nigam. Employing em and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 350–358, 1998. 8.2.1, 8.2.2, 8.4.2

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. Maximum entropy markov models for information extraction and segmentation. In *ICML*, pages 591–598, 2000. 2.2

David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA, June 2006. Association for Computational Linguistics. URL `https://aclanthology.org/N06-1020`. 1.2, 9.2.3, 9.5

David McClosky, Mihai Surdeanu, and Christopher Manning. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1626–1635, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `https://aclanthology.org/P11-1163`. 5.1

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL `https://aclanthology.org/P19-1334`. 1.1

Ryan McDonald and Joakim Nivre. Recent advances in dependency parsing. *Tutorial at EACL*, 2014. 2.2

Ryan McDonald and Fernando Pereira. Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 81–88, Trento, Italy, April 2006. Association for Computational Linguistics. URL `https://aclanthology.org/E06-1011`. 1.2, 2.2, 3.1.1

Ryan McDonald and Giorgio Satta. On the complexity of non-projective data-driven dependency parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 121–132, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `https://aclanthology.org/W07-2216`. 2.2, 3.1.1, 4.4

Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98, Ann Arbor, Michigan, June 2005a. Association for Computational Linguistics. doi: 10.3115/1219840.1219852. URL https://aclanthology.org/P05-1012. 1.2, 2.1, 2.2, 3.1, 3.1.1, 4.2.2, 5.3.2

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada, October 2005b. Association for Computational Linguistics. URL https://aclanthology.org/H05-1066. 3.1, 3.1.1, 3.1.1

Ryan McDonald, Slav Petrov, and Keith Hall. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics. URL https://aclanthology.org/D11-1006. 5.5

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. Universal Dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/P13-2017. 5.4.2, 5.5

Prem Melville and Vikas Sindhwani. Active dual supervision: Reducing the cost of annotating examples and features. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 49–57, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1907. 8.7.2

Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10367–10378, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.810. URL https://aclanthology.org/2021.emnlp-main.810. 9.5

Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *arXiv preprint arXiv:2106.08962*, 2021. 8.7.1, 10

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. The NomBank project: An interim report. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 24–31, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-2705. 2.3, 7.2.2

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. Crowdsourcing question-answer meaning representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 560–568, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2089. URL https://aclanthology.org/N18-2089. 7.3.1

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010. 5.1

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a. 1.1

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013b. 1.1

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018. 3.2.2

Timothy Miller, Dmitriy Dligach, and Guergana Savova. Active learning for coreference resolution. In *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pages 73–81, Montréal, Canada, June 2012. Association for Computational Linguistics. URL https://aclanthology.org/W12-2409. 8.3.1

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5316–5330, Dublin, Ireland, May 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.365. URL https://aclanthology.org/2022.acl-long.365. 9.4.2

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates, December 2022b. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.759. 1.1, 9.4.2, 10

Seyed Abolghasem Mirroshandel and Alexis Nasr. Active learning for dependency parsing using partially annotated sentences. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 140–149, Dublin, Ireland, October 2011. Association for Computational Linguistics. URL https://aclanthology.org/W11-2917. 8.3.1, 9.1, 9.2.1

Seyed Abolghasem Mirroshandel, Gholamreza Ghassem-Sani, and Alexis Nasr. Active learning strategies for support vector machines, application to temporal relation classification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 56–64, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I11-1007. 8.2.3

Akiva Miura, Graham Neubig, Michael Paul, and Satoshi Nakamura. Selecting syntactic, non-redundant segments in active learning for machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 20–29, San Diego, California, June 2016.

Association for Computational Linguistics. doi: 10.18653/v1/N16-1003. URL https://aclanthology.org/N16-1003. 8.3.1

Akash Kumar Mohankumar and Mitesh Khapra. Active evaluation: Efficient NLG evaluation with few pairwise comparisons. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8761–8781, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.600. URL https://aclanthology.org/2022.acl-long.600. 8.7.2

Joel Moniz, Barun Patra, and Matthew Gormley. On efficiently acquiring annotations for multilingual models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 69–85, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.9. URL https://aclanthology.org/2022.acl-short.9. 8.4.2, 8.6

Ali Mottaghi, Prathusha K Sarma, Xavier Amatriain, Serena Yeung, and Anitha Kannan. Medical symptom recognition from patient text: An active learning approach for long-tailed multilabel distributions. *arXiv preprint arXiv:2011.06874*, 2020. 8.6

Phoebe Mulcaire, Swabha Swayamdipta, and Noah A. Smith. Polyglot semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 667–672, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2106. URL https://www.aclweb.org/anthology/P18-2106. 6.4

Phoebe Mulcaire, Jungo Kasai, and Noah A. Smith. Polyglot contextual representations improve crosslingual transfer. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3912–3918, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1392. URL https://www.aclweb.org/anthology/N19-1392. 6.4

Max Müller-Eberstein, Rob van der Goot, and Barbara Plank. Probing for labeled dependency trees. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7711–7726, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.532. URL https://aclanthology.org/2022.acl-long.532. 1.1

Stephen Mussmann, Robin Jia, and Percy Liang. On the Importance of Adaptive Data Collection for Extremely Imbalanced Pairwise Tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3400–3413, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.305. URL https://aclanthology.org/2020.findings-emnlp.305. 8.6, 8.7.2

Skatje Myers and Martha Palmer. Tuning deep active learning for semantic role labeling. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 212–221, Groningen, The Netherlands (online), June 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.iwcs-1.20. 8.3.1, 9.5

Tahira Naseem, Regina Barzilay, and Amir Globerson. Selective sharing for multilingual dependency parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 629–637, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL `https://aclanthology.org/P12-1066`. 5.1, 5.5

Graham Neubig and Shinsuke Mori. Word-based partial annotation for efficient corpus construction. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010. European Language Resources Association (ELRA). URL `http://www.lrec-conf.org/proceedings/lrec2010/pdf/408_Paper.pdf`. 9.5

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 529–533, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `https://aclanthology.org/P11-2093`. 8.3.1

Grace Ngai and David Yarowsky. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 117–125, Hong Kong, October 2000. Association for Computational Linguistics. doi: 10.3115/1075218.1075234. URL `https://aclanthology.org/P00-1016`. 8.3.1, 8.3.2

Hieu T Nguyen and Arnold Smeulders. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 79, 2004. 8.2.2

Minh Van Nguyen, Nghia Ngo, Bonan Min, and Thien Nguyen. FAMIE: A fast active learning framework for multilingual information extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 131–139, Hybrid: Seattle, Washington + Online, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-demo.14. URL `https://aclanthology.org/2022.naacl-demo.14`. 8.4.1

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1034. URL `https://aclanthology.org/N16-1034`. 7.4

Ansong Ni, Pengcheng Yin, and Graham Neubig. Merging weak and active supervision for semantic parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8536–8543, 2020. 8.4.2

Qiang Ning, Zhongzhi Yu, Chuchu Fan, and Dan Roth. Exploiting partially annotated data in temporal relation extraction. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 148–153, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/S18-2018. URL `https://aclanthology.org/S18-2018`. 9.5

Qiang Ning, Hangfeng He, Chuchu Fan, and Dan Roth. Partial or complete, that's the question.

