Coreference Resolution for Arabic

By

Abdulrahman Mohammad Aloraini

School of Electronic Engineering And Computer Science

QUEEN MARY UNIVERSITY OF LONDON

Submitted in partial fulfillment of the requirements of the Degree of Doctor of Philosophy

August 2022

Abstract

Recently, there has been enormous progress in coreference resolution. These recent developments were applied to Chinese, English and other languages, with outstanding results. However, languages with a rich morphology or fewer resources, such as Arabic, have not received as much attention. In fact, when this PhD work started there was no neural coreference resolver for Arabic, and we were not aware of any learning-based coreference resolver for Arabic since [Björkelund and Kuhn, 2014]. In addition, as far as we know, whereas lots of attention had been devoted to the phemomenon of zero anaphora in languages such as Chinese or Japanese, no neural model for Arabic zero-pronoun anaphora had been developed. In this thesis, we report on a series of experiments on Arabic coreference resolution in general and on zero anaphora in particular. We propose a new neural coreference resolver for Arabic, and we present a series of models for identifying and resolving Arabic zero pronouns. Our approach for zero-pronoun identification and resolution is applicable to other languages, and was also evaluated on Chinese, with results surpassing the state of the art at the time. This research also involved producing revised versions of standard datasets for Arabic coreference.

Acknowledgement

I am very grateful to have had this opportunity to join the Computer Science program at Queen Mary University of London. This was both a challenging and a rewarding experience for me. In this journey, I was fortunate to meet and learn from many great people who have helped me reach this stage, and they have left me with many memories that I will cherish forever. I would first like to thank my mentor and supervisor, Massimo Poesio. I owe him a very great debt for many inspiring discussions that eventually led to this thesis. For the academic and beyond support, I would also like to thank my advisory panel, Matthew Purver and Arkaitz Zubiaga, whose valuable feedback and thoughtful comments have helped make this work better.

I would like to thank my colleagues and friends in Cognitive Science and the DALI project for their good company and camaraderie during our dissertation years and for their critiques and comments on my research and presentations. I would like to thank Juntao Yu, whom I have worked with and talked about various research and technical ideas which led to publishing a joint paper together.

I am forever grateful to my mother and father for their endless love and encouragement from a great distance away, for the countless sacrifices they have made for me, and for believing in me. I would also like to thank my brothers and sisters whose love and support sustained me throughout this journey. Last but not least, I thank my wife, Somaya, who devoted her time and effort to support me during my studies, and for taking care of me and our baby, Faris, while I was busy running experiments and attending meetings. It would not have been possible to reach this stage without all of you.

Author's Declaration

I, Abdulrahman Mohammad Aloraini, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material. I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university. The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Details of Collaboration:

- The mention detectors and the end-to-end annealing algorithm used in the Arabic coreference resolution system were developed by Juntao Yu (Section 3.4).

Publications:

- Abdulrahman Aloraini and Massimo Poesio. Cross-lingual zero pronoun resolution. In *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020b
- Abdulrahman Aloraini and Massimo Poesio. Anaphoric zero pronoun identification: A multilingual approach. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, 2020a

- Abdulrahman Aloraini, Juntao Yu, and Massimo Poesio. Neural coreference resolution for arabic. *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, 2020
- Abdulrahman Aloraini and Massimo Poesio. Data augmentation methods for anaphoric zero pronouns. *Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference*, 2021
- Abdulrahman Aloraini, Sameer Pradhan, and Massimo Poesio. Joint coreference resolution for zeros and non-zeros in arabic. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, 2022
- Wateen Aliady, Abdulrahman Aloraini, Chris Madge, Juntao Yu, Richard Bartle, and Massimo Poesio. Coreference annotation of an arabic corpus using a virtual world game. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, 2022

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Natural language processing (NLP) has achieved great progress in recent years through the use of neural networks and language models. But despite these many successes, much of NLP research is still concentrated on English. Arabic, for instance, is the 6th most spoken language worldwide and ranked 3rd as the most spoken native language (i.e., the same rank as English) [1], yet the performance on Arabic of NLP models for many tasks is lower than it is for English [Al-Ayyoub et al., 2018, Dahou et al., 2019, ElSahar and El-Beltagy, 2015, Nabil et al., 2015]. There are a number of reasons for this. Arabic is a morphologically rich language, requiring special preprocessing. Also, there are not as many high-quality labelled datasets for Arabic as there are for English. And crucial for this dissertation, Arabic is a pro-drop language (i.e., permits subject pronouns to be dropped in certain syntactic positions) which poses challenges to NLP systems [Farghaly and Shaalan, 2009, Shaalan et al., 2019]. As a result, while there has been interest and progress in some areas of Arabic NLP recently (e.g., sentiment analysis [Al-Ayyoub et al., 2019]), others have made little or no progress at all. One area of research in which not many advances have been made is coreference resolution.

Coreference resolution is the task of determining in a body of text which mentions refer to the same real-world entity or concept. Consider the following example:

أوباما زار الصين . هو يعتبر الصين من أهم المحطات في آسيا .

Obama visited China. He considers China one of the main stops in Asia.

[1] https://www.cia.gov/the-world-factbook/countries/world/#people-and-society

In the example, we have several mentions that refer to entities. A typical coreference resolution system would cluster these mentions based on their entity reference. In the example, we have two clusters. The first refers to Barack Obama, the person who served as the 44th president of the United States. The second refers to China, the country. Therefore, the following mentions Obama and He are clustered together and the two same word China are put in a different cluster.

Coreference resolution is not always straightforward. The task is typically preceded by preprocessing steps such as tokenization and mention detection. But issues often raise while tokenizing the text for Arabic. For instance, some mentions may contain affixes and clitics which require a morphological tokenizer. Also, Arabic is a pro-drop language that permits dropping subject pronouns which needs to be identified because these dropped subjects might refer to other mentions; they are known as anaphoric zero-pronoun (AZP). For instance, the pronoun he in the previous example can be dropped as:

أوباما زار الصين . * يعتبر الصين من أهم المحطات في آسيا .

Obama visited China. * Considers China one of the main stops in Asia.[2]

The asterisk * shows the gap position of the dropped subject. Arabic native speakers realize these dropped AZPs and to which mentions they refer to based on the sentence context. In the example, the verb يعتبر encodes information about the gender and number of the subject which is a masculine and singular mention, i.e Obama.

Many coreference resolution methods do not cover AZPs, because they do not have a surface realization and they do not occur in English. However, this is a limitation for multilingual applications, because they are common in pro-drop languages [Chen and Ng, 2016]. One of the main goals of this thesis is to study what are the requirements and challenges for building a complete coreference resolution framework that can resolve AZPs and non-AZPs together.

Due to the importance of the task, many initiatives have been concerned with creating coreference datasets, such as OntoNotes [Weischedel et al., 2011] and ACE [Doddington et al., 2004]. These datasets have been used for shared tasks such as CoNLL 2011 and 2012 [Pradhan et al., 2012] and CODI-CRAC-2021 [Khosla et al., 2021]. The CoNLL-2012 shared task on coreference created a standardized version of a portion of the OntoNotes corpus [Weischedel et al., 2011] for coreference resolution. This shared task covered three languages: English, Chinese and Arabic. Some CoNLL-2012 participants also considered Arabic when they evaluated their systems. However, their models reported lower results for Arabic compared with the other two languages. We intend to explore and investigate why Arabic coreference resolution has not achieved competitive results compared to

[2]The English sentence is grammatically incorrect, but only to show the equivalence translation of the Arabic sentence.

Chinese and English in CoNLL-2012, and how to overcome its problems.

In this thesis, we experiment on the modern standard Arabic (MSA) since it is standardized and has a few available datasets. Identifying and overcoming these challenges in MSA coreference resolution may provide multiple benefits, including providing a blueprint for other morphologically rich languages, especially Semitic languages. For example, modern Hebrew (MH) is a Semitic language and shares many linguistic proprieties with Arabic (e.g. has a rich morphology and permits AZPs) [Kamir et al., 2002]. However, MH does not have as many resources as MSA. As far as we know, there has been no research study on MH coreference resolution and its AZPs. Arabic dialects suffer from the same challenges. Arabic dialects are based on MSA and have not been investigated. Semitic languages and Arabic dialects might benefit from our proposals and discussions.

1.2 Research Questions

The goal of this PhD is to study Arabic coreference resolution, which raises several research questions:

RQ1: Is end-to-end coreference resolution feasible with the corpora of more limited size that we have for Arabic?

The question is the main motivation of the thesis. There has been various attempts to tackle Arabic coreference resolution. For instance, some CoNLL-2012 participants also tried Arabic together with the two other languages (i.e. Chinese and English). But these participants reported low results for Arabic. The main obstacle was the limited size of the Arabic corpus, which is only 1/3 in size of the Chinese and English versions. Another problem was the features used in these systems. Participants engineered features mainly for Chinese or English, not for Arabic. But Arabic, Chinese and English belong to different families and are very different languages, especially regarding their morphological complexity[3]. Even though the three languages share some characteristics, many features are only applicable to one language. Arabic is a Semitic and morphologically rich language which needs a system to tackle its morphological and linguistic proprieties; in fact, Arabic is considered as one of the most difficult languages for parsing [Tsarfaty et al., 2010]. Recent language models and end-to-end systems have addressed these issues to some extent. BERT, for example, learns surface, semantic and syntactic features without relying on expert feature designers [Jawahar et al., 2019]. End-to-end systems do not require specialized parsers and named entity extractors for a language. But although end-to-end systems have been shown to perform very well for NLP tasks, they are difficult to train without large size corpora. The ability for a coreference model to perform well without any hand-coded features and on a limited size corpus would thus be very appealing for

[3] Arabic has a complex morphology, English and Chinese have simpler morphology [Pradhan et al., 2012].

Arabic coreference resolution. The aim of this thesis is to explore the problem of training recent end-to-end models on Arabic. As far as we know, there was no neural resolver for Arabic prior to this work. In Chapter 3, we show our results and insights using the recent neural models with various settings. We also show the current limitations and how to tackle them.

RQ2: Can neural network models learn how to identify and resolve Arabic Anaphoric zero-pronouns (AZPs)? And do language models encode AZP information in their layers?

AZPs are common in pro-drop languages; however, they have not been always considered in previous coreference resolution proposals, for three reasons. First, the target language of most neural coreference models is English [White, 1985], which is not a pro-drop language. Second, AZPs are null arguments and the focus in coreference is usually on realized arguments [Lee et al., 2005]. Third, papers such as [Iida et al., 2015] have shown that treating the resolution of AZPs and realized mentions separately is beneficial. Two subtasks of AZP resolution have emerged in the literature: AZP identification and AZP resolution. The two tasks have been investigated for languages such as Chinese, Japanese and Spanish. However, there has been no research on these tasks for Arabic. We report our results for AZP identification in Chapter 4 and for AZP resolution in Chapter 5. We propose methods and evaluate on the Arabic portion of OntoNotes. However, the number of annotated Arabic AZPs in OntoNotes is small and from a single domain, newspaper texts. To see the efficiency of our proposed methods and if they can be generalized, we extend our models and evaluate on Chinese AZPs (the Chinese portion has a larger number of AZPs and covers different domains). In addition, we study the effects of AZPs on language models, an interesting area of study and related to the new field of 'BERTology'[4]. Language models learn linguistic features by masking words in the pre-training phase. However, AZPs do not have any surface realization (they are null arguments) so it is not possible to mask them. We provide evidence that information about AZPs is encoded within their context and language models learn of their existence implicitly. In fact, these encoded features can be used in neural network to identify AZPs and resolve them to their true antecedents. We discuss the current challenges for identifying and resolving Arabic AZPs and how can we possibly improve the tasks.

RQ3: Can we automatically generate data for AZPs without expert annotators?

One of the main obstacles for Arabic coreference resolution is the size of the available datasets, especially for some mention types. There have been various datasets for pronominal anaphora, such as QurAna [Sharaf and Atwell, 2012], AnATAr [Hammami et al., 2009] and others [Seddik et al., 2015]; however, there has been little annotation of AZPs. As far as we know, OntoNotes [Weischedel et al., 2011] is the only publicly available dataset that includes Arabic AZPs, but their total number is small. While it is possible to annotate AZP samples manually, creating really large datasets for this task would be

[4] A term refers to the study of language-model components.

expensive and time-consuming. Being able to do this annotation automatically would be much cheaper and faster.

Chapter 6 discusses data augmentation methods we developed to find and generate AZP samples from open text and identify their true antecedents automatically. We show that the automatically generated data improves the results on the AZP identification and resolution tasks. We also discuss how to generalize the augmented dataset to work for pronominal anaphora and coreference resolution.

RQ4: (Realized) Mentions and AZPs are different in nature: mentions have surface realizations while AZPs not (i.e. they are null arguments). Is it possible to cluster these two types of anaphoric expressions in one learning framework?

In **RQ2**, we saw three reasons why AZPs have not been handled by existing coreference resolution systems. Therefore, mentions and AZPs have been handled by different models, not jointly. Recently, however, it has been shown that it is possible to resolve them together [Chen et al., 2021b]. Recent language models have made it possible to learn both anaphors together because they can provide embeddings for gaps, in addition to words and characters. In fact, we will see in Chapters 4 and 5 that BERT encodes information about AZPs within its layers. But only a few joint models of this type have been proposed, and only for Chinese. In Chapter 7, we apply one of these proposals to Arabic and we also propose two methods to learn clustering mentions and AZPs together. Our method surpasses the existing methods when we cluster the two types.

1.3 Contributions

Our contributions are in the area of Arabic coreference resolution and its applications and include: a neural coreference resolver, models for AZP identification and AZP resolution, data augmentation methods for AZPs, and two coreference resolution frameworks that resolve AZPs and non-AZPs jointly.

A neural coreference resolver: starting from the neural coreference architecture of [Lee et al., 2018], we adapt it to Arabic using AraBERT [Antoun et al., 2020], a separate mention detector, an annealing algorithm and pre-processing techniques. We evaluate on OntoNotes and apply various ablation settings. We analyse the errors and show the existing limitations. In addition, we suggest different ways to progress in the area. This is the first neural resolver for Arabic on a limited corpus size and achieves state-of-the-art results.

Anaphoric zero pronoun detector and resolver: there has been no research on developing neural models for Arabic anaphoric zero-pronouns (AZPs) and its two sub-tasks: AZP identification and resolution. We propose multilingual methods for the two tasks and evaluate them on Arabic and Chinese. As far as we know, we are the first to develop neural

models for AZP resolution for Arabic, and our results surpass the state-of-the-art systems on Chinese as well.

Automatic methods to generate anaphoric zero pronoun samples: limited corpus data are available for Arabic coreference, and even more so for anaphoric zero pronouns. We show five methods for generating AZP samples automatically using Wikipedia articles. We also suggest a simple yet effective way to find AZP true antecedents. The collected data has improved AZP two tasks.

Models for resolving zero and non-zero anaphors jointly: most existing proposals focus on either zero or non-zero mentions. There have been very few proposals that have considered both, and these were only evaluated on Chinese. We review their approaches and apply one of them on Arabic. In addition, we suggest two methods for learning the two mentions types together: a joint-learning and a pipeline. Our joint-learning approach achieves higher results than the pipeline and the baseline when considering AZP and non-AZPs.

1.4 Publications

- Abdulrahman Aloraini and Massimo Poesio. Cross-lingual zero pronoun resolution. In *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020b
- Abdulrahman Aloraini and Massimo Poesio. Anaphoric zero pronoun identification: A multilingual approach. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, 2020a
- Abdulrahman Aloraini, Juntao Yu, and Massimo Poesio. Neural coreference resolution for arabic. *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, 2020
- Abdulrahman Aloraini and Massimo Poesio. Data augmentation methods for anaphoric zero pronouns. *Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference*, 2021
- Abdulrahman Aloraini, Sameer Pradhan, and Massimo Poesio. Joint coreference resolution for zeros and non-zeros in arabic. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, 2022
- Wateen Aliady, Abdulrahman Aloraini, Chris Madge, Juntao Yu, Richard Bartle, and Massimo Poesio. Coreference annotation of an arabic corpus using a virtual world game. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, 2022

Chapter 2

Background

In this Chapter, we provide essential background on Natural Language Processing (NLP), including some sub-tasks key to coreference such as tokenization, mention detection, and language models in Section 2.1. We overview Arabic and its morphology in Section 2.2 and Arabic NLP in Section 2.3. We define the following tasks: coreference resolution, anaphoric zero pronoun (AZP) identification and AZP resolution, and their related works in Sections 2.4 and 2.5 respectively. We discuss the existing datasets relevant to the study and show which ones we employ in Section 2.6.

2.1 Natural Language Processing

NLP involves a series of interpretive steps, which depends to some extent on the task. Coreference resolution typically involves the following three steps:

Tokenization →Mention Detection →Coreference Resolution

Tokenization divides text into tokens which then can be passed to a mention detector where entities are identified. Coreference resolution finds mentions that belong to the same entity and cluster them together. There can be other steps between these three, such as, Part-of-Speech tagging which can be useful for some mentions (e.g., anaphoric zero pronouns). Each step has its own obstacles and tackling them is essential to achieve high quality results.