In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2190–2200, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1227. URL https://aclanthology.org/N19-1227. 9.5

Joakim Nivre. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 149–160, Nancy, France, April 2003. URL https://aclanthology.org/W03-3017. 2.2, 2.2

Joakim Nivre. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-0308. 2.2

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 915–932, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1096. 2.3

Joakim Nivre, Željko Agić, Lars Ahrenberg, and et al. Universal dependencies 1.4, 2016a. URL http://hdl.handle.net/11234/1-1827. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. 6.3.2

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia, May 2016b. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1262. 1.2, 2.3, 6.1

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. Universal dependencies 2.3, 2018a. URL http://hdl.handle.net/11234/1-2895. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. 3.1.2

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. Universal dependencies 2.2, 2018b. URL http://hdl.handle.net/11234/1-2837. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. 5.1, 5.2

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.497. 1.1, 1.2, 2.3, 3.2.2, 4.3.1, 6.1, 9.3.1

Tim O'Gorman, Sameer Pradhan, Martha Palmer, Julia Bonn, Katie Conger, and James Gung. The new Propbank: Aligning Propbank with AMR through POS unification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1231. 2.3

Timothy J O'Gorman. *Bringing together computational and linguistic models of implicit role interpretation*. PhD thesis, University of Colorado at Boulder, 2019. 7.1, 2, 7.4

Fredrik Olsson. A literature survey of active machine learning in the context of natural language processing. 2009. 8.1

Fredrik Olsson and Katrin Tomanek. An intrinsic stopping criterion for committee-based active learning. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 138–146, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1118. 8.5.2

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1.1, 10

Robert Östling. Word order typology through multilingual word alignment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 205–211, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-2034. URL https://aclanthology.org/P15-2034. 5.2

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1191. URL https://www.aclweb.org/anthology/D18-1191. 3.2, 3.2.3, 7.1, 7.2.1

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022. 10

Sebastian Padó and Mirella Lapata. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36:307–340, 2009. 1.2, 6.4

Martha Palmer. Semlink: Linking propbank, verbnet and framenet. In *Proceedings of the generative lexicon conference*, pages 9–15. GenLex-09, Pisa, Italy, 2009. 2.3, 7

Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005. doi: 10.1162/0891201053630264. URL https://aclanthology.org/J05-1004. 2.2, 2.3, 7.1, 7.2.2

Martha Palmer, Olga Babko-Malaya, Ann Bies, Mona Diab, Mohamed Maamouri, Aous Mansouri, and Wajdi Zaghouani. A pilot Arabic Propbank. In *Proceedings of the Sixth Inter-*

national Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2008/pdf/880_paper.pdf. 2.3

Martha Palmer, Daniel Gildea, and Nianwen Xue. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, 2010. 2.2, 3.2, 6.1, 7.1

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. 1.2, 7.1

Xingyuan Pan, Maitrey Mehta, and Vivek Srikumar. Learning constraints for structured prediction using rectifier networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4843–4858, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.438. URL https://aclanthology.org/2020.acl-main.438. 4.4

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, RISHITA ANUBHAI, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. Structured prediction as translation between augmented natural languages. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=US-TP-xnXI. 7.2.1, 7.4, 10

Mark A Paskin. Cubic-time parsing and learning algorithms for grammatical bigram models. 2001. 2.2, 3.1.1, 4.3.1, 4.4

Wenzhe Pei, Tao Ge, and Baobao Chang. An effective neural network model for graph-based dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 313–322, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1031. URL https://aclanthology.org/P15-1031. 2.2, 3.1, 3.1.1

Haoruo Peng, Kai-Wei Chang, and Dan Roth. A joint framework for coreference resolution and mention head detection. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 12–21, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.18653/v1/K15-1002. URL https://www.aclweb.org/anthology/K15-1002. 3.2, 3.2.3

Haoruo Peng, Yangqiu Song, and Dan Roth. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1038. URL https://aclanthology.org/D16-1038. 7.4

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162. 1.1

Álvaro Peris and Francisco Casacuberta. Active learning for interactive neural machine translation of data streams. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 151–160, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-1015. URL https://aclanthology.org/K18-1015. 8.3.2

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, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202. 1.1, 5.1, 9.1

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250. URL https://aclanthology.org/D19-1250. 7.4

Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2089–2096, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf. 2.3, 5.5

Jason Phang, Thibault Févry, and Samuel R Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*, 2018. 6.3.2, 7.3.2

Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1493. URL https://www.aclweb.org/anthology/P19-1493. 6.1, 6.2.1, 6.4

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1119. URL https://aclanthology.org/N19-1119. 8.7.1

Amir Pouran Ben Veyseh, Tuan Ngo Nguyen, and Thien Huu Nguyen. Graph transformer networks with syntactic and semantic structures for event argument extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3651–3661, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.326. URL https://aclanthology.org/2020.findings-emnlp.326. 7.1, 7.4

Ameya Prabhu, Charles Dognin, and Maneesh Singh. Sampling bias in deep active classification: An empirical study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4058–4068, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1417. URL `https://aclanthology.org/D19-1417`. 8.2.2

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. CoNLL-2011 shared task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–27, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `https://aclanthology.org/W11-1901`. 2.3

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL `https://aclanthology.org/W12-4501`. 2.3, 6.3.5

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W13-3516`. 3.2.2

Sameer Pradhan, Julia Bonn, Skatje Myers, Kathryn Conger, Tim O'gorman, James Gung, Kristin Wright-bettner, and Martha Palmer. PropBank comes of Age—Larger, smarter, and more diverse. In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, pages 278–288, Seattle, Washington, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.starsem-1.24. URL `https://aclanthology.org/2022.starsem-1.24`. 2.3

Ondřej Pražák and Miloslav Konopík. Cross-lingual SRL based upon Universal Dependencies. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 592–600, Varna, Bulgaria, September 2017. INCOMA Ltd. doi: 10.26615/978-954-452-049-6_077. URL `https://doi.org/10.26615/978-954-452-049-6_077`. 6.1, 6.3.4, 6.4

Zac Pullar-Strecker, Katharina Dost, Eibe Frank, and Jörg Wicker. Hitting the target: Stopping active learning at the cost-based optimum. *arXiv preprint arXiv:2110.03802*, 2021. 8.5.2

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. Semantic role labeling via integer linear programming inference. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 1346–1352, Geneva, Switzerland, aug 23–aug 27 2004. COLING. URL `https://www.aclweb.org/anthology/C04-1197`. 3.2.3

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287, 2008. doi: 10.1162/coli.2008.34.2.257. URL `https://aclanthology.org/J08-2005`. 1.2, 3.2.3, 3.2.3, 4.4, 6.1

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. Universal Dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-2016. URL https://aclanthology.org/K18-2016. 2.2

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.14. URL https://aclanthology.org/2020.acl-demos.14. 2.2, 7.3.1, 4

Kun Qian, Poornima Chozhiyath Raman, Yunyao Li, and Lucian Popa. Learning structured representations of entity names using Active Learning and weak supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6376–6383, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.517. URL https://aclanthology.org/2020.emnlp-main.517. 8.4.2

Longhua Qian, Haotian Hui, Ya'nan Hu, Guodong Zhou, and Qiaoming Zhu. Bilingual active learning for relation classification via pseudo parallel corpora. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1055. URL https://aclanthology.org/P14-1055. 8.4.2, 8.6

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. Reasoning with language model prompting: A survey. *arXiv preprint arXiv:2212.09597*, 2022. 10

Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, et al. A survey on text-to-sql parsing: Concepts, methods, and future directions. *arXiv preprint arXiv:2208.13629*, 2022. 1.1

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*, 2023. 1.1, 10

Husam Quteineh, Spyridon Samothrakis, and Richard Sutcliffe. Textual data augmentation for efficient active learning on tiny datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7400–7410, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.600. URL https://aclanthology.org/2020.emnlp-main.600. 8.4.2

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 2.2

Puria Radmard, Yassir Fathullah, and Aldo Lipani. Subsequence based deep active learning for named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Nat-*

ural Language Processing (Volume 1: Long Papers), pages 4310–4321, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.332. URL https://aclanthology.org/2021.acl-long.332. 8.3.1, 9.1, 9.2.1, 9.3.2, 9.5

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. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020. 10

Piyush Rai, Avishek Saha, Hal Daumé, and Suresh Venkatasubramanian. Domain adaptation meets active learning. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 27–32, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/W10-0104. 8.2.2, 8.4.2

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264. 7.2.1, 7.2.3

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL https://aclanthology.org/P18-2124. 7.2.1, 7.2.3, 7.3.1

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.431. URL https://aclanthology.org/2020.emnlp-main.431. 7.3.3

Lance Ramshaw and Mitch Marcus. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*, 1995. URL https://aclanthology.org/W95-0107. 2.2, 3.2.1, 4.2.1

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015. 10

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1119. 2.2

Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*, 1996. URL https://aclanthology.org/W96-0213. 1

Christian Raymond and Giuseppe Riccardi. Generative and discriminative algorithms for spoken language understanding. In *Interspeech 2007-8th Annual Conference of the International*

Speech Communication Association, 2007. 1

Ines Rehbein and Josef Ruppenhofer. Evaluating the impact of coder errors on active learning. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 43–51, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1005. 8.6

Ines Rehbein, Josef Ruppenhofer, and Alexis Palmer. Bringing active learning to life. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 949–957, Beijing, China, August 2010. Coling 2010 Organizing Committee. URL https://aclanthology.org/C10-1107. 8.7.2

Roi Reichart, Katrin Tomanek, Udo Hahn, and Ari Rappoport. Multi-task active learning for linguistic annotations. In *Proceedings of ACL-08: HLT*, pages 861–869, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL https://aclanthology.org/P08-1098. 8.6, 9.3.4

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)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410. 9.4.2

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Computing Surveys (CSUR)*, 54 (9):1–40, 2021. 8.1, 8.1

Eric Ringger, Peter McClanahan, Robbie Haertel, George Busby, Marc Carmen, James Carroll, Kevin Seppi, and Deryle Lonsdale. Active learning for part-of-speech tagging: Accelerating corpus annotation. In *Proceedings of the Linguistic Annotation Workshop*, pages 101–108, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/W07-1516. 8.3.1

Eric Ringger, Marc Carmen, Robbie Haertel, Kevin Seppi, Deryle Lonsdale, Peter McClanahan, James Carroll, and Noel Ellison. Assessing the costs of machine-assisted corpus annotation through a user study. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2008/pdf/832_paper.pdf. 8.3.2, 8.3.2

Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009. 9.4.2

Martha-Alicia Rocha and Joan-Andreu Sanchez. Towards the supervised machine translation: Real word alignments and translations in a multi-task active learning process. In *Proceedings of Machine Translation Summit XIV: Posters*, Nice, France, September 2-6 2013. URL https://aclanthology.org/2013.mtsummit-posters.6. 8.3.1, 8.6

Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pages 413–424. Springer, 2006. 8.3.1

Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-2401. 4.4

Dan Roth and Wen-tau Yih. Global inference for entity and relation identification via a linear programming formulation. *Introduction to statistical relational learning*, pages 553–580, 2007. 4.4

Guy Rotman and Roi Reichart. Multi-task active learning for pre-trained transformer-based models. *Transactions of the Association for Computational Linguistics*, 10:1209–1228, 2022. doi: 10.1162/tacl_a_00515. URL https://aclanthology.org/2022.tacl-1.70. 8.6, 9.3.4

Nicholas Roy and Andrew McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 441–448, 2001. 8.2.1, 8.2.2, 8.3.2

Dongyu Ru, Jiangtao Feng, Lin Qiu, Hao Zhou, Mingxuan Wang, Weinan Zhang, Yong Yu, and Lei Li. Active sentence learning by adversarial uncertainty sampling in discrete space. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4908–4917, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.441. URL https://aclanthology.org/2020.findings-emnlp.441. 8.3.2