2.1.1 Tokenization

Tokenization is the process of segmenting text into sentences and words. It is the initial step in NLP tasks. English, for example, is usually segmented by spaces for words and full stops for sentences (if there are multiple sentences):

> Input: Queen Mary University is situated in London.
> Output: [Queen] [Mary] [University] [is] [situated] [in] [London] [.]

English is considered one of the most straightforward languages for tokenization. However, English tokenization still faces many issues. Examples of these issues are segmenting abbreviated words (e.g., Mr.) and contracted words (e.g., it's).

Tokenization is challenging for languages that do not use spaces for word boundaries and for non-Latin alphabet languages. Tokenization can be more challenging for morphologically rich languages (e.g., Arabic and Hebrew) because they express multiple levels of information already at the word level (i.e., that also known as morphemes). These morphemes should be segmented independently for analysis to understand their rule in the sentence [Guo, 1997]. In Arabic NLP section, we will discuss how to tokenize Arabic text properly.

2.1.2 Mention Detection

Mention detection is the task of identifying entities in text. The mention detector recognizes mentions of entities in texts, whether consisting of a single token or multiple tokens. Consider the following example:

> Input: Queen Mary University is situated in London.
> Output: [Queen Mary University] [London]

Mention detection is an important step for many NLP tasks besides coreference resolution– including, e.g., named entity recognition and relation extraction [Yu et al., 2020]. In the past, rule-based methods were prevalent for mention detection for coreference. The Stanford mention detector [Lee et al., 2013] was used in many coreference resolution systems until the emerge of end-to-end models. The Stanford mention detector uses various heuristic rules to identify mentions using string match, pronoun match, and other information as shown in Figure 2.1. Many participants in CoNLL-2012 applied the Stanford mention detection for Arabic; however, string match sieve was the only one suitable. After the release of the end-to-end coreference resolution system of [Lee et al., 2017], coreference resolution proposals have started learning how to detect mention spans while learning their coreference clusters.

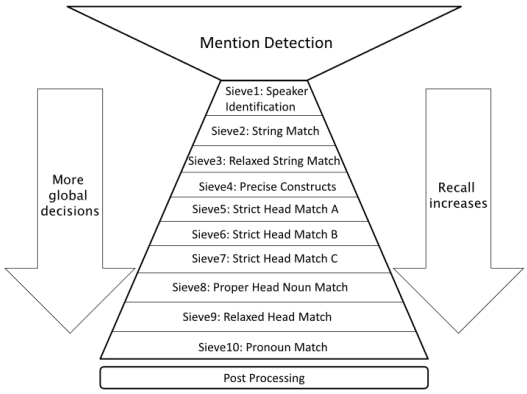

Figure 2.1: Stanford rule-based mention detector [Lee et al., 2013]

Recently, Yu et al. [2020] investigated neural mention detectors on coreference resolution and named entity recognition tasks. They compared three settings of mention detectors. The first is a modified version of Lee et al.'s [2018] system. The second is a bi-directional LSTM with a biaffine attention of Dozat and Manning [2017]. The third uses BERT representations into a feed-forward neural network to detect mentions. They have also shown that separating the mention detection and coreference resolution is beneficial. In Chapter 3, we will show these three settings on Arabic text and also training a separately mention detector is actually better than training it jointly with the coreference resolution.

2.1.3 Language Models

Early language models learned word vector representations by capturing the distributed syntactic and semantic properties, such as, Word2vec [Mikolov et al., 2013], GloVe [Pennington et al., 2014], and FastText [Bojanowski et al., 2016]. However, these models suffered many problems. One serious limitation is how these models fail to learn word meanings based on the context. Consider the following two sentences:

I **play** football everyday.
The **play** is based on a real-life event.

The word "play" has two completely different meanings. In the first sentence, it means to act while in the second means a literary performance on the stage. Early models would assign a single fixed representation for such cases. Another limitation of these early models is that they fail to represent untrained words (i.e., words that were not part of the pre-training and also known as unknown words). Recent pre-trained language models have led to great advances in NLP understanding, achieved state-of-the-art performances in various NLP tasks and addressed previous limitations in early word embedding systems [Qiu

et al., 2020]. For example, ELMo [Peters et al., 2018] proposed generating deep contextualized representations based on the context. ELMo succeeded in representing words with respect to context and semantic for general-purpose tasks; however, ELMo is used as an additional feature and cannot be fine-tuned to a specific task [Devlin et al., 2018]. In addition, ELMo was initially pretrained solely on English which delayed NLP advancements for many languages, including Arabic.

An alternative approach is applying language models by fine-tuning them for a specific task using a small number of examples and this has resulted in remarkable improvements [Radford et al., 2018]. This approach pre-trains a large corpus to learn word vectors generally and language model's parameters are optimized for a target task. However, to learn a good quality of general representations, language models require large datasets from various domains to learn word representations before they applied to a specific task. Also, this requires expensive computational resources and utilizes parallelization of training which is an important factor when working with large amounts of data. Transformers [Vaswani et al., 2017] remedy this gap and allows for shorter training time and for concurrent resource processing. BERT [Devlin et al., 2018] is a recent language model that exploits the advantage of Transformers and stacks multiple ones to learn a general-purpose representations for words. BERT also provides a trivial approach for fine-tuning and this has enabled massive advances to many NLP tasks, Figure 2.2 shows how BERT is applied to other tasks.

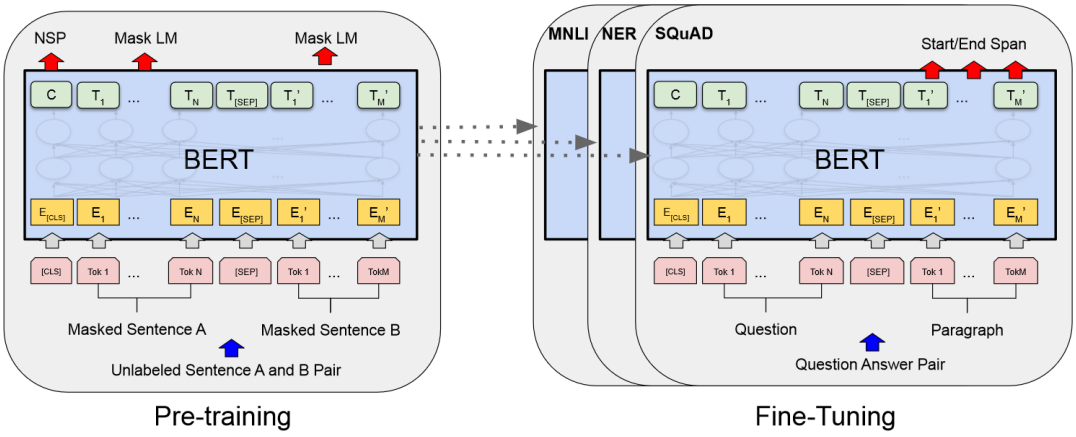

Figure 2.2: BERT is pretrained generally then applied to specific-tasks [Devlin et al., 2018].

There are two versions of BERT: BERT-base and BERT-large. The difference between these versions is the number of stacked Transformers and hidden units. It has been shown that larger language models tend to learn better quality words representations [Tenney et al., 2019]. BERT was initially pretrained for English in one model, and for multiple languages in another model (Multilingual). In the next section, we look in details how BERT works.

BERT

BERT is a language representation model consisting of multiple stacked Transformers [Vaswani et al., 2017]. BERT was pretrained on a large amount of unlabeled text, and produces distributional vectors for words and contexts. BERT uses two objectives to learn word and context representations: masked-language modeling (MLM), and next sentence prediction (NSP). BERT is pre-trained by feeding two sentences which are separated by [SEP] tag. In MLM objective, BERT masks a few words in the input with the tag [MASK][1]. In NSP, BERT predicts whether the two sentences of input are consecutive or not.

BERT has two modes of adaptation: feature extraction and fine-tuning. Feature extraction (also called feature-based) is when BERT representations are used as they were originally pretrained, without any further training, same way as ELMo. Fine-tuning is the process of slightly adjusting BERT's parameters for a target task. Feature extraction is computationally cheaper and might be more suitable for some NLP tasks. BERT was pre-trained on different settings, we used Multilingual BERT-base and AraBERT [Antoun et al., 2020] for Arabic coreference resolution, Anaphoric Zero-Pronoun identification and resolution tasks. Also for automatically creating AZP samples.

Mutlilingual BERT has been trained on English and also on other languages. However, it has been shown that pretraining BERT for a single language tend to have better results on NLP tasks. When pretraining multiple languages together, models learn little data representation because of the small language specific vocabulary. The process of training multiple languages can be beneficial for languages that share similar structure and vocabulary [Conneau et al., 2019]. This is not the case for Arabic. Arabic is a Semitic language and differ in morphological and syntactic structure, also letters with other languages, including Latin-based languages. There are other languages Semitic languages that belong to the same language family as Arabic, but they use different script system (e.g, Aramaic). Therefore, the existence of a language model pre-rained specifically for Arabic is essential to advance in the area, as shown in AraBERT experiments. We will talk about Arabic and its proprieties in the next section and explain the differences and difficulties in comparison with other languages, especially, English.

2.2 Arabic

Arabic is widely spoken in the Middle East and certain parts of Africa–21 countries–and the total number of its speakers is approximately 420 million. It is also one of the six official languages of the United Nations. Arabic alphabet contains 28 letters, but can be extended to 90 considering all different shapes and attached vowels [Tayli and Al-

[1] Devlin et al. [2018] used a 15% probability of masking each token.

Salamah, 1990]. Arabic can refer to any of these three [Elmahdy et al., 2009]:

- Classical (Quranic) Arabic: the oldest Arabic version and used mainly in reading or reciting religious texts.
- Modern Standard Arabic (MSA): derived from the classical Arabic and differs slightly in grammar, vocabulary and pronunciation. MSA is the current standard version of Arabic and used in formal writing and speeches.
- Dialectal (colloquial) Arabic: the natural spoken language in day-to-day activities. There are many dialects, one possible categorization is in Figure 2.3. Dialects differ in their grammar structure, vocabulary and some are influenced by other languages, such as, the influence of French and Amazigh on Algerian and Moroccan dialects [Harrat et al., 2018]. Some dialects share many similarities with MSA and some are very different. Dialects are not standardized. Therefore, the dialects are usually harder to work on for NLP tasks and annotations [Zaidan and Callison-Burch, 2014].

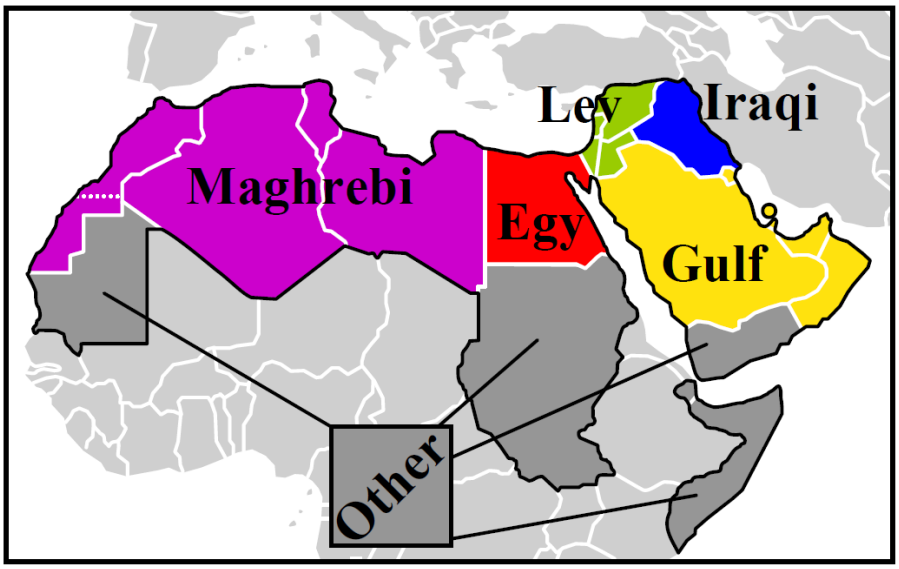

Figure 2.3: [Habash, 2010] classifies Arabic into 6 groups: Gulf, Iraqi, Levantine, Egyptian, Maghrebi, and other.

Understanding Arabic morphology is essential in Arabic NLP. Morphology is the study of words and how they are structured [Ritchey, 1998]. Morphological basic unit is morpheme, which is the smallest component that has a meaning in a language. For example, the English word ”prehistoric” contains two morphemes: ”pre” and ”historic”. Arabic is a morphological rich language that involves both orthography and syntax [Althobaiti et al., 2014b]. There are three basic types of morphemes that form words:

- Stem: the core part of a word [Payne, 2006]. For example, the words مدرسة, مدرس, دارس, دراسة mean teacher, school, research study and scholar respectively. They all share one stem (درس, to study).

- Affixes: there are three types: prefixes, suffixes, and circumfixes. Prefixes appear before the stem, suffixes after the stem, and circumfixes enclose the stem. Multiple affixes can be within a word. For example, يدرسون means ”they study” and it contains one circumfix. يذهب and ذهبوا are two examples of a prefix and suffix and they mean ”he goes” and ”they went” respectively.

- Clitics: clitics are morphemes that can be linked with a stem or affixes. There are two types: proclitics and enclitics. Proclitic morphemes appear at the beginning of words. Enclitic morphemes appear at the end of words. For example, وسيدرسونها means ”and they will study it” and it has one proclitic and one enclitic.

The difference between affixes and clitics is that clitics are considered grammatically independent. For example, the ending ـهم in the word سيارتـهم is a clitic and it means *their*, a possessive pronoun. The ending ـت in قالت is an affix inflection for a singular female which means *said*.

English nouns can be inflected for number and some nouns are specific for a gender (e.g., actor and actress). Arabic is more general where every noun can encode more details about gender, number and grammatical case as the following:

- Gender: nouns can be masculine or feminine, no neural option. For example, the word (طالب, male student) and the word (طالبة, female student).

- Number: nouns can be singular, dual, or plural. For example, the word (طالب, student), (طالبان, two students), and (طلاب, students). There are two types of plurals: sound plural and broken plural. Sound plurals are formed by adding suffixes to singular nouns. For example, معلمات, طالبات both have the suffix ات which indicates a sound plural for feminine nouns. The words معلمون, مدرسون end with the suffix ون a sound plural for masculine nouns. Broken plural is irregular and common (e.g., قلوب / hearts and أبواب / doors) .

- Grammatical cases: can be nominative, genitive, or accusative. A noun is nominative when it is the subject, genitive when it is the object of a preposition, and accusative when it is the object of a verb. Small markings are attached at the end of the noun to show its grammatical case. For example, (مدرسةُ, مدرسةِ, مدرسةَ) are nominative, genitive, and accusative respectively. They all mean ”school”, but have different pronunciation and markings at the end.

Arabic verbs can be conjugated from either trilateral or quadrilateral roots (e.g. كتب and تزلزل). Verbs are capable of carrying various information due to their richness in form and meaning which are [Shaalan et al., 2015]:

- Person: Verbs can be in the first person (أنا قرأت , I read), second (أنت تقرأ , you read), or third person (هو يقرأ , he reads).

- Gender: Verbs can tell about the subject's gender whether it is masculine (هو أكل , he ate) or feminine (هي أكلت , she ate).

- Number: Verbs can be in singular, dual or plural. For example, the singular form of the verb (to go, ذهب) is (هو يذهب , he goes), the dual is (هما يذهبا , they go), and the plural is (هم يذهبوا , they go).

- Tense: Verbs can encode the tense whether it is in the past, present or future. For example the verb (to go, ذهب) in the singular first person become (أنا ذهبت , I went) in the past tense, in the present tense (أنا اذهب , I go), and in the future tense (أنا سأذهب , I will go).

- Voice: Just like English, Arabic has two voices: active and passive. Diacritics are used to show which voice is used. An example of active voice is (هي أَكلت الكيكة , she ate the cake) and of passive voice is (الكيكة أُكلت , the cake was eaten).

- Mood: There are four moods for Arabic verbs: indicative, subjunctive, jussive, and energetic. Diacritical marks are placed on verbs to state which mood of the four is used.

Because of these verbal inflections, verbs can drop subjects (i.e. null arguments) and some of these arguments are anaphoric [Altamimi, 2015]. Another aspect of Arabic is its lexicon size. Its lexicon size is huge 1.76 times the size of English lexicon. The main reason is the heavy usage of clitics and affixes within Arabic words. It has been shown that the total number of unique words is about 2.2 million words in a 600 million word corpus while English was about 1.26 million [Alotaiby et al., 2009].