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. 1.2, 6.4

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-5004. URL https://aclanthology.org/N19-5004. 1.2, 7.1

Alexander M Rush and MJ Collins. A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing. *Journal of Artificial Intelligence Research*, 45: 305–362, 2012. 4.4

Mrinmaya Sachan, Eduard Hovy, and Eric P Xing. An active learning approach to coreference resolution. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015. 8.3.1

Avishek Saha, Piyush Rai, Hal Daumé, Suresh Venkatasubramanian, and Scott L DuVall. Active supervised domain adaptation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 97–112. Springer, 2011. 8.4.2

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 4.1

Shota Sasaki, Shuo Sun, Shigehiko Schamoni, Kevin Duh, and Kentaro Inui. Cross-lingual learning-to-rank with shared representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 458–463, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2073. URL `https://aclanthology.org/N18-2073`. 5.5

Manabu Sassano and Sadao Kurohashi. Using smaller constituents rather than sentences in active learning for Japanese dependency parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 356–365, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `https://aclanthology.org/P10-1037`. 8.3.1, 9.1

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022. 10

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *International Symposium on Intelligent Data Analysis*, pages 309–318. Springer, 2001. 8.2.1, 8.3.1, 9.2.1

Andrew I Schein and Lyle H Ungar. Active learning for logistic regression: an evaluation. *Machine Learning*, 68(3):235–265, 2007. 8.2.1

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: Main Volume*, pages 255–269, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.20. URL `https://aclanthology.org/2021.eacl-main.20`. 7.4

Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 839–846, 2000. 8.2.1, 8.5.2

Christopher Schröder and Andreas Niekler. A survey of active learning for text classification using deep neural networks. *arXiv preprint arXiv:2008.07267*, 2020. 8.1

Christopher Schröder, Andreas Niekler, and Martin Potthast. Revisiting uncertainty-based query strategies for active learning with transformers. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2194–2203, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.172. URL `https://aclanthology.org/2022.findings-acl.172`. 8.2.1

Raphael Schumann and Ines Rehbein. Active learning via membership query synthesis for semi-supervised sentence classification. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 472–481, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1044. URL `https://aclanthology.org/K19-1044`. 8.4.2

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL `https:`

//openreview.net/forum?id=H1aIuk-RW. 8.2.2

Burr Settles. Active learning literature survey. 2009. 1.2, 8.1, 8.2.1, 8.2.2, 9.1, 9.2.1, 9.4.1

Burr Settles. From theories to queries: Active learning in practice. In *Active learning and experimental design workshop in conjunction with AISTATS 2010*, pages 1–18. JMLR Workshop and Conference Proceedings, 2011. 8.7.2

Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, Honolulu, Hawaii, October 2008. Association for Computational Linguistics. URL https://aclanthology.org/D08-1112. 8.2.1, 8.2.2, 8.2.3, 8.3.1, 9.1, 9.5

Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. *Advances in neural information processing systems*, 20, 2007. 8.2.1

Burr Settles, Mark Craven, and Lewis Friedland. Active learning with real annotation costs. In *Proceedings of the NIPS workshop on cost-sensitive learning*, volume 1, 2008. 8.3.2, 8.3.2, 9.3.2

H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294, 1992. 8.2.1

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. 8.2.1

Manali Sharma, Di Zhuang, and Mustafa Bilgic. Active learning with rationales for text classification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 441–451, Denver, Colorado, May–June 2015. Association for Computational Linguistics. doi: 10.3115/v1/N15-1047. URL https://aclanthology.org/N15-1047. 8.7.2

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2074. URL https://www.aclweb.org/anthology/N18-2074. 3.2.2, 5.3.1, 5.4.3

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. Active learning for sequence tagging with deep pre-trained models and Bayesian uncertainty estimates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1698–1712, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.145. URL https://aclanthology.org/2021.eacl-main.145. 8.2.1, 8.4.1

Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Annual Meeting of the Asso-*

ciation for Computational Linguistics (ACL-04), pages 589–596, Barcelona, Spain, July 2004. doi: 10.3115/1218955.1219030. URL https://aclanthology.org/P04-1075. 8.2.2, 8.2.3, 8.3.1, 9.1

Shirong Shen, Zhen Li, and Guilin Qi. Active learning for event extraction with memory-based loss prediction model. *arXiv preprint arXiv:2112.03073*, 2021. 8.2.1

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. Deep active learning for named entity recognition. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ry018WZAZ. 8.2.1, 8.3.1, 9.1, 9.2.1, 9.5

Peng Shi and Jimmy Lin. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*, 2019. 2.2, 3.2, 3.2, 3.2.3, 4.1, 6.1, 6.2.1, 6.2.3

Tianze Shi, Igor Malioutov, and Ozan Irsoy. Semantic role labeling as syntactic dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7551–7571, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.610. URL https://www.aclweb.org/anthology/2020.emnlp-main.610. 6.4

Tianze Shi, Adrian Benton, Igor Malioutov, and Ozan İrsoy. Diversity-aware batch active learning for dependency parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2616–2626, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.207. URL https://aclanthology.org/2021.naacl-main.207. 8.2.2, 8.2.3

Xiaoxiao Shi, Wei Fan, and Jiangtao Ren. Actively transfer domain knowledge. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 342–357. Springer, 2008. 8.4.2

Aditya Siddhant and Zachary C. Lipton. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1318. URL https://aclanthology.org/D18-1318. 8.2.1

Avirup Sil, Gourab Kundu, Radu Florian, and Wael Hamza. Neural cross-lingual entity linking. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 5.1, 5.5

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Chris Manning. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2897–2904, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/1089_Paper.pdf. 6.3.1

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5972–5981, 2019. 8.2.2

Maria Skeppstedt. Annotating named entities in clinical text by combining pre-annotation and active learning. In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*, pages 74–80, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/P13-3011. 8.3.2

David A. Smith and Noah A. Smith. Probabilistic models of nonprojective dependency trees. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 132–140, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1014. 2.2, 3.1.1, 4.4

Noah A Smith. Linguistic structure prediction. *Synthesis lectures on human language technologies*, 4(2):1–274, 2011. 1, 2.1, 8.3.1, 9.1

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *Internation Conference on Learning Representations*, 2017. 5.3, 5.4.1

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii, October 2008. Association for Computational Linguistics. URL https://aclanthology.org/D08-1027. 8.6