2.3 Natural Language Processing for Arabic

2.3.1 Arabic Preprocessing

Training on Arabic texts that are not preprocessed properly can suffer from sparsity (i.e., various forms for the same word) and ambiguity (i.e., same form corresponding to multiple words). There are two reasons for these problems. First, certain letters can have different forms, such as the letter "alif". The letter has various forms as shown in the following table:

Alif variation	Name	Example
ا	alif	امرأة / woman
أ	alif with hamza	أسد / Lion
إ	alif with hamza below	إبرة / Needle
آ	alif with maddah	آية / Verse

Table 2.1: Various forms of the letter ”alif” with examples.

Many Arabic speakers use correctly and incorrectly the letter ”alif”. Alkhatib et al. [2020] have shown that spelling error of ”alif” is prevalent in Arabic. For example, writing the word ”Lion” أسد can be misspelled and written as اسد . Even though the misspelled word can be common and understood by native speakers, the language model would assign two different representations. One for the actual correct word and another for the misspelled one. These errors can increase sparsity and can create noise in the learning process [Singla et al., 2014]. The second problem is the placement of diacritics on words which are assumed to be undiacritized [Habash and Sadat, 2006]. In formal media such as religious books, words have diacritics for every letter. However informally, Arabic speakers usually write without diacritics which makes Arabic text highly ambiguous [Attia, 2008]. Arabic has five diacritics which are:

Diacritic	Name	Example
ـُ	Dammah	صُحف / newspaper
ـَ	Fathah	جَزيرة / Island
ـِ	Kasrah	مِفتاح / Key
ـْ	Skoon	مدْرسة / School
ـّ	Shaddah	معلّم / Teacher

Table 2.2: All Arabic diacritics with some examples.

Diacritics are important and battle ambiguity in some words that have same letters but different diacritics (e.g., الجَدّ / ”grandfather” and الجِدّ / ”seriousness”)[2]. However, using diacritics can increase the lexical size dramatically. Some diacritics can also be used together or doubled (Tanween) for special grammatical cases. Removing diacritic marks

[2]more of these words at https://www.alriyadh.com/1832139

is beneficial to Arabic NLP because it decreases lexical sparsity and ambiguity [Habash and Sadat, 2006].

Therefore, we follow the steps proposed in [Althobaiti et al., 2014a] to pre-process the data. These steps include:

- Normalizing the various forms of the letter "alif" (آ,أ,إ) to the letter "ا".
- Removing all diacritic marks.

We show an example of an original and pre-processed sentence from OntoNotes 5.0 in Table 2.3. Pre-processing the data increases the overall performance of coreference system with 7 percentage points more, as we will see in the coreference resolution experiments, Chapter 3. We also pre-process the Arabic texts for the AZP identification, AZP resolution, and other models, in Chapters 4, 5, 6, and 7.

Original text	إِلى ذلِكَ كَتَبَت مِئاتِ المَقالاتِ النَقْدِيةِ الأَدَبِيةِ
pre-processed text	الى ذلك كتبت مئات المقالات النقدية الادبية

Table 2.3: An example on how we pre-process Arabic text. The letter "alif" is normalized and all diacritic marks are removed.

2.3.2 Arabic Tokenization

Arabic text tokenization is challenging because of its complex morphology. If we try to tokenize Arabic texts as we tokenize English, we would fail to detect crucial parts. Consider for example the sentence:

Input: ذهب أحمد إلى المدرسة. أخذه أبوه إليها.
Output: [ذهب] [أحمد] [إلى] [المدرسة] [.] [أخذه] [أبوه] [إليها] [.]
Translation: [Ahmed] [went] [to] [school] [.] [His-father] [took-him] [to-it] [.]

Some Arabic morphemes are concatenated together in one word, and we represent such cases using dash (-) in the English translation. If we follow the same approach for English words (tokenizing based on spaces and dots) to segment Arabic text, the tokenizer fails to segment the morphemes, such as, the possessive pronoun (ه /his) and object pronoun (ها /it). These morphemes should be identified individually when we employ the mention detection and coreference resolution. An inefficient tokenizer can negatively impact the results of any Arabic NLP task. Therefore, a better tokenizer would segment the previous example as the following:

Input: ذهب أحمد إلى المدرسة. أخذه أبوه إليها.

Output: [ذهب] [أحمد] [إلى] [المدرسة] [.] [أخذ] [ه] [أبوه] [إلي] [ها] [.]
Translation: [Ahmed] [went] [to] [school] [.] [father] [his][3] [took] [him] [to] [it].

There has been various proposals for Arabic text segmentation. Farasa [Abdelali et al., 2016] and Madamira [Pasha et al., 2014] are morphological tokenizers made specifically for Arabic. They were trained in a corpus segmented based on morphemes. Byte-Pair Encoding (BPE) [Bostrom and Durrett, 2020], WordPiece [Wu et al., 2016] and SentencePiece [Kudo and Richardson, 2018] are subword tokenizers that can detect morphemes occasionally, but they might violate morpheme boundaries and produce misleading segmentations. For pre-training language models, morpheme-based and subword tokenizers relatively share similar performance in many NLP tasks [Abdul-Mageed et al., 2021, Antoun et al., 2020]. It has been shown that Arabic tokenizers have high impact on mention detection and a good quality tokenizer based on morphemes surpasses other types for coreference resolution [Zitouni et al., 2005]. In our experiments, we use SentencePiece and Madamira [Obeid et al., 2020] tokenizers. We applied SentencePiece in our experiments in Chapters 3, 4, 5 and 7. We used Madamira tokenizer in Chapter 6 to tokenize Wikipedia sentences.

2.3.3 Arabic Mention Detection

Mention detection for Arabic can be challenging. Because of Arabic complex morphology, some Arabic mentions can be attached to other words (e.g., clitics). Consider the following example:

Input: ذهب أحمد إلى المدرسة. أخذه أبوه إليها.
Output: [أحمد] [المدرسة] [ه] [ه] [ها]
Translation: [Ahmed] [school] [his] [him] [it]

The first pronoun (ه, his) and the second pronoun (ها , it) are attached as clitics to other words (أبو , father) and (إليه , to); however, the pronouns should be recognized independently. Detected mentions depend highly on how the words are segmented. There has been very few studies dedicated for Arabic mention detection. Benajiba and Zitouni [2010] studied four types of mention detection. First, character-level segmentation where they segmented each letter as a token. Second, morphological-level which is tokenizing words based on their morphemes. Arabic TreeBank (ATB) segmentation is the third type where they segmented words based on their affixes. Fourth, punctuation-level where words were segmented based on the punctuation marks. They found that morphological-

[3] Arabic possessive pronoun appears at the end of its possessor word, in the example 'father'.

level and ATB tokenizers to provide the best representations for mentions in many cases, and the morphological-level outperforms ATB segmentation in many NLP tasks generally. Habash and Sadat [2006] also tried several segmentation methods and the effect of mention detection on machine translation. They found Arabic Tree Bank(ATB) segmentation detected many mentions correctly and gave better results than other methods. Benajiba et al. [2008] applied ATB segmentation to detect mention and evaluated on named-entity-recognition task, and reported great results. Recently, Yu et al. [2020] have proposed several neural mention detectors and showed their ability to detect English mentions, achieving state-of-the-art results. Therefore, we decided to experiment their models on Arabic. In Chapter 3, we show that their systems also detect accurately many Arabic mentions and improves the coreference resolution results.

2.3.4 Arabic Language Models

Unlike English, there have been a few studies devoted to Arabic language models. There might be two reasons. First, Arabic texts require many pre-processing steps and require solid background knowledge of the language. Second, the number of existing clean datasets are still small, most existing texts are from social network and they are extremely noisy. The earliest language model that addresses these two issues is AraVec [Soliman et al., 2017]. AraVec is a word2vec-based model that was pre-etrained on pre-processed tweets, World Wide Web, and Wikipedia. AraVec showed a few steps on how to prepare the texts which helped to reduce the noise of Arabic representations. However, AraVec suffered similarly from the same limitations that existed in the early English language models (polysemous/homonymous words and unknown words). Another attempt is the Arabic version of FastText, but this also suffered from the same limitations as AraVec. In addition, Arabic FastText was pretrained on noisy and un-pre-processed Arabic texts and this explains why Fasttext results fall behind AraVec in some NLP tasks [Bensoltane and Zaki, 2021]. AraBERT [Antoun et al., 2020] is an Arabic language model based on BERT. It was pre-trained on a collection of Wikipedia and newspaper articles and it has led to great advancement in Arabic NLP tasks. In our experiments, we apply AraBERT and also multilingual-BERT.

2.4 Coreference Resolution

Coreference resolution is the task of grouping mentions in a text that refer to the same real-world entity into clusters [Poesio et al., 2016] . Coreference resolution is a difficult task that requires reasoning, context understanding, and background knowledge of real-world entities, and has driven research in both natural language processing and machine learning, particularly since the release of the OntoNotes multilingual corpus providing

annotated coreference data for Arabic, Chinese and English and used for the 2011 and 2012 CoNLL shared tasks [Pradhan et al., 2012]. In addition to tokenization, which is a general prerequisite to many NLP tasks, coreference resolution can be further divided into two subtasks–mention detection and mention clustering–as illustrated in Figure 2.4.

1. Mention Detection	Obama nominated Clinton as his secretary of state on Monday. He chose her because she had foreign affairs experience.
2. Mention Clustering	Obama nominated Clinton as his secretary of state on Monday. He chose her because she had foreign affairs experience.

Figure 2.4: The first step in coreference resolution is mention detection. The detected mentions are underlined. The second step is mention clustering. We have two clusters {Obama, his, he} and {Clinton, her, she}. The mention detector might identify other words as mentions, but for simplicity we present only the mentions of the two clusters and we do not show the tokenization step which is a pre-step.

Since the release of OntoNotes, there has been substantial research on English and Chinese coreference, leading to a significant increase in the performance of coreference resolvers for English. By contrast, there has been almost no research on Arabic coreference; the performance for Arabic coreference resolution has not improved much since the CoNLL 2012 shared task, and in particular no neural architectures have been proposed–the current state-of-the-art system remains the model proposed in [Björkelund and Kuhn, 2014].

2.4.1 Types

We offer background definitions of coreference types, and focus on the ones that were considered as part of OntoNotes annotation[4]. We explain each type and provide an example.

- **Noun Phrases (NP):** Any definite and indefinite NP can be coreferenced.

  Input: [كوفي أنان] قام بإلقاء خطاب . [هو] أيضا شدد على أهمية الحوار .

  Translation: [Kofi Annan] gave a speech. [He] also emphasized on the importance of dialogue.

- **Verbs:** Some verbs can be coreferenced.

  Input: الأسعار [ارتفعت] مؤخرا . [هذا الارتفاع] هو من التضخم .

[4]Some types of anaphora were not considered in OntoNotes, such as, *Bridging* and *Plural* anaphors.

Translation: Prices [increased] recently. [This increase] is due to inflation.

- **Pronouns:** Pronouns can be categorized into:

  1. Overt Pronouns: In Arabic, there are three types of pronouns: subject, object, and possessive pronouns. Subject pronouns appear as a separate mention while object and possessive appear as an affix attached to nouns, verbs, and/or prepositions. We show all these pronouns in Table 2.4.
  2. Zero Pronouns: in pronoun-drop languages (e.g., Arabic and Italian), some pronouns can be omitted in certain syntax positions and appear as gaps.

     Input: [بوش] قام بزيارة فرنسا . و [*] تكلم عن الاحتباس الحراري

     Translation: [Bush] visited France. And [he*] talked about global warming.

Subject Pronouns		Object Pronouns		Possessive Pronouns	
أنا	I	ـي	Me	ـي	My
أنت	You	ـك	You	ـك	Your
هو	He	ـه	Him	ـه	His
هي	She	ـها	Her	ـها	Her
أنتما	You (d.)	ـكما	You (d.)	ـكما	Your (d.)
هما	They (d.)	ـهم	Them (d.)	ـهم	Their (d.)
نحن	We	ـنا	Us	ـنا	Our
هم	They (p., m.)	ـهم	Them (p., m.)	ـهم	Their (p., m.)
هن	They (p., f.)	ـهن	Them (p., f.)	ـهن	Their (p., f.)
أنتم	You (p., m.)	ـكم	You (p., m.)	ـكم	Your (p., m.)
أنتن	You (p., f.)	ـكن	You (p., f.)	ـكن	Your (p., f.)

Table 2.4: The Table shows Arabic pronouns and their English translation.

- **Generic Mentions:** Generic nouns that can be linked with other expressions.

  Input: [اجتماعات] مهمة مع الرئيس . [هذه الاجتماعات] ستتطور الاقتصاد .

  Translation: Important [Meetings] with the president. [These meetings] will improve economy.

- **Pre-modifiers:** Modifiers are coreferenced only if they are proper nouns.

Input: سياسة [الأمم المتحدة] تقوم على نبذ الحرب . وتحرص [الأمم المتحدة] على السلام العالمي .

Translation: [UN] policy condemns wars. [UN] also supports world peace.

- **Temporal expressions:** When a mention is about date and time, it can be coreferenced with other relative mentions.

  Input: الوقت المتوقع [من سنة إلى سنتين] . بعد [هذه الفترة] سيتخذ القرار .

  Translation: The expected time is [from one to two years]. After [this period] the decision will be taken.

- **Appositives:** These clauses consist of attribute and head. The entire clause is considered a mention.

  Input: [الاقتصادي الشهير فريدمان] توقع بإنهيار الاقتصاد الأمريكي . [هو] يتوقع أيضا بأن العالم سيتبعه .

  Translation: [The famous economist Friedman] anticipates the US economic collapse. [He] also anticipates that the world would follow.

Note that some expressions are not annotated as part of the coreference clusters:

- **Copular verbs:** Also called linking verbs. What these verbs describe are not annotated as part of coreference clusters.

  Input: [ميلتون فريدمان] يعتبر اقتصادي مشهور . [هو] مات في عام 2006 .

  Translation: [Milton Friedman] is a famous economist. [He] died in 2006.

  In the example, the phrase "*famous economist*" is not annotated as part of the coreference chain of *Milton Friedman* since it is preceded by a copular verb.

- **Small Clauses:** constructed clauses and assumptions are not annotated.

  Input: فرنسا تعتبر روسيا الدولة المعتدية وليست أوكرانيا .

  Translation: France considers Russia the aggressor state not Ukraine.

  Russia and the phrase *the aggressor state* are not coreferenced.

- **Verbal Inflections:** Arabic verbs can have different inflectional endings based on the speaker/subject. These endings are not coreferenced.

Input: صرحت رئيسة الجامعة على أهمية الدراسة .

Translation: The university president emphasized on the importance of studying.

The verb صرحت has an affix that points to the speaker, the university president, which is a singular and female. However, this affix is not corefered with *the university president*.

2.4.2 Related Work

We review English and Chinese coreference resolution systems since they are the most investigated languages and they are part of CoNLL-2012 shared task. We then review Arabic coreference.

English: Like with other NLP tasks, most state-of-the-art coreference resolution systems are evaluated on English data. Coreference resolution for English is an active area of research. Until the appearance of neural systems, state-of-the-art systems for English coreference resolution were either rule-based [Lee et al., 2011] or feature-based [Björkelund and Nugues, 2011, Björkelund and Kuhn, 2014, Clark and Manning, 2015, Fernandes et al., 2014, Soon et al., 2001]. Wiseman et al. [2015] introduced a neural network-based approach to solving the task in a non-linear way. In their system, the heuristic features commonly used in linear models are transformed by a tanh function to be used as the mention representations. Clark and Manning [2016b] integrated reinforcement learning to let the model optimize directly on the B^3 scores. Lee et al. [2017] first presented a neural joint approach for mention detection and coreference resolution. Their model does not rely on parse trees; instead, the system learns to detect mentions by exploring the outputs of a bi-directional LSTM. Their system has led to a great progress in coreference resolution and was adopted by many other proposals. Figures 2.5 and 2.6 show their end-to-end system. Lee et al.'s [2018] system is an extended version of Lee et al. [2017] mainly enhanced by using ELMo embeddings [Peters et al., 2018], in addition, the use of second-order inference enabled the system to explore partial entity level features and further improved the system by 0.4 percentage points. Later the model was further improved by Kantor and Globerson [2019] who use BERT embeddings [Devlin et al., 2018] instead of ELMo embeddings. In these systems, both BERT and ELMo embeddings are used in a pre-trained fashion. More recently, [Joshi et al., 2019b] fine-tuned the BERT model for coreference resolution system of Lee et al. [2017], resulting in a small further improvement. Later, Joshi et al. [2019a] introduced SpanBERT which is trained for tasks that involve spans and combined it with Lee et al. [2017]. Using SpanBERT, they achieved a substantial gain of 2.7% when compared with the Joshi et al.'s [2019b] model. Wu et al. [2020] reformulated the coreference resolution task as question answer-

ing task and achieved the state-of-the-art results by pretraining the system first on the large question answering corpora, and then applied it to coreference resolution task.

Summary of reviewed English coreference resolution systems in Table 2.5.

	MUC			B^3			CEAF			
	P	R	F1	P	R	F1	P	R	F1	Avg.
[Martschat and Strube, 2015]	76.7	68.1	72.2	66.1	54.2	59.6	59.5	52.3	55.7	62.5
[Clark and Manning, 2015]	76.1	69.4	72.6	65.6	56.0	60.4	59.4	53.0	56.0	63.0
[Wiseman et al., 2015]	76.2	69.3	72.6	66.2	55.8	60.5	59.4	54.9	57.1	63.4
[Wiseman et al., 2016]	77.5	69.8	73.4	66.8	57.0	61.5	62.1	53.9	57.7	64.2
[Clark and Manning, 2015]	79.2	70.4	74.6	69.9	58.0	63.4	63.5	55.5	59.2	65.7
[Lee et al., 2017]	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
[Lee et al., 2018]	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
[Fei et al., 2019]	85.4	77.9	81.4	77.9	66.4	71.7	70.6	66.3	68.4	73.8
[Kantor and Globerson, 2019]	82.6	84.1	83.4	73.3	76.2	74.7	72.4	71.1	71.8	76.6
[Lee et al., 2018] +BERT	84.9	82.5	83.7	76.7	74.2	75.4	74.6	70.1	72.3	77.1
[Lee et al., 2018] + SpanBERT	85.8	84.8	85.3	78.3	77.9	78.1	76.4	74.2	75.3	79.6
[Wu et al., 2020] + SpanBERT	85.2	87.4	86.3	78.7	76.5	77.6	76.0	75.6	75.8	79.9

Table 2.5: The Table shows some English coreference resolution systems. The used BERT and SpanBERT are in their base version.

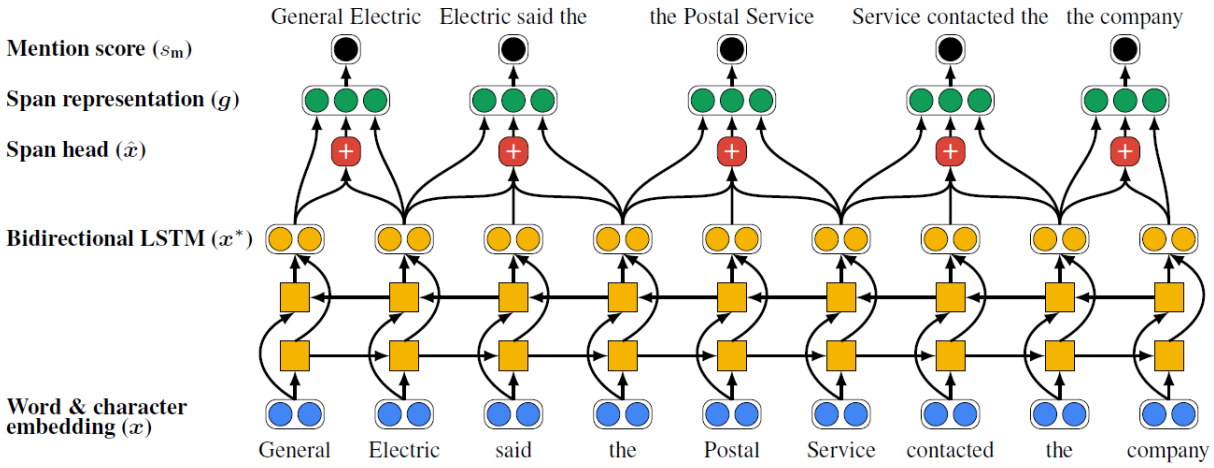

Figure 2.5: Finding the scores of mention spans from [Lee et al., 2017]. First layer finds word and character embeddings and feed them into bi-LSTM network. The network learns the span representations and produces their scores.

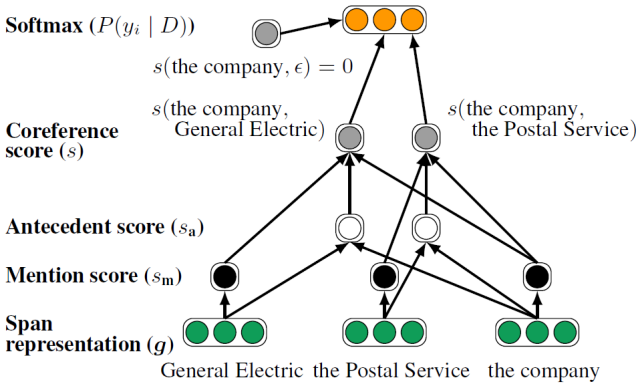

Figure 2.6: Finding the final score by aggregating mention span scores and their pairwise scores from [Lee et al., 2017].

Chinese: In CoNLL-2012 shared task, most participants who evaluated their models on English have also tried Chinese. Ming [2020] used the original BERT and skip-gram representations in Bi-LSTM network to learn clusters. Recently, Jiang and Cohn [2022] combined constituent trees with Lee et al.'s [2018] neural model and a Chinese BERT-base with a whole word masking training strategy [Cui et al., 2021], achieving new state-of-the-art for Chinese. In Table 2.6, we summarize the recent neural Chinese results.

	MUC			B^3			CEAF			
	P	R	F1	P	R	F1	P	R	F1	Avg.
[Ming, 2020]	77.7	63.9	70.13	70.0	54.01	60.9	65.8	51.4	57.5	62.9
[Lee et al., 2018] + BERT-wwm	76.7	70.9	73.7	68.3	62.4	65.2	67.4	60.8	63.9	67.6
[Jiang and Cohn, 2022] + BERT-wwm	84.1	78.6	81.3	77.4	71.5	74.4	76.5	70.0	73.1	76.3

Table 2.6: The Table shows some Chinese coreference resolution systems. BERT-wwm is a pre-trained BERT-base with **w**hole **w**ord **m**asking strategy on Chinese [Cui et al., 2021] .

Arabic: There have been several studies of Arabic coreference resolution task. In particular, several of the systems involved in the CONLL 2012 shared task attempted Arabic as well. li [2012] used a maximal entropy classifier with a set of surface, syntactic, and semantic features to cluster mentions. Zhekova and Kübler [2010] proposed a memory-based approach which finds the k nearest neighbors in the training data. Chen and Ng [2012] employed multiple Sieves [Lee et al., 2011] where each Sieve builds on the previous Sieve, as in Figure 2.7. However, they only used "String Match" Sieve for Arabic since others did not yield any improvements. Stamborg et al. [2012] used a pairwise classifier as well, but they increment basic features by combining them into bigram, trigram, four-gram features. Uryupina et al. [2012] extended BART modular toolkit Versley et al. [2008] to work with Chinese and Arabic. Björkelund and Kuhn [2014] stacked multiple pairwise coreference resolvers and decoders to cluster mentions. Fernandes et al. [2014] proposed a latent tree to capture hidden structure that can lead to finding coreference chains. Their

model is the current state-of-the-art for Arabic evaluated on CoNLL2012. Their system consists of six steps:

1. Mentions detection: detects all nouns, pronouns, named entities, and possessive marks, and consider them possible candidates for any cluster .

2. Basic features: creates pre-defined features of lexical, syntactic, semantic and positional properties.

3. Context feature induction: increments the number of features using Entropy-Guided Feature Induction [Fernandes and Milidiú, 2012] which conjoin the most efficient basic features together.

4. Candidate pair generation: creates a graph **G** by linking all mentions to an artificial node, and then connects mentions that appear earlier to the mentions that appear later. Since the number of mentions can be large, Sieves are used to delete some mentions. Features in steps 2 and 3 decide the edge weights between mentions. An example of the created graph in Figure 2.8 (A).

5. Coreference tree learning: learns feature parameters such that they lead to the maximum arborescence subgraph extracted from **G**. An example in Figure 2.8 (B).

6. Delete the artificial node in **G** and its forward edges, the result is a forest graph with a set of directed spanning trees. Each spanning tree represents a cluster.

A summary of all Arabic coreference resolution systems is shown in Table 2.7.

	MUC			B^3			CEAF			
	P	R	F1	P	R	F1	P	R	F1	Avg.
[li, 2012]	55.60	10.77	18.05	93.34	36.17	52.14	20.95	55.45	30.41	33.53
[Zhekova and Kübler, 2010]	62.13	19.64	29.85	90.72	41.91	57.33	24.81	56.79	34.53	40.57
[Chen and Ng, 2012]	39.96	38.13	39.02	62.51	60.59	61.53	39.84	41.89	41.89	47.13
[Stamborg et al., 2012]	43.49	39.11	41.18	67.95	61.57	64.61	40.36	44.86	42.49	49.43
[Uryupina et al., 2012]	41.66	41.33	41.49	69.23	65.77	67.46	42.13	42.43	42.28	50.41
[Björkelund and Kuhn, 2014]	52.51	43.90	47.82	75.32	62.89	68.54	40.80	48.45	44.30	53.55
[Fernandes et al., 2014]	49.69	43.63	46.46	72.19	62.70	67.11	46.09	52.49	49.08	54.22

Table 2.7: Summary of Arabic coreference resolution systems

All existing proposals suffer from a few limitations. They are built-on English settings, and many important morphological features are not considered. Also, no proposal applied a suitable segmentation tool for Arabic text which helps to segment correctly connected important morphems to the nouns, such as, possessive pronouns. Another important subtask of Arabic coreference resolution that has not been investigated is resolving anaphoric

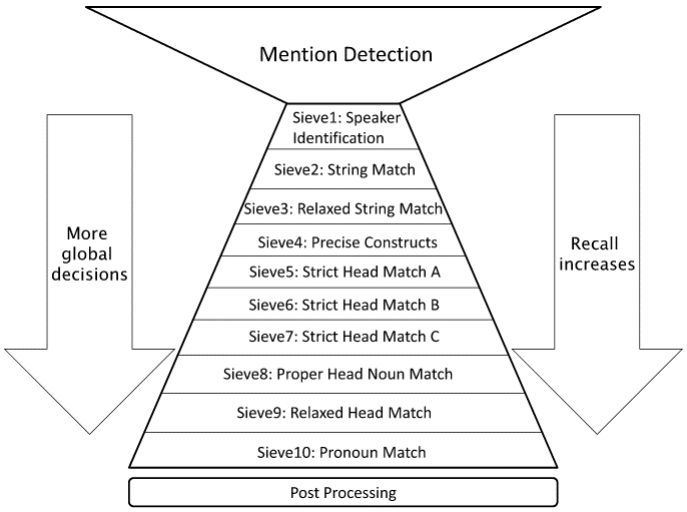

Figure 2.7: Example of coreference resolution sieves [Lee et al., 2013]

zero pronouns. As far as I know, no work has reported anaphoric zero pronouns in their reports.

In early work, coreference's two subtasks were usually carried out in a pipeline fashion [Björkelund and Kuhn, 2014, Clark and Manning, 2016a,b, Fernandes et al., 2014, Soon et al., 2001, Wiseman et al., 2015, 2016], with candidate mentions selected prior the mention clustering step. Since [Lee et al., 2017] introduced an end-to-end neural coreference architecture that achieved state of the art by carrying out the two tasks jointly, as first proposed by [Daume and Marcu, 2005], most state-of-the-art systems have followed this approach. However, no end-to-end solution was attempted for Arabic. We intend to explore whether an end-to-end solution would be practicable with a corpus of more limited size as we will see in Chapter 3.

2.5 Zero Pronouns

Empty categories provide an important source of syntactic information about the phonetically null arguments in pro-drop languages such as Arabic [Eid, 1983], Chinese [Li and Thompson, 1979], Italian [Di Eugenio, 1990], Japanese [Kameyama, 1985], and others [Bever and Sanz, 1997, Kim, 2000]. The use of empty categories started with Penn Treebanks [Marcinkiewicz, 1994], followed by Arabic Treebank [Maamouri et al., 2004], Chinese Treebank [Xue et al., 2005] and other Penn-style series. Empty categories are used to represent traces, such as, movement operations in interrogative sentence, also to represent right node raising which is a shared argument in the rightmost constituent of a coordinate structure. Another usage of empty categories is zero-pronouns (ZP) which are omitted pronouns in places where they are expected to be, and function as overt pronouns. Anaphoric zero pronouns (AZP) are ZPs that corefer to one or more noun phrases

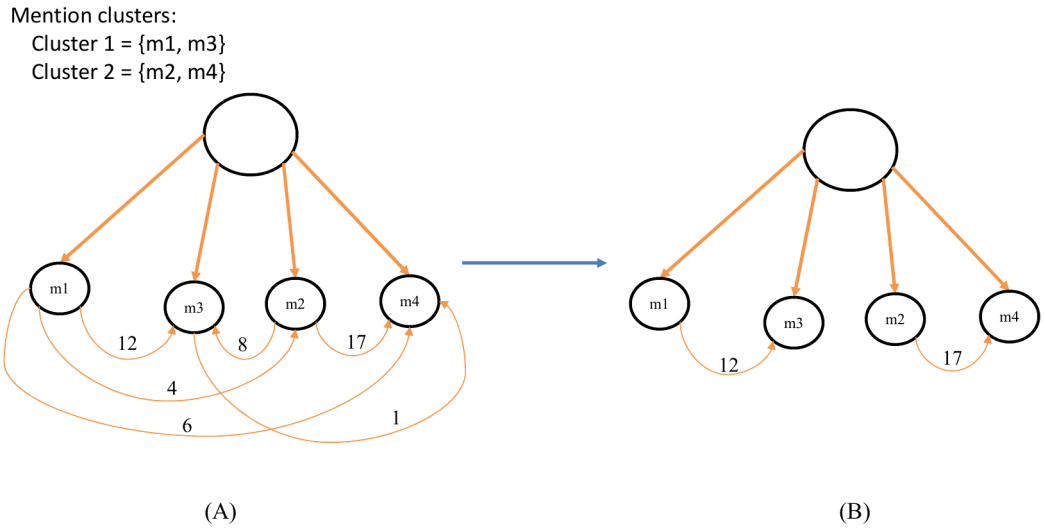

Figure 2.8: An example of Fernandes et al.'s [2014] model, we have four mentions: m1, m2, m3, m4 that appear in a text sequentially, and two clusters : {m1, m3} and {m2, m4}. The mentions are connected with forward edges. The model calculates the edge weights and extracts the optimal arborescence subgraph.

in a preceding text. The following example of an AZP comes from the Arabic section of OntoNotes:

.. المفارقة الأخرى عن بوش هي عدم حماسته للمؤتمر الدولي، اذ انه من البداية، يريد * اجتماعا مختلفا ....

Ironically, Bush did not show any enthusiasm for the international conference, because since the beginning, (he) wanted to attend another conference ...

In the example, the ZP indicated with '*' refers to the gap position of an omitted pronoun (In OntoNotes 5.0, ZPs are denoted as * in Arabic text, and ***pro*** in Chinese). The omitted pronoun refers to a singular masculine person that has been mentioned previously, in the example "Bush/بوش". In Arabic, we deduce the reference information from the context, especially the verb that precedes the AZP, in the example the verb is "wanted/يريد". Since English is not a pro-drop language [White, 1985], the AZP gap position is translated into an overt pronoun (he). The AZP problem has inspired much research because it benefits many natural language processing tasks such as machine translation [Mitkov and Schmidt, 1998], and coreference resolution [Mitkov et al., 2000]. Recently, there has been a great deal of research on AZPs for Chinese [Chang et al., 2017, Kong et al., 2019, Liu et al., 2017, Yin et al., 2017, 2018], Japanese [Shimazu et al., 2020], Korean [Jung and Lee, 2018], and other languages [Gopal and Jha, 2017, Grigorova, 2016]. A major drawback of many existing studies is the assumption that AZP locations are given; hence, they focus primarily on resolving AZPs to their correct antecedent. However, such assumption does not reflect real-life applications. Another drawback is that current AZP resolution systems

rely on language-dependent features and fail to resolve many AZPs. In addition, some languages do not have an AZP system, one of which is Arabic. In the following, we review zero-pronoun related works for Arabic, Chinese and also other languages.

Arabic: There have been a few studies devoted to AZPs and empty categories in general. [Green et al., 2009] proposed a conditional-random-field (CRF) sequence classifier to detect Arabic noun phrases, and captured ZPs implicitly. [Bakr et al., 2009] applied a statistical approach to detect empty categories. [Gabbard, 2010] proposed a pipeline made of maximum entropy classifiers which jointly make a CRF to retrieve Arabic empty categories. As far as we know, no previous work has considered Arabic AZP identification and resolution.