Anders Søgaard. Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 682–686, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-2120. 5.5

Kevin Stowe, Jenette Preciado, Kathryn Conger, Susan Windisch Brown, Ghazaleh Kazeminejad, James Gung, and Martha Palmer. SemLink 2.0: Chasing lexical resources. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 222–227, Groningen, The Netherlands (online), June 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.iwcs-1.21. 2.3, 7

Milan Straka, Jan Hajič, and Jana Straková. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1680. 2.2

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1548. URL https://aclanthology.org/D18-1548. 1.2, 2.2, 3.2, 3.2, 3.2.1, 3.2.2, 3.2.3, 3.2.3, 6.1, 6.4

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1355. URL `https://aclanthology.org/P19-1355`. 10

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. Viable dependency parsing as sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 717–723, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1077. URL `https://aclanthology.org/N19-1077`. 2.2

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England, August 2008. Coling 2008 Organizing Committee. URL `https://www.aclweb.org/anthology/W08-2121`. 2.2, 2.3, 3.2, 3.2.1, 3.2.3

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014. 1.1, 5.1

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A. Smith. Syntactic scaffolds for semantic structures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3772–3782, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1412. URL `https://aclanthology.org/D18-1412`. 1.2, 3.2.2, 3.2.3, 6.1, 6.4

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.746. URL `https://aclanthology.org/2020.emnlp-main.746`. 8.2.1, 10

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487, Montréal, Canada, June 2012. Association for Computational Linguistics. URL `https://aclanthology.org/N12-1052`. 5.5

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12, 2013a. doi: 10.1162/tacl_a_00205. URL `https://aclanthology.org/Q13-1001`. 9.5

Oscar Täckström, Ryan McDonald, and Joakim Nivre. Target language adaptation of discriminative transfer parsers. In *Proceedings of the 2013 Conference of the North American Chap-*

ter of the Association for Computational Linguistics: Human Language Technologies, pages 1061–1071, Atlanta, Georgia, June 2013b. Association for Computational Linguistics. URL https://aclanthology.org/N13-1126. 5.1, 5.5

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41, 2015. doi: 10.1162/tacl_a_00120. URL https://www.aclweb.org/anthology/Q15-1003. 3.2.3

Dima Taji, Nizar Habash, and Daniel Zeman. Universal Dependencies for Arabic. In *Proceedings of the Third Arabic Natural Language Processing Workshop*, pages 166–176, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1320. URL https://www.aclweb.org/anthology/W17-1320. 6.3.5

Alex Tamkin, Dat Nguyen, Salil Deshpande, Jesse Mu, and Noah Goodman. Active learning helps pretrained models learn the intended task. *arXiv preprint arXiv:2204.08491*, 2022. 8.4.2

Wei Tan, Lan Du, and Wray Buntine. Diversity enhanced active learning with strictly proper scoring rules. *Advances in Neural Information Processing Systems*, 34:10906–10918, 2021. 8.2.1

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. Deep semantic role labeling with self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. 2.2, 3.2, 3.2, 3.2.1, 3.2.1, 3.2.3, 4.1, 6.1, 6.2.3

Min Tang, Xiaoqiang Luo, and Salim Roukos. Active learning for statistical natural language parsing. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 120–127, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073105. URL https://aclanthology.org/P02-1016. 8.2.2, 8.2.3, 8.3.1

Ben Taskar, Dan Klein, Mike Collins, Daphne Koller, and Christopher Manning. Max-margin parsing. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 1–8, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-3201. 3.1.1

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1452. URL https://aclanthology.org/P19-1452. 1.1, 3.2.3, 10

L Tesnière. Eléments de syntaxe structurale. 1959. 2.2

Cynthia A Thompson, Mary Elaine Califf, and Raymond J Mooney. Active learning for natural language parsing and information extraction. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 406–414, 1999. 8.3.1

Jörg Tiedemann. Cross-lingual dependency parsing with Universal Dependencies and predicted PoS labels. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 340–349, Uppsala, Sweden, August 2015. Uppsala University, Uppsala, Sweden. URL https://aclanthology.org/W15-2137. 5.5

Jörg Tiedemann and Zeljko Agić. Synthetic treebanking for cross-lingual dependency parsing. *Journal of Artificial Intelligence Research*, 55:209–248, 2016. 6.4

Erik F. Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002. URL https://aclanthology.org/W02-2024. 2.3

Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*, 2000. URL https://aclanthology.org/W00-0726. 1, 2.3

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003. URL https://aclanthology.org/W03-0419. 2.3, 4.3.1, 9.3.1

Katrin Tomanek and Udo Hahn. Approximating learning curves for active-learning-driven annotation. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2008/pdf/335_paper.pdf. 8.5.2

Katrin Tomanek and Udo Hahn. Semi-supervised active learning for sequence labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1039–1047, Suntec, Singapore, August 2009a. Association for Computational Linguistics. URL https://aclanthology.org/P09-1117. 8.3.1, 8.4.2, 9.1, 9.2.2, 9.3.2

Katrin Tomanek and Udo Hahn. Reducing class imbalance during active learning for named entity annotation. In *Proceedings of the fifth international conference on Knowledge capture*, pages 105–112, 2009b. 8.6

Katrin Tomanek and Udo Hahn. A comparison of models for cost-sensitive active learning. In *Coling 2010: Posters*, pages 1247–1255, Beijing, China, August 2010. Coling 2010 Organizing Committee. URL https://aclanthology.org/C10-2143. 8.3.2

Katrin Tomanek and Fredrik Olsson. A web survey on the use of active learning to support annotation of text data. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 45–48, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1906. 8.7.2

Katrin Tomanek, Joachim Wermter, and Udo Hahn. An approach to text corpus construction which cuts annotation costs and maintains reusability of annotated data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 486–495, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1051. 8.4.1, 8.5.2

Katrin Tomanek, Florian Laws, Udo Hahn, and Hinrich Schütze. On proper unit selection in active learning: Co-selection effects for named entity recognition. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 9–17, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1902. 8.3.1, 8.5.1, 8.6

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001. 8.2.1

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259, 2003. URL https://aclanthology.org/N03-1033. 3.1.2

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 10

Rocco Tripodi, Simone Conia, and Roberto Navigli. UniteD-SRL: A unified dataset for span- and dependency-based multilingual and cross-lingual Semantic Role Labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2293–2305, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.197. URL https://aclanthology.org/2021.findings-emnlp.197. 6.4

Yuta Tsuboi, Hisashi Kashima, Shinsuke Mori, Hiroki Oda, and Yuji Matsumoto. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 897–904, Manchester, UK, August 2008. Coling 2008 Organizing Committee. URL https://aclanthology.org/C08-1113. 9.2.1, 9.5

Akim Tsvigun, Artem Shelmanov, Gleb Kuzmin, Leonid Sanochkin, Daniil Larionov, Gleb Gusev, Manvel Avetisian, and Leonid Zhukov. Towards computationally feasible deep active learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1198–1218, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.90. URL https://aclanthology.org/2022.findings-naacl.90. 8.3.2, 8.4.1

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*, 2019. 4.3.3

Jannis Vamvas and Rico Sennrich. As little as possible, as much as necessary: Detecting over- and undertranslations with contrastive conditioning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 490–500, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.53. URL https://aclanthology.org/2022.acl-short.53. 1.1

Yuval Varkel and Amir Globerson. Pre-training mention representations in coreference models.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8534–8540, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.687. URL https://aclanthology.org/2020.emnlp-main.687. 7.3.3

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 1.1, 3.2, 3.2.1, 5.1, 5.3.1, 6.1, 7.2.1

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *Advances in neural information processing systems*, 28, 2015a. 7.2.1

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. Grammar as a foreign language. *Advances in neural information processing systems*, 28, 2015b. 10

Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269, 1967. 1.2, 2.2, 4.2.1, 4.4

Andreas Vlachos. Active annotation. In *Proceedings of the Workshop on Adaptive Text Extraction and Mining (ATEM 2006)*, 2006. URL https://aclanthology.org/W06-2209. 8.3.2, 8.5.1

Andreas Vlachos. A stopping criterion for active learning. *Computer Speech & Language*, 22 (3):295–312, 2008. 8.5.2

Thuy-Trang Vu, Ming Liu, Dinh Phung, and Gholamreza Haffari. Learning how to active learn by dreaming. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4091–4101, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1401. URL https://aclanthology.org/P19-1401. 8.2.1

Tu Vu, Minh-Thang Luong, Quoc Le, Grady Simon, and Mohit Iyyer. STraTA: Self-training with task augmentation for better few-shot learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5715–5731, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.462. URL https://aclanthology.org/2021.emnlp-main.462. 9.5

Ivan Vulić and Marie-Francine Moens. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 363–372, 2015. 5.5

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. ACE 2005 multilingual training corpus. *Linguistic Data Consortium*, 57, 2006. 4.3.1, 7.3.1, 9.3.1

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1439. URL https://aclanthology.org/P19-1439. 6.3.2, 7.3.2

Chenguang Wang, Laura Chiticariu, and Yunyao Li. Active learning for black-box semantic role labeling with neural factors. In *IJCAI*, 2017. 8.2.1, 9.5

Dingquan Wang and Jason Eisner. Fine-grained prediction of syntactic typology: Discovering latent structure with supervised learning. *Transactions of the Association for Computational Linguistics*, 5:147–161, 2017. doi: 10.1162/tacl_a_00052. URL https://aclanthology.org/Q17-1011. 5.2

Dingquan Wang and Jason Eisner. Surface statistics of an unknown language indicate how to parse it. *Transactions of the Association for Computational Linguistics*, 6:667–685, 2018a. doi: 10.1162/tacl_a_00248. URL https://aclanthology.org/Q18-1046. 5.5

Dingquan Wang and Jason Eisner. Synthetic data made to order: The case of parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1325–1337, Brussels, Belgium, October-November 2018b. Association for Computational Linguistics. doi: 10.18653/v1/D18-1163. URL https://aclanthology.org/D18-1163. 5.5

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.51. URL https://aclanthology.org/2020.emnlp-main.51. 4.4

Haoyu Wang, Hongming Zhang, Muhao Chen, and Dan Roth. Learning constraints and descriptive segmentation for subevent detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5216–5226, Online and Punta Cana, Dominican Republic, November 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.423. URL https://aclanthology.org/2021.emnlp-main.423. 4.4

Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*, 2016. 3.2.1, 6.2.3

Wenhui Wang and Baobao Chang. Graph-based dependency parsing with bidirectional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2306–2315, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1218. URL https://aclanthology.org/P16-1218. 3.1

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)*, pages 5777–5783, Hong Kong, China, November 2019b. Association for Computational Linguistics.

doi: 10.18653/v1/D19-1584. URL https://aclanthology.org/D19-1584. 7.1, 7.4

Xinyu Wang, Jingxian Huang, and Kewei Tu. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618, Florence, Italy, July 2019c. Association for Computational Linguistics. doi: 10.18653/v1/P19-1454. URL https://aclanthology.org/P19-1454. 4.4

Xinyu Wang, Yong Jiang, Zhaohui Yan, Zixia Jia, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. Structural knowledge distillation: Tractably distilling information for structured predictor. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 550–564, Online, August 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.46. URL https://aclanthology.org/2021.acl-long.46. 9.2.3

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162, Online, November 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.496. URL https://aclanthology.org/2020.emnlp-main.496. 10

Zijie J. Wang, Dongjin Choi, Shenyu Xu, and Diyi Yang. Putting humans in the natural language processing loop: A survey. In *Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing*, pages 47–52, Online, April 2021c. Association for Computational Linguistics. URL https://aclanthology.org/2021.hcinlp-1.8. 8.7.1

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. CLEVE: Contrastive Pre-training for 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 (Volume 1: Long Papers)*, pages 6283–6297, Online, August 2021d. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.491. URL https://aclanthology.org/2021.acl-long.491. 7.4

Dittaya Wanvarie, Hiroya Takamura, and Manabu Okumura. Active learning with subsequence sampling strategy for sequence labeling tasks. *Information and Media Technologies*, 6(3): 680–700, 2011. 8.3.1, 9.2.1

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022a. 10

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022b. 10

Tianwen Wei, Jianwei Qi, Shenghuan He, and Songtao Sun. Masked conditional random fields for sequence labeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Tech-*

nologies, pages 2024–2035, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.163. URL https://aclanthology.org/2021.naacl-main.163. 2.2, 4.4

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. Zero-shot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*, 2023. 1.1, 10

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. Ontonotes release 5.0. *Linguistic Data Consortium, Philadelphia, PA*, 23, 2013. 2.3, 4.3.1, 6.3.1, 7.3.3