Chinese: Converse [2006] studied AZP resolution and applied a rule-based approach that employed Hobbs algorithm [Hobbs, 1978] to resolve ZPs in the Chinese Treebank; however, did not attempt to automatically identify AZP. The proposal of Yeh and Chen [2006] is another rule-based approach, for AZP resolution and also used a set of hand-engineered rules to identify AZPs. Zhao and Ng [2007] proposed the first machine learning approach to Chinese AZPs identification and resolution, by applying decision trees incorporated with a set of syntactic and positional features. Kong and Zhou [2010] employed a tree kernel-based approach to AZP identification and resolution. Chen and Ng [2013] extended a previous work[Zhao and Ng, 2007] by incorporating contextual features for AZP resolution and applied a combination of syntactic, lexical and other features for the identification. Chen and Ng [2014] proposed unsupervised techniques to resolve AZPs and applied a set of rules to identify AZP. Chen and Ng [2015] implemented an unsupervised approach on the AZP resolution. Chen and Ng [2016] trained a binary classifier to identify AZP and applied a feed-forward neural network to the AZP resolution; Yin et al. [2016] used Chen and Ng's [2016] classifier to identify AZPs. For AZP resolution, they employed an LSTM to represent AZP and two subnetworks (general encoder and local encoder) to capture context-level and word-level information of the candidates; Yin et al. [2017] also applied Chen and Ng's [2016] classifier to detect AZPs and proposed an improved deep memory network to resolve AZPs; and Liu et al. [2017] applied an attention-based neural network to resolve AZPs and enhanced the performance by training on automatically generated large-scale training data. Chang et al. [2017] focused primarily on AZP identification and applied an LSTM neural-network with text and part-of-speech information. Yin et al. [2018] also used an attention-based model, but combined their network with Chen and Ng's [2016] features to resolve AZPs. Yin et al. [2019] applied the same heuristics in Chen and Ng's [2015] system to identify AZPs and applied a collaborative-filtering approach to resolve AZPs. Kong et al. [2019] identified AZPs using a learning-based classifier with semantic, lexical and syntactic features, and used coreferential chain information to improve AZP resolution.

Other languages: There has been also a great deal of research on identification and

resolution of AZPs, particularly in Japanese [Aone and Bennett, 1995, Hangyo et al., 2013, Iida et al., 2006, 2007, 2015, Isozaki and Hirao, 2003, Kim and Ehara., 1995, Sasano and Kurohashi, 2011, Sasano et al., 2008, 2009, Seki et al., 2002, Yamashiro et al., 2018, Yoshikawa et al., 2011, Yoshimoto, 1988, Yoshino et al., 2013], but also in other languages, including Korean [Byron et al., 2006, Han, 2004], Spanish [Ferrández and Peral, 2000, Rello and Ilisei, 2009], Portuguese [Rello et al., 2012], Romanian [Mihăilă et al., 2011], Bulgarian [Grigorova, 2013], and Sanskrit [Gopal and Jha, 2017]. [Iida and Poesio, 2011] proposed the first multilingual approach for AZP resolution.

We address these challenges in Chapters 4 and 5, we also present our systems for identifying and resolving AZPs. We run our proposal on Arabic AZPs and also Chinese. Our method shows great improvements and achieve state-of-the-art results for both languages.

2.6 A summary of the datasets used in this study

There are various anaphora datasets and some target a specific type. For example, pronominal anaphora is considered to be the most widespread type [Mitkov, 1999] and has inspired many researchers to investigate its annotation and resolution for English [Brennan et al., 1987, Kocijan et al., 2019, Lappin and Leass, 1994, Liang and Wu, 2004, Rahman and Ng, 2012, Webster et al., 2018]. This has also inspired many researchers to do the same for Arabic. Hammami et al. [2009] annotated a corpus with coreference chains for Arabic using AnaATAr, a custom-designed XML-tool. Seddik et al. [2015] annotated the Holy Qur'an Annotated with pronominal anaphora information. Sharaf and Atwell [2012] also annotated the Qura'an, but more than previous proposals. They annotated approximately 24,679 pronouns, the largest pronominal anaphora at present. However, previous works only target pronouns and their overall size is relatively small, less than 200,000 words. There are two coreference corpora with larger size and cover more mention types for Arabic. The first is the Automatic Content Extraction (ACE) Doddington et al. [2004] which has ~500,000 tokens, but mentions are restricted to seven semantic types[5] and some can be singletons (mentions that do not corefer). The second is OntoNotes [Weischedel et al., 2011], which covers all entities and does not consider singletons, but the size is smaller than ACE, with ~300,000 tokens. OntoNotes has been the standard for coreference resolution evaluation since the CoNLL-2012 shared task [Pradhan et al., 2012]. However, its Arabic portion is small and includes a single domain (newspapser articles). Chinese and English portions are larger and cover a wide variety of domains (e.g. telephone conversations, newswire, newsgroups, broadcast news, and others). This scarcity of size and domains poses a considerable barrier to improving coreference resolution for Arabic.

[5] The semantic types are person, organization, geo-political entity, location, facility, vehicle, and weapon.

2.6.1 OntoNotes

The OntoNotes project is an annotated corpus covering syntax, predicate argument structure, word senses, named entities and coreference clusters. The goal of OntoNotes project is to ensure high accuracy with an inter-annotator agreement of (~90%). OntoNotes covers three languages English, Chinese, and Arabic [Pradhan et al., 2012]. The corpus statistics in Table 2.9.

Corpora	Language	Words				Documents			
		Total	Train	Dev	Test	Total	Train	Dev	Test
OntoNotes	English	1.6M	1.3M	160K	170K	2,384 (3493)	1,940 (2,802)	222 (343)	222 (348)
	Chinese	950K	750K	110K	90K	1,729 (2,280)	1,391 (1,810)	172 (252)	166 (218)
	Arabic	300K	240K	30K	30K	447 (447)	359 (359)	44 (44)	44 (44)

Figure 2.9: OntoNotes statistics from [Pradhan et al., 2012]

From the Table, we can see English and Chinese have larger corpus size than Arabic. The English data has 1.6 million words while Chinese has 950k, but Arabic has very small size, 300k. This data scarcity poses a challenge for many NLP tasks, and motivates us to see how would a coreference resolution model act under such setting. Another challenge is Arabic anaphoric zero-pronouns which was part of OntoNotes; however, no research proposal studied them. We can have a closer look at the Arabic portion. The data is divided into three splits: train, development, and test. In our experiments, we used each split for its purpose, the train for training the model, the development for optimizing the settings, and the test for evaluating the overall performance. Detailed information about the number of documents, sentences, and words can be found in Table 2.8.

Category	Training	Dev	Test
Documents	359	44	44
Sentences	7,422	950	1,003
Words	264,589	30,942	30,935
AZPs	3,495	474	412

Table 2.8: Detailed statistics of Arabic portion in OntoNotes showing the number of documents, sentences, words, and anaphoric zero pronouns (AZP).

In our experiments, we used OntoNotes for several reasons. First, it is part of CoNLL-2012 shared task which has a standardized evaluation metrics and fixed data splits for train, dev and test. This makes our findings and results can be reproducible and comparable with previous proposals for coreference resolution. Second, OntoNotes has AZPs annotated which we can study for AZP resolution and identification. Third, it covers wide range of

Language	Syntactic category	Train		Development		Test	
		Count	%	Count	%	Count	%
Arabic	Noun Phrase	10.8K	34.93	1.3K	35.02	1.3K	36.51
	Pronoun	8.9K	28.77	1.0K	28.33	1.1K	30.58
	Dropped Pro.	3.5K	11.52	477	12.57	429	11.78
	Proper Noun	4.0K	13.01	450	11.86	390	10.71
	Other Noun	3.3K	10.90	439	11.57	345	9.47
	Verb	25	0.08	4	0.11	0	0.00
	Other	247	0.79	21	0.55	35	0.96

Figure 2.10: Distribution of mentions in CoNLL-2012 dataset the table from [Pradhan et al., 2012]

semantic and anaphora types. We use the dataset and discuses it more with respect to each task, in Chapters 3, 4, 5 and 7.

2.6.2 Wikipedia

Wikipedia articles can be used for various NLP tasks (e.g., text summarization) and also for data augmentation. One of current limitations of Arabic NLP is the scarcity of dataset, especially anaphoric zero-pronouns (AZPs). As far as we know, OntoNotes is the only existing dataset that has labeled AZPs. This is challenging because AZPs are common and often used instead of overt pronouns. In OntoNotes, overt pronouns make up around 29% of mentions while zero-pronouns around 11% as seen in Figure 2.10. The small number of AZP samples can complicate the learning process for AZP systems because they would be exposed to limited number AZP contexts. To overcome this problem, we should have more AZP samples. Manually annotating data is costly and time-consuming for any domain. An alternative way of creating reasonably good and cheap labeled data is using data augmentation techniques and methods. One common way is *synonym thesaurus*, using Wordnet Miller [1995]. Consider the following example:

Input: This is **very** interesting.
Alternative samples using Wordnet:
This is **so** interesting.
This is **pretty** interesting.
This is **really** interest.

In the example, the word ”very” has other synonyms in Wordnet and we can generate more samples by replacing it with its synonyms. It is also possible to replace multiple words, for example, replacing also ”interesting” with ”appealing”. Recently, pre-trained language models have been also used to generate samples. Kumar et al. [2020] have shown different data augmentation methods using BERT and BART. Data augmentation using

Wikipedia has been proven to improve many NLP tasks, such as, machine translation [Xia et al., 2019], question answering [Yang et al., 2019], and others [Raille et al., 2020, Zhu et al., 2022]. Another technique is using round-trip translation [Lau et al., 2015] which is translating a sample from language to another then translating it back to the original language. The technique helps to rephrase the words of the sample and introduces novel words.

Wikipedia is a great source for open text and contains diverse topics. Table 2.9 shows the number of Articles and words of Arabic Wikipedia. Therefore, we intend to apply data augmentation techniques on Arabic Wikipedia articles to generate AZP samples, as we will see in Chapter 6.

Wikipedia:	Total number
Articles	1,028,511
Words	347,825,819

Table 2.9: Wikipedia statistics (as on 08, February, 2020)

Chapter 3

A Neural Coreference Resolver for Arabic

In this Chapter, we will present the first Arabic neural coreference resolver trained for Arabic. Previously, many proposals attempted Arabic as part of CoNLL-2012 shared task. However, the results in terms of CONLL score were much lower than those for English and Chinese. Our coreference resolution system for Arabic is based on Lee et al.'s [2018] end-to-end architecture, combined with the Arabic version of BERT and an external mention detector. As far as we know, this is the first neural coreference resolution system designed specifically for Arabic, and it substantially outperforms the existing state-of-the-art on OntoNotes 5.0 with a gain of 15.2 points CoNLL F1. We also discuss the current limitations of the task and possible approaches that can tackle these challenges. To provide fair comparison with previous proposals, we have not considered resolving anaphoric zero-pronouns (AZPs). In Chapter 7, we will discuss resolving AZP and non-AZP together.

3.1 Motivation

When one looks at the CoNLL-2012 shared task results, it is immediately obvious that the performance on Arabic was lower than English and Chinese in gold and predicted settings for all participants. As shown in Figure 3.1, we can see Arabic results of the top three models, along with Chinese and English. The top three systems did not achieve good results for Arabic. After studying these models, we found these models were cross-lingual and many have used features that were not applicable to Arabic. This inspired us to deep more into the literature to see if there has been any study for Arabic coreference resolution in general especially on OntoNotes (i.e, the largest publicly available coreference resolution corpus for Arabic), but we have not found any. There have been a few studies who investigated

a specific type of anaphora, pronominal anaphora [Beseiso and Al-Alwani, 2016, Bouzid et al., 2017, Sharaf and Atwell, 2012, Trabelsi et al., 2016], but they did not not consider other mentions (e.g. noun phrases, zero-pronouns, verb and other). Understanding the current limitations and challenges of the task is our main motivation.

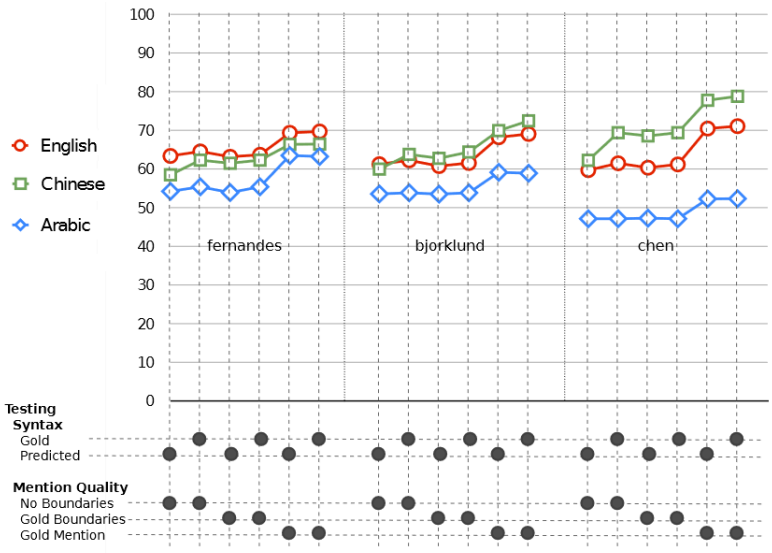

Figure 3.1: Top 3 systems in CoNLL-2012 evaluated on Arabic, Chinese, and English under various settings [Pradhan et al., 2012].

Another motivation is the lack of neural coreference resolver for Arabic language, in fact we are not aware of any learning-based coreference resolver for Arabic since [Björkelund and Kuhn, 2014]. One explanation for this might simply be the lack of training data large enough for the task. Another explanation might be that Arabic is more problematic than English because of its rich morphology, its free-word property, and its high degree of ambiguity. We explore the first of these possibilities and suggest other ways to mitigate the problems of the second. The task is important by itself and to other tasks. Studying the task will help to understand the current limitations and how to overcome them.

3.2 System Architecture

We use the Lee et al.'s [2018] system as our baseline and replace their ELMo embeddings with BERT following the approach of [Kantor and Globerson, 2019]. The input to the system is the concatenated embeddings ($(emb_t)_{t=1}^{T}$) of both word and character levels. The word-level fastText [Bojanowski et al., 2016] and BERT [Devlin et al., 2018] embeddings are used together with the character embeddings learned from a convolution neural network (CNN) during training. The input is then put through a multi-layer bi-directional LSTM to create the token representations ($(x_t)_{t=1}^{T}$). The $(x_t)_{t=1}^{T}$ are used together with

head representations (h_i) to form the mention representations (M_i). The h_i of a mention is calculated as the weighted average of its token representations ($\{x_{b_i}, ..., x_{e_i}\}$), where b_i and e_i are the indices of the start and the end of the mention respectively. The mention score ($s_m(i)$) is then computed by a feedforward neural network to determine the likeness of a candidate to be mention. Formally, the system computes h_i, M_i and $s_m(i)$ as follows:

$$\alpha_t = \text{ffnn}_\alpha(x_t)$$

$$a_{i,t} = \frac{exp(\alpha_t)}{\sum_{k=b_i}^{e_i} exp(\alpha_k)}$$

$$h_i = \sum_{t=b_i}^{e_i} a_{i,t} \cdot x_t$$

$$M_i = [x_{b_i}, x_{e_i}, h_i, \phi(i)]$$

$$s_m(i) = \text{ffnn}_m(M_i)$$

where $\phi(i)$ is the mention width feature embeddings. To make the task computationally tractable, the system only considers mentions up to a maximum width of 30 tokens (i.e. $e_i - b_i < 30$). Further pruning on candidate mentions is applied before approaching the antecedent selection step. The model keeps a small portion (0.4 mention/token) of the top-ranked spans according to their mention scores ($s_m(i)$).

Next, the system uses a bilinear function to compute a light-weight mention pair scores ($s_c(i, j)$) between all the valid mention pairs[1]. The scores are then used to select top candidate antecedents for all candidate mentions (coarse antecedent selection). More precisely, the $s_c(i, j)$ are computed as follows:

$$s_c(i, j) = M_i^\top W_c M_j$$

After that, the system further computes a more accurate mention pair scores between the mentions and their top candidate antecedents $s_a(i, j)$:

$$P_{(i,j)} = [M_i, M_j, M_i \circ M_j, \phi(i, j)]$$

$$s_a(i, j) = \text{ffnn}_a(P_{(i,j)})$$

where $P_{(i,j)}$ is the mention pair representation, M_i, M_j is the representation of the antecedent and anaphor respectively, $\circ$ denotes element-wise product, and $\phi(i, j)$ is the distance feature between a mention pair.

The next step is to compute the final pairwise score ($s(i, j)$). The system adds an artificial antecedent ϵ to deal with cases of non-mentions, discourse-new mentions or cases

[1]Candidate mentions are paired with all the mentions appeared before them (candidate antecedents) in the document.

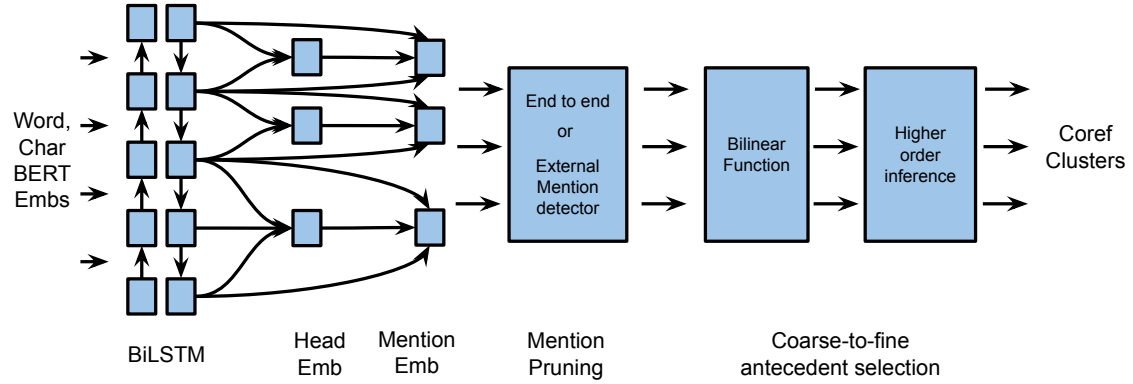

Figure 3.2: The proposed system architecture.

when the antecedent does not appear in the candidate list. The $s(i, j)$ is calculated as follows:

$$s(i,j) = \begin{cases} 0 & i = \epsilon \\ s_m(i) + s_m(j) + s_c(i,j) + s_a(i,j) & i \neq \epsilon \end{cases}$$

For each mention the predicted antecedent is the one that has the highest $s(i, j)$. An anaphora-antecedent link will be created only if the predicted antecedent is not ϵ.

Additionally, the model has an option to use higher-order inference to allow the system to access entity level information. We refer the reader to the original [Lee et al., 2018] paper for more details. We use the default setting of [Lee et al., 2018] to do second-order inference. The final clusters are created using the anaphora-antecedent pairs predicted by the system. Figure 3.2 shows the proposed system architecture of our system.

3.2.1 Multilingual vs. monolingual BERT (AraBERT)

Recently, it has been shown that BERT can capture structural properties of a language, such as its surface, semantic, and syntactic aspects [Jawahar et al., 2019] which seems related to what we need for the coreference resolution. Therefore, we set BERT to produce embeddings for the mentions. BERT is available for English, Chinese, and there is a version for multiple languages, called multilingual BERT [2]. Multilingual BERT is publicly available and covers a wide range of languages including Arabic. Even though the multilingual version provides great results for many languages, it has been shown their monolingual counterparts to achieve better. Therefore, recent research adopts the monolingual approach to pretrain BERT, developing, e.g., CamemBERT for French [Martin et al., 2019], AlBERTo for Italian [Polignano et al., 2019], and others [Kuratov and Arkhipov, 2019, Lee et al., 2020, Souza et al., 2019]. AraBERT [Antoun et al., 2020] is a monolingual BERT model for Arabic which was pre-trained on a collection of Wikipedia

[2] https://github.com/google-research/bert

and newspaper articles. There are two versions, AraBERT 0.1 and AraBERT 1.0, the difference being that the latter pretrained on the word morphemes obtained using Farasa [Darwish and Mubarak, 2016]. The two versions yield relatively similar scores in various NLP tasks. In our experiments, we used AraBERT 0.1 because empirically it proved more compatible with the coreference resolution system.

3.2.2 Mention Detection and Coreference resolution

Mention detection is a crucial part of the coreference resolution system, better candidate mentions usually lead to better overall performance. As suggested by [Yu et al., 2020], a separately trained mention detector can achieve a better mention detection performance when compared to its end-to-end counterpart. In this work, we adapt the state-of-the-art mention detector of [Yu et al., 2020] to aid our system. In their paper, [Yu et al., 2020] evaluated three different architectures for English mention detection task, we use their best settings (Biaffine MD) and replace their ELMo embeddings with BERT embeddings in the same way we did for our coreference system[3]. The Biaffine MD uses contextual word embeddings and a multi-layer bi-directional LSTM to encode the tokens. It then uses a biaffine classifier [Dozat and Manning, 2017] to assign every possible span in the sentence a score. Finally, the candidate mentions are chosen according to their scores. In addition to the standard high-F1 setting, the system has a further option (high-recall) to output top mentions in the proportion of the number of tokens, this is similar to our mention detection part of the system. Here we use the high-recall settings of the mention detector we modify the baseline system to allow the system using the mentions supplied by the external mention detector.

To confirm our hypothesis that a separately trained mention detector can achieve a better mention detection performance, we compare the mention detection performance of our system with the separately trained mention detector. For our system, we train the models end-to-end and assess the quality of candidate mentions before feeding them into the mention clustering part of the system. Table 3.1 shows the comparison of both systems in three different settings (MultiBERT (baseline), MultiBERT + Pre (multilingual BERT and data pre-processing), AraBERT + Pre (AraBERT and data pre-processing)). As we can see from the table, the separately trained mention detector constantly have a better recall of up to 3% when compared with the jointly trained mention detector[4].

The preliminary experiments show that by simply using the mentions generated by the external mention detector in a pipeline setting result in a lower coreference resolution performance. We believe this is mainly because in an end-to-end setting, the model is exposed to different negative mention examples; hence, has a better ability to handle false

[3]We tried to add the fastText and character-based embeddings to the system but found they do not improve the mention detection results

[4]Here we only care about the recall as the number of candidate mentions is fixed

Algorithm 1: End-to-end annealing algorithm.

Input: Training step: N; Candidate mentions from external mention detector: $\text{Candidate}_{\text{external}}$
Output: Trainable variables: W

$n = 0$;
while $n \leq N$ **do**
  $\text{pipeline}_{\text{ratio}} \leftarrow n/N$;
  $rand = random.random()$;
  if $rand \leq \text{pipeline}_{\text{ratio}}$ **then**
    CandidateMention $\leftarrow \text{Candidate}_{\text{external}}$;
  else
    Generate mention candidates $\text{Candidate}_{\text{end-to-end}}$;
    CandidateMention $\leftarrow \text{Candidate}_{\text{end-to-end}}$;
  end
  Predict antecedent for candidate mentions;
  Compute training loss;
  Update W;
  $n \leftarrow n + 1$
end

Models	Joint			Separate		
	R	P	F1	R	P	F1
Baseline (MultiBERT)	85.6	24.4	38.0	**88.1**	25.2	39.2
MultiBERT + Pre	91.2	26.0	40.5	**93.3**	26.6	41.5
AraBERT + Pre	92.5	26.4	41.1	**95.5**	27.2	42.4

Table 3.1: The mention detection performance comparison between the separately and jointly trained mention detectors in a high recall setting.

positive candidates. To leverage the benefits between better candidate mentions and more negative mention examples, we introduce a new hybrid training strategy (end-to-end annealing) that initially training the system in an end-to-end fashion and linearly decreasing the usage of end-to-end approach. At the end of the training, the system is trained purely in a pipeline fashion. The resulted system is then tested in a pipeline fashion. Algorithm 1 shows the details of our end-to-end annealing training strategy.

3.3 Experimental Setup

Since the BERT models are large, the fine-tuning approaches are more computationally expensive: GPU/TPUs with large memory (32GB+) are required. In this work, we use BERT embeddings in a pre-trained fashion to make our experiment feasible on a GTX-1080Ti GPU with 11GB memory. The running time was approximately 62 hours.

Parameter	Value
bi-directional LSTM layers/size/dropout	3/200/0.4
FFNN layers/size/dropout	2/150/0.2
CNN filter widths/size	[3,4,5]/50
Char/fastText/Feature embedding size	8/300/20
BERT embedding size/layer	768/Last 4
Embedding dropout	0.5
Max span width	30
Max num of antecedents	50
Mention/token ratio	0.4
Optimiser	Adam (1e-3)
Training step	400K

Table 3.2: Hyperparameters for our models.

3.4 Hyperparameters

We use the default settings of [Lee et al., 2018], and replace their GloVe/ELMo embeddings with the fastText/BERT embeddings. Table 3.2 shows the hyperparameters used in our system.

3.5 Results

Baseline We first evaluate our baseline system using the un-pre-processed data and the multilingual BERT model. As we can see from Table 3.3, the baseline system already outperforms the previous state-of-the-art system which is based on handcrafted features by a large margin of 2.6 percentage points. The better F1 scores are mainly as a result of a much better precision in all three metrics evaluated, the recall is lower than the previous state-of-the-art system [Björkelund and Kuhn, 2014].

Data pre-processing We then apply heuristic rules to pre-process the data. The goal of pre-processing is to reduce the sparsity of the data by normalizing the letters that have different forms and removing the diacritics. By doing so, we created a 'clean' version of the data. As we can see from Table 3.5, the simple pre-processing on the data achieved a large gain of 7 percentage points when compared with the baseline model trained on the original data. Since the pre-processing largely reduced the data sparsity, the recall of all three matrices has been largely improved. We further compare the mention scores of two models (see Table 3.4). As illustrated in the table, the system trained on the pre-processed data achieved a much better recall and a similar precision when compared with the baseline. This suggests that data pre-processing is an efficient and effective way to improve the performance of the Arabic coreference resolution task.

Language Specific BERT Embeddings Next, we evaluate the effect of the language-

Models	MUC			B^3			$CEAF_{\phi_4}$			Avg.
	R	P	F1	R	P	F1	R	P	F1	F1
[Björkelund and Nugues, 2011]	43.9	52.5	47.8	35.7	49.8	41.6	40.5	41.9	41.2	43.5
[Fernandes et al., 2012]	43.6	49.7	46.5	38.4	47.7	42.5	48.2	45.0	46.5	45.2
[Björkelund and Kuhn, 2014]	47.5	53.3	50.3	44.1	49.3	46.6	49.2	49.5	49.3	48.7
baseline (multiBert)	45.7	66.9	54.3	38.8	64.3	48.4	45.7	57.9	51.1	51.3
multiBert+pre	56.1	67.1	61.1	50.0	63.4	56.0	54.8	61.1	57.8	58.3
araBert+pre	62.3	70.8	66.3	56.3	65.8	60.7	58.8	**66.1**	62.2	63.1
araBert+pre+md	**63.2**	**70.9**	**66.8**	**57.1**	**66.3**	**61.3**	**61.6**	65.5	**63.5**	**63.9**

Table 3.3: Coreference resolution results on Arabic test set.

Models	R	P	F1
Baseline (MultiBERT)	56.5	79.1	65.9
MultiBERT + Pre	67.4	78.8	72.6
AraBERT+Pre	70.6	79.9	75.0
AraBERT+Pre+MD	**72.9**	**80.4**	**76.4**

Table 3.4: Mention detection results on Arabic test set.

specific BERT embeddings. The monolingual BERT model (AraBERT) trained specifically on Arabic Wikipedia and several news corpora has been shown that it can outperform the multilingual BERT model on several NLPtasks for Arabic. Here we replace the multilingual BERT model with the AraBERT model to generate the pre-trained word embeddings. We test our system with AraBERT on the pre-processed text, the results are shown in Table 3.3 and Table 3.4. As we can see from the Tables, the model enhanced by the AraBERT achieved large gains of 4.7 and 2.4 percentage points when compared to the model using multilingual BERT on coreference resolution and mention detection respectively. Both recall and precision are improved for all the metrics evaluated which confirmed the finding in [Antoun et al., 2020] that AraBERT model is better suited for Arabic NLP tasks.

External Mention Detector Finally, we use a separately trained mention detector to guide our models with a better candidate mentions. We train a mention detector using the same CoNLL 2012 Arabic datasets and store the top-ranked mentions in the file. We use the top-ranked mentions from the external mention detector in a pipeline fashion, the mentions are fixed during the training of the coreference resolution task. We use the output of the mention detector model trained on the pre-processed data and using the AraBERT embeddings as this model performs best over three settings we tested, as shown in Table 3.1. In the Table, we had a high recall of correct spans which is our goal in the mention detection[5]. We use the end-to-end annealing training strategy proposed in Section

[5]In mention detection, we had low precision because the model did not detect many FP (gold non-span mentions).

3.2.2 to train our model with both end-to-end and pipeline approaches. The model is then tested in a pipeline fashion. Table 3.3 shows our results on coreference resolution, the model enhanced by the external mention detector achieved a gain of 0.8% when compared to the pure end-to-end model. We further compared the mention detection performance between two models in Table 3.4, as expected the new model has a much better mention recall (2.3%) when compared to the pure end-to-end model (araBert+pre), this suggests our training strategy successfully transferred the higher recall achieved by the external mention detector to our coreference system.

Overall, our best model enhanced by the data pre-processing, monolingual Arabic BERT and the external mention detector achieved a CoNLL F1 score of 63.9% and this is 15.2 percentage points better than the previous state-of-the-art system [Björkelund and Kuhn, 2014] on Arabic coreference resolution.

3.6 Discussion

Coreference resolution is a difficult task, and even more for languages such as Arabic with more limited resourced. The main challenge is the lack of large scale coreference resolution corpora. At present there are two multilingual coreference corpora that cover Arabic. The first is the Automatic Content Extraction (ACE) [Doddington et al., 2004] which has ~500,000 tokens, but mentions are restricted to seven semantic types[6] and some can be singletons (mentions that do not corefer). The second is OntoNotes [Pradhan et al., 2012], which covers all entities and does not consider singletons, but the size is smaller than ACE, with 300,000 tokens. A summary of the two corpora is in Table 3.5. OntoNotes has been the standard for coreference resolution evaluation since the CoNLL-2012 shared task. However, its Arabic portion is small and this scarcity poses a considerable barrier to improving coreference resolution.

Another challenge of the task is the absence of large pre-trained language models. There are two versions of BERT: BERT-base and BERT-large. BERT-large integrates more parameters to encode better representations for mentions which usually leads to a better performance in many NLP tasks. AraBERT and multilingual BERT are pre-trained using the BERT-base approach because BERT-large is computationally expensive. We are not aware of any publicly available BERT-large for Arabic that we could have used in our experiments. We surmise that a BERT-large version of Arabic can improve the overall performance as shown in prior works [Joshi et al., 2019b, Kantor and Globerson, 2019].

[6] The semantic types are person, organization, geo-political entity, location, facility, vehicle, and weapon.

Corpora	Language	Tokens	Documents
ACE	English	~960,000	-
	Chinese	~615,000	-
	Arabic	~500,000	-
OntoNotes	English	~1,600,000	2384
	Chinese	~950,000	1729
	Arabic	~300,000	447

Table 3.5: General domain coreference resolution corpora that include Arabic.

3.7 Summary

In this Chapter we make the following contributions. First, we propose a neural coreference resolver for Arabic that achieves substantial improvements over the state of the art. We achieved an F1 score of 63.9%, which is 15.2% higher than the previous state-of-the-art. Second, we show the current challenges of the task. The existing datasets are small and does not cover several mention types, and also the lack of large pre-trained language models.

We found that a neural coreference resolver with a sophisticated language model works reasonably well on the Arabic portion of CoNLL-2012. The portion size is small compared to its English and Chinese counterparts, yet the end-to-end resolver managed to resolve many mentions correctly. This suggests the model is capable of learning under such setting, but not enough to learn about different coreference types. We surmise that annotating a large scale dataset with various types of anaphora can be beneficial to the field.

Chapter 4

Zero Pronoun Identification

In languages like Arabic, Chinese, Italian, Japanese, Korean, Portuguese, Spanish, and many others, predicate arguments in certain syntactic positions are not realized and are thus called zero- or null-pronouns. Identifying and resolving such omitted arguments is crucial to machine translation, information extraction and other NLP tasks, but depends heavily on semantic coherence and lexical relationships. In this Chapter, we propose a BERT-based cross-lingual model for zero pronoun identification (AZP resolution will be in the next Chapter), and evaluate it on the Arabic and Chinese portions of OntoNotes 5.0. Typically, AZP identification consists of two steps. The first is the *extraction* step where potential ZP locations are extracted. The extraction procedure is based on heuristics and depends on the target language structure. The second step is *classification*; this determines which of the extracted candidate are AZP, excluding other, non-anaphoric ZP (e.g., subject-movement). The classification step is more challenging because of the varieties and size of the extracted candidates. We propose a multilingual approach to AZP identification based on BERT. We evaluate on languages that differ completely in their morphological structure: Arabic and Chinese (Arabic is morphologically rich, whereas Chinese's morphology is relatively simple [Pradhan et al., 2012]). Ours is the first neural network-based AZP identification model for Arabic, and it substantially surpasses the current state-of-the-art results on Chinese.

4.1 Motivation

Anaphoric zero-pronouns (AZP) identification is the task of detecting the locations of the omitted arguments and these detected are actually anaphoric. Much NLP work on AZP is based on gold mentions, but models for their identification are a fundamental prereq-

uisite for their resolution in real-life applications. Such identification requires complex language understanding and knowledge of real-world entities. The task is crucial and precedes another task which is AZP resolution. While there has been various research studies dedicated to Chinese, Japanese, Spanish and other, there has been none for Arabic, which is our main motivation. Another motivation is the current limitation of existing proposals for other languages. They rely on language-dependent and complex features to detect many AZP locations. These two reasons encouraged us to investigate the task of AZP identification. We propose a general multi-lingual model that is applicable not only to Arabic, but also to other languages. To confirm the efficiency of our method, we evaluate on Chinese and our results achieve the state-of-the-art.