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 323–333, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1032. URL https://aclanthology.org/P15-1032. 2.2, 3.1

Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1137. URL https://aclanthology.org/D16-1137. 10

Fangzhao Wu, Yongfeng Huang, and Jun Yan. Active sentiment domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1701–1711, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1156. URL https://aclanthology.org/P17-1156. 8.2.3, 8.4.2, 8.5.1

Shijie Wu and Mark Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1077. URL https://www.aclweb.org/anthology/D19-1077. 6.1, 6.2.1, 6.4

Zhaofeng Wu, Hao Peng, and Noah A. Smith. Infusing finetuning with semantic dependencies. *Transactions of the Association for Computational Linguistics*, 9:226–242, 2021. doi: 10.1162/tacl_a_00363. URL https://aclanthology.org/2021.tacl-1.14. 10

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8625–8646, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.764. URL https://aclanthology.org/2020.acl-main.764. 8.7.1

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Com-*

putational Linguistics (Volume 1: Long Papers), pages 1513–1528, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.107. URL https://aclanthology.org/2022.acl-long.107. 10

Min Xiao and Yuhong Guo. Online active learning for cost sensitive domain adaptation. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 1–9, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-3501. 8.6

Min Xiao and Yuhong Guo. Distributed word representation learning for cross-lingual dependency parsing. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 119–129, Ann Arbor, Michigan, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-1613. URL https://aclanthology.org/W14-1613. 5.1, 5.5

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1034. URL https://aclanthology.org/D18-1034. 5.5

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020. 1.2

Benfeng Xu, Quan Wang, Zhendong Mao, Yajuan Lyu, Qiaoqiao She, and Yongdong Zhang. kNN prompting: Beyond-context learning with calibration-free nearest neighbor inference. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=fe2S7736sNS. 9.4.2

Nan Xu, Fei Wang, Bangzheng Li, Mingtao Dong, and Muhao Chen. Does your model classify entities reasonably? diagnosing and mitigating spurious correlations in entity typing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8642–8658, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.592. 1.1

Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee. Cross-language transfer learning for deep neural network based speech enhancement. In *The 9th International Symposium on Chinese Spoken Language Processing*, pages 336–340. IEEE, 2014. 5.5

Zhao Xu, Kai Yu, Volker Tresp, Xiaowei Xu, and Jizhi Wang. Representative sampling for text classification using support vector machines. In *European conference on information retrieval*, pages 393–407. Springer, 2003. 8.2.2, 8.2.3

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238, 2005. 2.3

Nianwen Xue. Labeling Chinese predicates with semantic roles. *Computational Linguistics*, 34 (2):225–255, 2008. doi: 10.1162/coli.2008.34.2.225. URL https://aclanthology.

org/J08-2004. 2.3

Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-3212. 2.2, 3.2.3

Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machines. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 195–206, Nancy, France, April 2003. URL https://aclanthology.org/W03-3023. 2.2, 2.2

Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G Dy. Active learning from crowds. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 1161–1168, 2011. 8.6

Fan Yang and Paul Vozila. Semi-supervised Chinese word segmentation using partial-label learning with conditional random fields. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–98, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1010. URL https://aclanthology.org/D14-1010. 9.5

Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. Multi-task cross-lingual sequence tagging from scratch. *arXiv preprint arXiv:1603.06270*, 2016. 5.5

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019. 1.1, 9.1

Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. Spoken language understanding using long short-term memory neural networks. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 189–194. IEEE, 2014. 1

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA, June 1995. Association for Computational Linguistics. doi: 10.3115/981658.981684. URL https://aclanthology.org/P95-1026. 1.2, 9.1, 9.2.3, 9.5

David Yarowsky and Grace Ngai. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*, 2001. URL https://aclanthology.org/N01-1026. 1.2, 6.4

Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 93–102, 2019. 8.2.1

Yue Yu, Lingkai Kong, Jieyu Zhang, Rongzhi Zhang, and Chao Zhang. AcTune: Uncertainty-based active self-training for active fine-tuning of pretrained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1422–1436, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.102.

URL https://aclanthology.org/2022.naacl-main.102. 8.2.2, 8.4.2

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.637. URL https://aclanthology.org/2020.emnlp-main.637. 8.2.2, 8.4.2, 8.5.1

Michelle Yuan, Patrick Xia, Chandler May, Benjamin Van Durme, and Jordan Boyd-Graber. Adapting coreference resolution models through active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7533–7549, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.519. URL https://aclanthology.org/2022.acl-long.519. 8.4.2

Daniel Zeman and Philip Resnik. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, 2008. URL https://aclanthology.org/I08-3008. 5.5

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. CoNLL 2017 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-3001. URL https://aclanthology.org/K17-3001. 2.3, 3.1.2

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. CoNLL 2018 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-2001. URL https://aclanthology.org/K18-2001. 2.3, 3.1.2

Daniel Zeman, Joakim Nivre, Mitchell Abrams, and et al. Universal dependencies 2.7, 2020. URL http://hdl.handle.net/11234/1-3424. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. 6.3.4

Xiangkai Zeng, Sarthak Garg, Rajen Chatterjee, Udhyakumar Nallasamy, and Matthias Paulik.

Empirical evaluation of active learning techniques for neural MT. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 84–93, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-6110. URL https://aclanthology.org/D19-6110. 8.2.1, 8.2.2, 9.5

Xueying Zhan, Qingzhong Wang, Kuan-hao Huang, Haoyi Xiong, Dejing Dou, and Antoni B Chan. A comparative survey of deep active learning. *arXiv preprint arXiv:2203.13450*, 2022. 8.1, 8.1

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*, 2022a. 1.1

Mike Zhang and Barbara Plank. Cartography active learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 395–406, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.36. URL https://aclanthology.org/2021.findings-emnlp.36. 8.2.1

Pei Zhang, Xueying Xu, and Deyi Xiong. Active learning for neural machine translation. In *2018 International Conference on Asian Language Processing (IALP)*, pages 153–158. IEEE, 2018. 8.2.2

Rongzhi Zhang, Yue Yu, and Chao Zhang. SeqMix: Augmenting active sequence labeling via sequence mixup. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8566–8579, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.691. URL https://aclanthology.org/2020.emnlp-main.691. 8.4.2

Rongzhi Zhang, Yue Yu, Pranav Shetty, Le Song, and Chao Zhang. Prompt-based rule discovery and boosting for interactive weakly-supervised learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 745–758, Dublin, Ireland, May 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.55. URL https://aclanthology.org/2022.acl-long.55. 8.4.2, 8.7.2

Shujian Zhang, Chengyue Gong, Xingchao Liu, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. ALLSH: Active learning guided by local sensitivity and hardness. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1328–1342, Seattle, United States, July 2022c. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.99. URL https://aclanthology.org/2022.findings-naacl.99. 8.2.1, 8.4.2

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022d. 10

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for*

Computational Linguistics: Volume 1, Long Papers, pages 665–676, Valencia, Spain, April 2017a. Association for Computational Linguistics. URL https://aclanthology.org/E17-1063. 3.1, 3.1.1, 4.2.2

Ye Zhang, Matthew Lease, and Byron Wallace. Active discriminative text representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017b. 8.2.1

Yiming Zhang, Shi Feng, and Chenhao Tan. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates, December 2022e. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.622. 8.2.1

Yu Zhang, Zhenghua Li, and Min Zhang. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3295–3305, Online, July 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.302. URL https://aclanthology.org/2020.acl-main.302. 3.1.3, 4.4

Yuan Zhang and Regina Barzilay. Hierarchical low-rank tensors for multilingual transfer parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1857–1867, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1213. URL https://aclanthology.org/D15-1213. 5.1, 5.5

Yue Zhang and Stephen Clark. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii, October 2008. Association for Computational Linguistics. URL https://aclanthology.org/D08-1059. 3.1.2

Zhisong Zhang, Hai Zhao, and Lianhui Qin. Probabilistic graph-based dependency parsing with convolutional neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1382–1392, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1131. URL https://aclanthology.org/P16-1131. 3.1, 3.1.1

Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy. A two-step approach for implicit event argument detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7479–7485, Online, July 2020c. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.667. URL https://www.aclweb.org/anthology/2020.acl-main.667. 3.2, 3.2.3

Zhisong Zhang, Emma Strubell, and Eduard Hovy. Transfer learning from semantic role labeling to event argument extraction with template-based slot querying. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2627–2647, Abu Dhabi, United Arab Emirates, December 2022f. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.169. 10

Guang Zhao, Edward Dougherty, Byung-Jun Yoon, Francis Alexander, and Xiaoning Qian. Uncertainty-aware active learning for optimal bayesian classifier. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=Mu2ZxFctAI. 8.2.1

Mingjun Zhao, Haijiang Wu, Di Niu, and Xiaoli Wang. Reinforced curriculum learning on pre-trained neural machine translation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, number 05, pages 9652–9659, 2020a. 8.7.1

Yuekai Zhao, Haoran Zhang, Shuchang Zhou, and Zhihua Zhang. Active learning approaches to enhancing neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1796–1806, Online, November 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.162. URL https://aclanthology.org/2020.findings-emnlp.162. 8.2.2, 8.4.2

Yunpeng Zhao, Mattia Prosperi, Tianchen Lyu, Yi Guo, Le Zhou, and Jiang Bian. Integrating crowdsourcing and active learning for classification of work-life events from tweets. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 333–344. Springer, 2020c. 8.6

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR, 2021b. 1.1, 9.4.2, 10

Fedor Zhdanov. Diverse mini-batch active learning. *arXiv preprint arXiv:1901.05954*, 2019. 8.2.2, 8.2.3

Hao Zhou, Yue Zhang, Shujian Huang, and Jiajun Chen. A neural probabilistic structured-prediction model for transition-based dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1213–1222, Beijing, China, July 2015a. Association for Computational Linguistics. doi: 10.3115/v1/P15-1117. URL https://aclanthology.org/P15-1117. 2.2

HuiWei Zhou, Long Chen, Fulin Shi, and Degen Huang. Learning bilingual sentiment word embeddings for cross-language sentiment classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 430–440, Beijing, China, July 2015b. Association for Computational Linguistics. doi: 10.3115/v1/P15-1042. URL https://aclanthology.org/P15-1042. 5.5

Jie Zhou and Wei Xu. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1109. URL https://www.aclweb.org/anthology/P15-1109. 2.2, 3.2, 3.2, 3.2.3, 6.1, 6.2.3

Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W. Tsang, and Shen-Shyang Ho. Transfer learning for cross-language text categorization through active correspondences construction. In *Proceed-*

ings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, pages 2400–2406, 2016a. 5.1

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. LIMIT-BERT : Linguistics informed multi-task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4450–4461, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.399. URL `https://aclanthology.org/2020.findings-emnlp.399`. 10

Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. Cross-lingual sentiment classification with bilingual document representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1403–1412, Berlin, Germany, August 2016b. Association for Computational Linguistics. doi: 10.18653/v1/P16-1133. URL `https://aclanthology.org/P16-1133`. 5.5

Hua Zhu, Wu Ye, Sihan Luo, and Xidong Zhang. A multitask active learning framework for natural language understanding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4900–4914, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.430. URL `https://aclanthology.org/2020.coling-main.430`. 8.6, 9.3.4

Jingbo Zhu and Eduard Hovy. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 783–790, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `https://aclanthology.org/D07-1082`. 8.5.2, 8.6

Jingbo Zhu, Huizhen Wang, and Eduard Hovy. Learning a stopping criterion for active learning for word sense disambiguation and text classification. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*, 2008a. URL `https://aclanthology.org/I08-1048`. 8.5.2

Jingbo Zhu, Huizhen Wang, and Eduard Hovy. Multi-criteria-based strategy to stop active learning for data annotation. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1129–1136, Manchester, UK, August 2008b. Coling 2008 Organizing Committee. URL `https://aclanthology.org/C08-1142`. 8.5.2

Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K Tsou. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1137–1144, Manchester, UK, August 2008c. Coling 2008 Organizing Committee. URL `https://aclanthology.org/C08-1143`. 8.2.2, 8.2.3, 8.5.1

Jingbo Zhu, Huizhen Wang, Benjamin K Tsou, and Matthew Ma. Active learning with sampling by uncertainty and density for data annotations. *IEEE Transactions on audio, speech, and language processing*, 18(6):1323–1331, 2009. 8.2.2, 8.3.2

Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005. 1.2, 10

Zi Long Zhu, Vikrant Yadav, Zubair Afzal, and George Tsatsaronis. Few-shot initializing of active learner via meta-learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1117–1133, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.findings-emnlp.80`. 8.4.2

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1163. URL `https://aclanthology.org/D16-1163`. 5.1