4.2 Proposed Model

To identify AZPs, context understanding and semantic knowledge of entities are essential in Chinese [Huang, 1984] as well as in Arabic which requires, in addition, deep understanding of its complex morphology [Alnajadat, 2017]. Recently, it has been shown that BERT [Joshi et al., 2019b] can capture structural properties of a language, such as its surface, semantic, and syntactic aspects [Jawahar et al., 2019] which seems suitable for the AZP identification task. Therefore, we use BERT to produce representations for ZP candidates. Our model is a binary classifier that takes an automatically predicted ZP candidate as input, and classifies it as an AZP or not. In this section, we describe how we represent AZP candidates, and how we create its candidates. Finally, we present the training objective, hyperparameter tuning settings and the results. In our experiments, we used multilingual BERT-base [Devlin et al., 2018] to represent words. Multilingual BERT was pre-trained for many languages, including Arabic and Chinese.

4.2.1 Input Representation

We represent AZPs by their surrounding context, specifically, we represent each candidate by its VP headword and its context window of two words (left and right). Consider a sentence with a gap candidate *C* at position *i*, so its surrounding context at positions *i-2*, *i-1*, *i+1*, *i+2*.

$$sentence = (w_1, w_2, ...w_{i-2}, w_{i-1}, C_i, w_{i+1}, w_{i+2}, .., w_n) \tag{4.1}$$

We feed *sentence* into BERT feature extraction mode as input and it outputs *embeddings* of every word of *sentence*.

$$embeddings = BERT(sentence) \tag{4.2}$$

We extract the embeddings of the candidate position and its surrounding context. In our experiments, BERT Tokenizer, Wordpiece [Wu et al., 2016], segmented many Arabic words into multiple sub-tokens, each with its own embeddings. For example, the word *sleeping* might be segmented into two sub-tokens *sleep* and *##ing*. One way to represent word sub-tokens is to compute their mean; therefore, we create the function μ which computes the mean of sub-token embeddings. We join the AZP context representations together into a value called *azp*.

$$a_1 = \mu(embeddings_{(i-2)}) \tag{4.3}$$
$$a_2 = \mu(embeddings_{(i-1)}) \tag{4.4}$$
$$a_3 = \mu(embeddings_{(i)}) \tag{4.5}$$
$$a_4 = \mu(embeddings_{(i+1)}) \tag{4.6}$$
$$a_5 = \mu(embeddings_{(i+2)}) \tag{4.7}$$
$$azp = [a_1, a_2, a_3, a_4, a_5] \tag{4.8}$$

azp encodes information about the candidate context and serves as input to our classifier. It is possible to extend the AZP window to more context but we empirically find context window of size 2 to be sufficiently effective.

$$layer_1 = f(W_1 \boldsymbol{azp} + b_1) \tag{4.9}$$
$$layer_2 = f(W_2\, layer_1 + b_2) \tag{4.10}$$
$$output = f(W_3\, layer_2 + b_3) \tag{4.11}$$

The binary classifier is a multi-layered perceptrons consisting of two layers and one output layer. f is the ReLU activation function [Nair and Hinton, 2010]. Each layer in the classifier has learning parameters W and b. The input is then classified to be either an AZP or not.

4.2.2 Candidate Generation

Although ZPs are annotated in OntoNotes, our model works off automatically predicted candidates. ZP locations differ in Chinese and Arabic. In Chinese, ZPs appear before a VP node [1] while in Arabic they appear after the head of a VP node [2]. An example of Chinese and Arabic ZP locations in Figure 4.1, Chinese AZP is denoted with *pro* while Arabic is with *. We extract Chinese ZP locations as in Zhao and Ng's [2007] work.

[1] We follow the annotation guidelines as in the Penn Chinese TreeBank [Xue et al., 2005].

[2] There are two types of word order for Arabic: Subject-Verb-Object and Verb-Subject-Object. Both are used and acceptable. In the annotation process, Arabic Treebank sets the Verb-Subject-Object as the official order [Weischedel et al., 2011].

They consider every gap before a VP node as a candidate. The number of candidates can be large. [Kong and Zhou, 2010] showed that if a VP node is in a coordinate structure or modified by an adverbial node, only its parent VP node is considered, thus decreasing the number of necessary candidates. For Arabic, we consider every gap after every head of a VP node as a candidate. A candidate is positive if it is an AZP, negative otherwise. Both approaches result in extracting many negatives examples and a small number of positive examples. The high imbalance between the two classes can make a model biased; we address the problem in Section 4.3.

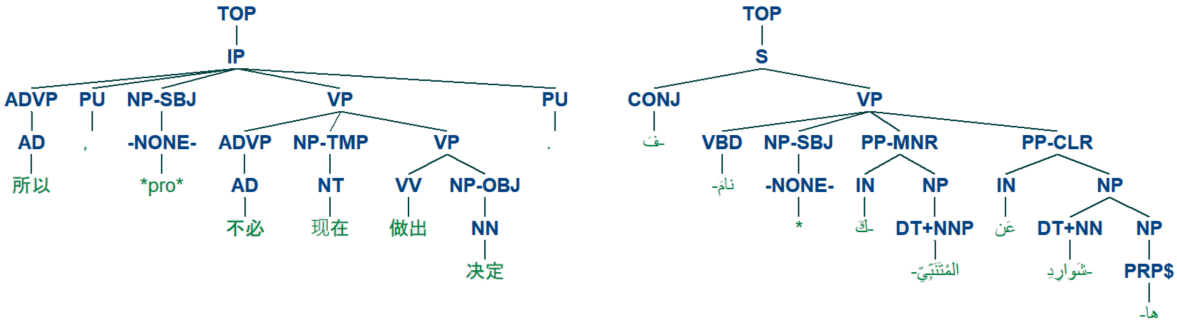

Figure 4.1: Chinese ZPs appear before a VP node (left), and Arabic ZPs appear after the verb of a VP head (right).In OntoNotes 5.0, Chinese AZPs are annotated as ***pro*** and Arabic AZPs as *.

4.2.3 Training Objective

The training objective of our classifier is binary cross-entropy loss:

$$J(\theta) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)] \tag{4.12}$$

θ represents the set of learning parameters in the model. N is the number of training data. y_i is the true label of training i and $\hat{y}_i$ its predicted label.

4.2.4 Hyperparameter Tuning

In the classifier, we employ two layers and initialize each one's weights using Glorot and Bengio's [2010] method. We also add a dropout regularization between the two layers and the output layer. We tune the hyperparameters based on the development sets. Table 5.1 shows the hyperparameter settings. We used three GPUs, each is 8GB. The training time is approximately 12 hours.

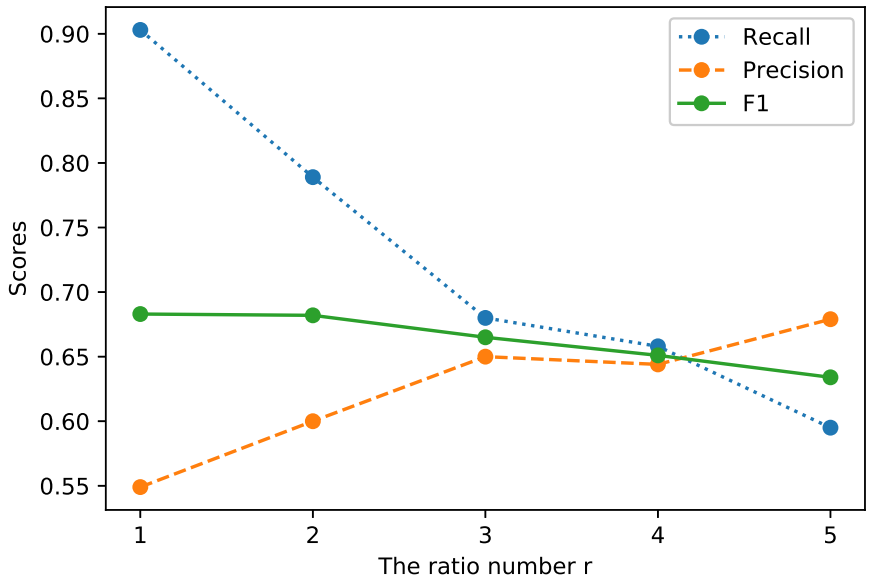

Figure 4.2: The effect of tuning the ratio r on recall, precision and F1 scores on the Arabic test set.

4.3 Results

AZP identification results for Arabic are in Table 4.1, and Chinese in Table 4.2. The training data is highly imbalanced because of the ratio of negatives examples to the positive examples. In Arabic there are 5.6 times of negative examples compared to the positive examples, and in Chinese the negative examples are 16.2 times compared to the positive ones. To address this problem, we follow Zhao and Ng's [2007] approach by changing the ratio weight r of sampling positive examples with respect to negative examples. The value r affects precision and recall scores. If r is high, precision increases but recall decreases. The effect of tuning r on precision, recall and F1 scores on Arabic and Chinese are in Figures 4.2 and 4.3 respectively. F1 scores with different variations of r are not very significant; however, we choose r that balances between the precision and recall scores.

Prior works [Chang et al., 2017, Chen and Ng, 2013, 2014, 2016, Kong et al., 2019] evaluate AZP identification under two settings: gold and system parse because annotation quality can impact the number of recovering candidates in the *extraction* step. Gold annotations are available for both languages and we also automatically parse the data with syntactic trees using the Berkeley Parser [Kitaev et al., 2018] which is a pre-trained parser using neural networks and self-attention.

Arabic

As far as we know, there has been no published proposal on Arabic AZP identification. Therefore, we implemented as a baseline Chang et al.'s [2017] model, which employs

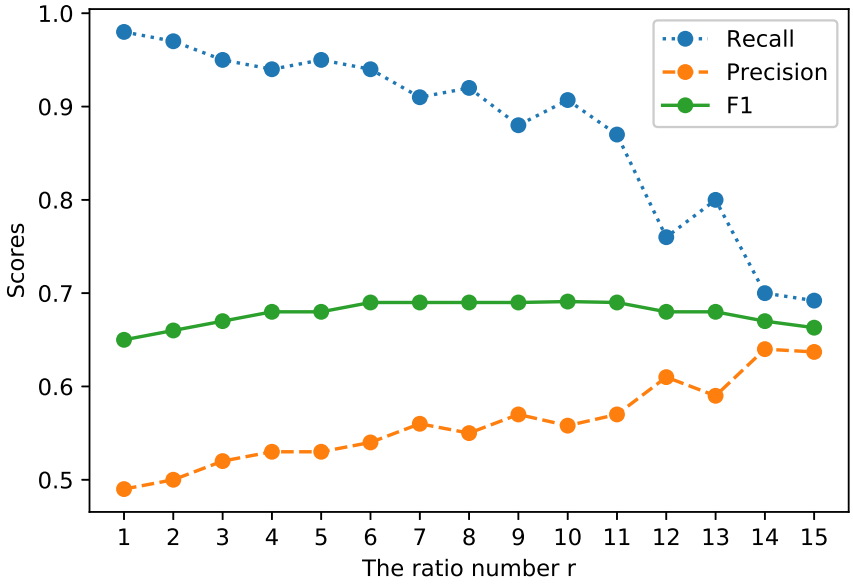

Figure 4.3: The effect of tuning the ratio *r* on recall, precision and F1 scores on the Chinese test set.

sentence and Part-of-Speech information into a Bi-LSTM neural network to identify ZPs. We set its embedding layer to the Arabic version of Fasttext [Bojanowski et al., 2016]. We can see in Table 4.1 that our approach outperforms the baseline in both gold and system settings with F1 scores of 68.2% and 47.0%. There is a a big gap between gold and system parse because the automatic parser failed to recognize many VP nodes in the *extraction* step. Thus, many AZP samples were not recognized for training and evaluation which lead to a great decrease in performance. To gain additional insights into our model, we analyzed its outputs. The model correctly identifies many AZP cases, however, it struggles to recognize some patterns especially AZPs that are preceded by a verb in the first person. The errors can be attributed to the distribution of the training data. Most training AZP data are headed by verbs in the third person, and the number of verbs in the first and second persons is very small; thus, the model did not learn to classify many of these cases. A corpus that include a larger distribution of such cases can help a model to learn them.

	Settings 1: Gold Parse			Settings 2: System Parse		
	R	P	F1	R	P	F1
Baseline	**67.7**	45.2	54.2	31.7	30.6	31.1
Our model (r=2)	60.0	**78.9**	**68.2**	**38.6**	**60.1**	**47.0**

Table 4.1: AZP identification results for Arabic. The highest score is in **bold**.

Chinese

We compare our approach with other proposals in Table 4.2. As we can see, our approach achieves the highest F1 scores of 69.1% and 68.7% with gold and system parse settings, outperforming all prior proposals. The F1-score difference between our approach and the state-of-the-art approach is 4.7% with gold parse settings and 11.3% with system parse. The F1-score difference of gold and system settings of our approach is relatively small (0.4%) because the Berkeley parser annotated many VP nodes correctly. We analyzed the errors, and noticed many unidentified AZPs are located at the beginning of their samples. These cases depend on previous sentences, and their information might have not been encoded in the AZP input; thus, our model failed to identify them.

	Settings 1: Gold Parse			Settings 2: System Parse		
	R	P	F1	R	P	F1
[Chen and Ng, 2013]	50.6	55.1	52.8	30.8	34.4	32.5
[Chen and Ng, 2014]	72.4	42.3	53.4	42.3	26.8	32.8
[Chen and Ng, 2016]	75.1	50.1	60.1	43.7	30.7	36.1
[Chang et al., 2017]	63.5	**65.3**	64.4	57.2	55.7	56.4
[Kong et al., 2019]	70.1	59.4	64.3	60.2	40.2	48.2
Our model (r=10)	**90.7**	55.8	**69.1**	**81.9**	**59.2**	**68.7**

Table 4.2: AZP identification results for Chinese. The highest score is in **bold**.

4.4 Discussion

BERT representations work interestingly well on AZPs even though empty categories have not been considered during the BERT's pretraining. Recent works [Clark et al., 2019, Goldberg, 2019, Jawahar et al., 2019, Kovaleva et al., 2019] have shown that BERT learns various linguistic information such as, syntactic roles, coreference resolution, semantic relations and others. Our experimental results suggest that these information might be encoded in AZP contexts which make them distinctive.

Current approaches for AZP identification evaluate under two settings: gold and system annotations because the task depends highly on the annotation quality of parse trees. In our experiments, gold settings for both Arabic and Chinese achieve outstanding results. In system parse, Chinese achieves results similar to its gold setting; however, Arabic does not. The reason is that Berkeley Parser [Kitaev et al., 2018] fails to parse correctly Arabic sentences which means many correct AZP locations are not detected in the extraction step. A sophisticated Arabic parser can improve the overall performance for system-parse settings.

4.5 Summary

In this Chapter, we proposed a multi-lingual approach for detecting anaphoric zero-pronouns (AZPs) and we evaluated on Arabic and Chinese. The task is essential to AZP resolution which we will discuss in the next Chapter. Our contributions are the following. First, the proposed AZP detector is the first model applied to Arabic and it can be extended to other languages, the model achieved state-of-the-art results for Chinese. Second, we experimentally found that BERT encodes AZPs within their contexts even though they have not been explicitly considered during pre-training.

There are a few limitations to our method possibly the main one is that the performance relies on part-of-speech parser (e.g. an essential step to find potential candidates). A possible future direction is to work on an end-to-end model that learn how to detect AZP gaps without using any other tools or external knowledge.

Chapter 5

Zero Pronoun Resolution

In the previous Chapter, we looked at AZP identification. In this Chapter, we will look at AZP resolution. Both AZP tasks are related to coreference resolution, as we will see in Chapter 7. AZP resolution system finds the correct antecedent of an AZP of all the previously mentioned candidates. We propose combining BERT representations with additional task-related features to improve AZP resolution. Our model applies BERT feature extraction mode to produce embeddings for AZPs and their antecedents, and add two features: *same_sentence* and *find_distance*. *same_sentence* feature finds whether an AZP and a candidate appear in the same sentence or not, and *find_distance* computes the word distance between an AZP and a candidate. These two features are cross-lingual and highly related to the task because AZPs and their antecedents usually appear near each other [Chen and Ng, 2014]. Our method is applicable to other languages; so we have also evaluated on Chinese and achieved state-of-the-art results.

5.1 Motivation

Anaphoric zero-pronoun (AZP) resolution is the task of resolving AZPs to their true antecedents. Zero-pronoun anaphora is part of coreference resolution and important to other NLP tasks, such as, machine translation and information extraction [Mitkov and Schmidt, 1998, Mitkov et al., 2000]. One of our motivation is to understand the challenges and problems of resolving Arabic AZPs. Another motivation is to build a framework that can resolve AZP and non-AZP together. The models in this and previous Chapters are essential components of our two methods in Chapter 7 where we discuss resolving AZPs and non-AZPs. These two reasons inspired us to investigate resolving AZPs. Our proposed method can be generalized and applied to other languages, not only for Arabic. Therefore,

we evaluate on the Chinese portion of OntoNotes and achieved state-of-the-art results.

5.2 Proposed Model

In our experiments, we used multilingual BERT-base [Devlin et al., 2018] to represent words. Multilingual BERT was pre-trained for many languages, including Arabic and Chinese. Consider a sentence consisting of *n* words and containing an AZP mention at position *i*, so that its previous word is at position *i-1*, and the next word at *i+1*. Let us assume we also have a candidate$_k$ starting at position *k*, and appearing before the AZP.[1] There can be a number of candidates, each of which is a noun phrase.

$$sentence = (w_1, w_2, ..., w_{i-1}, azp_i, w_{i+1}, ..., w_n) \tag{1}$$

$$candidate_k \subset sentence \tag{2}$$

We compute the positional features for every (azp, candidate) pair as follows:
- *same_sentence (azp, candidate)*: returns 1 if an AZP and its candidate are in the same sentence, 0 otherwise.
- *find_distance (azp, candidate)*: finds the word distance between an AZP and its candidate.

$$s = same\_sentence(azp_i,\ candidate_k) \tag{3}$$

$$d = find\_distance(azp_i,\ candidate_k) \tag{4}$$

We feed *sentence* into BERT feature extraction mode, which produces the input's *embeddings*. *embeddings* contain BERT pretrained vectors of every word in *sentence*.

$$embeddings = BERT(sentence) \tag{5}$$

A word can have one representation or several based on the segment step of Tokenizer. For example, in Figure 5.1 *My* has only one embedding while *sweetheart* has two because it has been segmented into two sub-tokens (sweet and ##heart). In 6, 7, and 5.8 equations, the subscript of *embeddings* represents the word location in the sentence. μ is a function to compute the mean of a mention representation which can made of several subtoken

[1] An AZP and its candidate may appear in distinct sentences. This could be specified using BERT's parameters 'text_a', and 'text_b'. In such cases, however, we empirically found that we get better results by merging the two sentences into one, and add a [SEP] token in between. Thus, we only use 'text_a'.

embeddings [2].

$$a_1 = \mu(embeddings_{(i-1)}) \tag{6}$$

$$a_2 = \mu(embeddings_{(i+1)}) \tag{7}$$

$$c_k = \mu(embeddings_{(k)}) \tag{8}$$

To obtain a mention representation for an AZP, we compute the weighted sum of the AZP previous word and the next word, and join them together. For every candidate, we calculate the mean of its embeddings which then joined with the positional features. We combine the AZP and its candidate representations to form the input to our classifier.

$$azp = [a_1,\ a_2] \tag{9}$$

$$c = [c_k,\ s,\ d] \tag{10}$$

$$input = [azp,\ c] \tag{11}$$

Our classifier consists of multiple multi-layer perceptrons (MLPs) scoring the <azp, candidate> pair "*input*".

$$layer_1 = f(W_1 \boldsymbol{input} + b_1) \tag{12}$$

$$layer_2 = f(W_2\ layer_1 + b_2) \tag{13}$$

$$layer_3 = f(W_3\ layer_2 + b_3) \tag{14}$$

$$scoring = f(W_4\ layer_3 + b_4) \tag{15}$$

f is the RELU activation function [Nair and Hinton, 2010]. layer$_1$, layer$_2$, layer$_3$, and *scoring* are the resolver's layers; each has learning parameters W and b. The overall architecture of our model and data representations are shown in Figure 5.2. In the figure, there is one AZP and two candidates: noun phrase 1 (NP1) at position x and noun phrase 2 (NP2) at position y. We run the sentence into BERT to get their word embeddings. AZP is represented with the mean of its previous word, and next word. Candidates are also represented with the mean of their subtoken embeddings, and combined with their positional features. We join each candidate representation with the AZP. We compute <AZP, NP1> and <AZP, NP2> scores which normalized using the softmax layer.

5.3 Input representation

Consider a sentence consisting of *n* words and containing an AZP mention at position *i*, so that its previous word is at position *i-1*, and the next word at *i+1*. Let us assume we

[2] In our experiments, Tokenizer segmented many Arabic text into several sub-tokens, but rarely did segment Chinese.

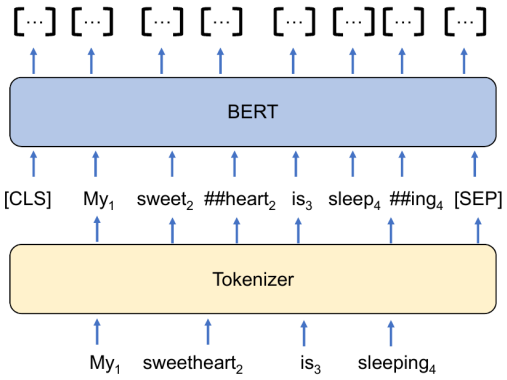

Figure 5.1: The sentence "My sweetheart is sleeping" preprocessed through Tokenizer. Tokenizer segments words, and introduces '[CLS]' and '[SEP]' tokens. After the Tokenizer step, the input is fed into BERT, which outputs embeddings.

also have a candidate$_k$ starting at position *k*, and appearing before the AZP.[3] There can be a number of candidates, each of which is a noun phrase.

$$sentence = (w_1, w_2, ..., w_{i-1}, azp_i, w_{i+1}, ..., w_n) \tag{5.1}$$

$$candidate_k \subset sentence \tag{5.2}$$

We compute the positional features for every (azp, candidate) pair as follows:
- *same_sentence (azp, candidate)*: returns 1 if an AZP and its candidate are in the same sentence, 0 otherwise.
- *find_distance (azp, candidate)*: finds the word distance between an AZP and its candidate. The word distance is normalized between 0 and 1 based on the training instances.

$$s = same\_sentence(azp_i,\ candidate_k) \tag{5.3}$$

$$d = find\_distance(azp_i,\ candidate_k) \tag{5.4}$$

We feed *sentence* into BERT feature extraction mode, which produces the input's *embeddings*. *embeddings* contain BERT pretrained vectors of every word in *sentence*.

$$embeddings = BERT(sentence) \tag{5.5}$$

A word can have one representation or several based on the segment step of Tokenizer. For example, in Figure 5.1 *My* has only one embedding while *sweetheart* has two because it has been segmented into two sub-tokens (sweet and ##heart). In 5.6, 5.7, and 5.8 equations, the subscript of *embeddings* represents the word location in the sentence. μ is a function to compute the mean of a mention representation which can made of several subtoken

[3] An AZP and its candidate may appear in distinct sentences. This could be specified using BERT's parameters 'text_a', and 'text_b'. In such cases, however, we empirically found that we get better results by merging the two sentences into one, and add a [SEP] token in between. Thus, we only use 'text_a'.

embeddings [4].

$$a_1 = \mu(embeddings_{(i-1)}) \tag{5.6}$$

$$a_2 = \mu(embeddings_{(i+1)}) \tag{5.7}$$

$$c_k = \mu(embeddings_{(k)}) \tag{5.8}$$

To obtain a mention representation for an AZP, we compute the average embeddings of the AZP previous word and the next word, and join them together. For every candidate, we calculate the mean of its embeddings which then joined with the positional features. We combine the AZP and its candidate representations to form the input to our classifier.

$$azp = [a_1,\ a_2] \tag{5.9}$$

$$c = [c_k,\ s,\ d] \tag{5.10}$$

$$input = [azp,\ c] \tag{5.11}$$

Our classifier consists of multiple multi-layer perceptrons (MLPs) scoring the <azp, candidate> pair "*input*".

$$layer_1 = f(W_1 \boldsymbol{input} + b_1) \tag{5.12}$$

$$layer_2 = f(W_2\ layer_1 + b_2) \tag{5.13}$$

$$layer_3 = f(W_3\ layer_2 + b_3) \tag{5.14}$$

$$scoring = f(W_4\ layer_3 + b_4) \tag{5.15}$$

f is the ReLU activation function [Nair and Hinton, 2010]. layer$_1$, layer$_2$, layer$_3$, and *scoring* are the resolver's layers; each has learning parameters W and b. After scoring all candidates, we choose the candidate with the highest coreference score as the correct antecedent for the AZP. The overall architecture of our model and data representations are shown in Figure 5.2. In the figure, there is one AZP and two candidates: noun phrase 1 (NP1) and noun phrase 2 (NP2). We run the sentence into BERT to get their word embeddings. AZP is represented with the mean of its previous word, and next word. Candidates are also represented with the mean of their subtoken embeddings, and combined with their positional features. We join each candidate representation with the AZP. We compute <AZP, NP1> and <AZP, NP2> scores which normalized using the softmax layer.

[4] In our experiments, Tokenizer segmented many Arabic text into several sub-tokens, but rarely did segment Chinese.

Number of units in the first layer	3300
Number of units in the second layer	2200
Number of units in the third layer	1200
Number of training epochs	10
Learning rate	1e-5
Dropout rate	0.5
Optimizer	Adam

Table 5.1: Hyperparameter settings.

5.3.1 Candidate generation

For every AZP, we consider as candidate antecedents all maximal and modifier noun phrases (NPs) at most two sentences away, as done by [Chen and Ng, 2016, Yin et al., 2017]. This strategy results in high recall of mentions in both Arabic and Chinese.

5.3.2 Training objective

We minimize the cross entropy error between every AZP and its candidates using:

$$J(\theta) = -\sum_{t \in T}^{n} \sum_{c \in C}^{k} \delta(azp, c) \;\; log(P(azp, c))$$

$$\delta(azp, c) = \begin{cases} 1 & \text{if } c \text{ in } azp \text{ coreference chain} \\ 0 & \text{otherwise} \end{cases}$$

$$\theta = \{W_1, W_2, W_3, W_4, b_1, b_2, b_3, b_4\}$$

θ denotes the set of learning parameters. T consists of the n training instances of AZPs, and C represents the k candidates of an *azp*. *δ(azp, c)* returns whether a candidate *c* is correct antecedent of the *azp*. *log(P(azp, c)* is the predicted log probability of the (*azp*, *c*) pair.

5.3.3 Hyperparameter tuning

We initialize the model's parameters using Glorot and Bengio's [2010] method. We also add a dropout regularization between every two layers. To optimize the classifier, we use the development sets for tuning the hyperparameters. Table 5.1 shows the used settings. We used three GPUs, each is 8GB. The training time is approximately 12 hours.

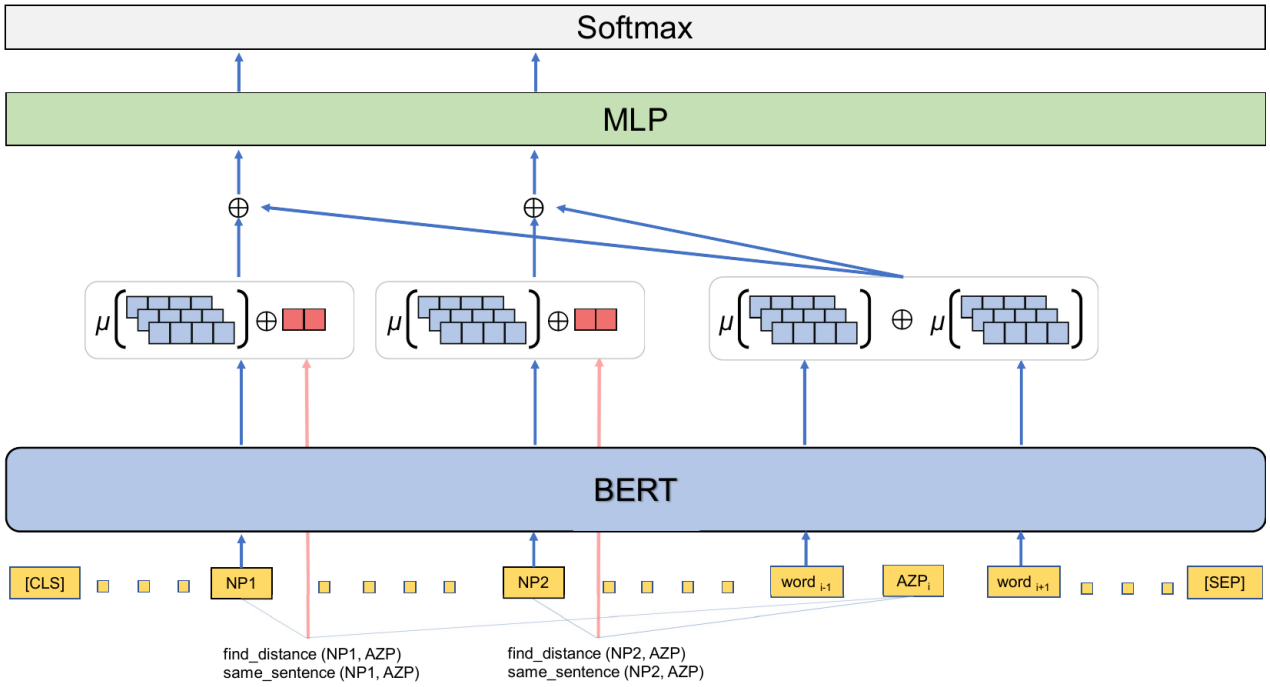

Figure 5.2: An example of one AZP and two candidates: NP1 and NP2. For every candidate, we calculate its task-specific features *find_distance* and *same_sentence*, the features are represented as red concatenated squares. We compute the average embeddings of each candidate and AZP surrounding words, a subtoken embedding is represented as as blue concatenated squares. We form <AZP, NP1> and <AZP, NP2> pairs and feed them into a classifier made of MLPs. The classifier finds their scores which then normalized using Softmax. $\oplus$ is a concatenation operation.

5.4 Results

We compare our results with other published results, and with the results using BERT's two adaptation modes. BERT fine-tuning already has a built-in classification layer on top of the stacked Transformers. The feature extraction mode only produces the learned vectors and needs a framework to be trained on. To do so, we implement a bi-attentive neural network to train feature extraction embeddings and optimize it as done in [Peters et al., 2019] who empirically analyzed fine-tuning and feature extraction modes for a few pretrained models, including BERT. In both modes, we train AZPs and their antecedents without the proposed additional features.

Arabic

We report our results for Arabic in Table 5.2. Given that there was no existing ZP resolver for Arabic, we implemented Chen and Ng's [2016] model and used it as a baseline in our experiments, as it features an extensive range of syntactic, positional, and grammatical features which were then used in other systems as well [Yin et al., 2018].

However, Table 5.2 shows that these features did not work well with Arabic. We can

Model	Recall	Precision	F-score
[Chen and Ng, 2016]	8.1	10.1	8.9
BERT (feature extraction)	47.9	59.5	53.1
BERT (fine-tuning)	50.3	62.5	55.8
Our Model	51.8	64.4	**57.4**

Table 5.2: Arabic AZPs results.

think of two likely reasons for this. First, the size of Arabic OntoNotes is small, thus might not have provided enough training data for the learning phase. Second, some of Chen and Ng's [2016] features might only apply for Chinese; therefore, they might have hurt the performance rather than helped. Also, Chen and Ng's [2016] model lacked morphological features because Chinese morphology is considered relatively simple. In contrast, Arabic morphology is highly derivational and inflectional, and very important for resolving ZPs. Arabic ZPs are preceded by verbs, and verbs encode information about gender, person, and number. The context of ZPs and their antecedents share similar morphological characteristics.

Interestingly, BERT seems to be capable of modeling these morphological connections and resolve correctly many AZPs. BERT's feature extraction and fine-tuning modes produce F-scores of 53.1% and 55.8%. Our model outperforms BERT both modes and achieves an F-score of 57.4%. The incorporated features seem to help with an increase of 1.6% compared to fine-tuning, and 4.3% to feature extraction. The average number of candidates is 14, mostly false candidates. These findings suggest that while BERT learns many details of a language, it might also need more information to achieve the optimal performance.

Chinese

Our experimental results for Chinese can be seen in Table 5.3. The Chinese dataset consists of 6 different categories: Broadcast News (BN), Newswires (NW), Broadcast Conversations (BC), Telephone Conversations (TC), Web Blogs (WB), and Magazines (MZ). The state-of-the-art, attention-based model of [Yin et al., 2018] performs better than the others in all categories except TC. The TC category contains many short sentences; perhaps Yin et al.'s [2017] model struggles to learn short size inputs. Our model achieves the best overall F-score of 63.5% outperforming all prior models in all categories except in (NW). Specifically, our approach outperforms the current state-of-the-art F-scores in these categories: 1.9% (MZ), 7.4% (WB), 10% (BN), 3.2% (BC), and 8.9% (TC). Feature extraction and fine-tuning modes report 60.4% and 62.1% respectively. Fine-tuning process leads to 1.7% increase than feature extraction. Our model outperforms BERT both modes with an increase of 3.1% and 1.4% compared to feature extraction and fine-tuning modes. The average number of candidates is 27, mostly false candidates. The results in Chinese

	NW (84)	MZ (162)	WB (284)	BN (390)	BC (510)	TC (283)	Overall
[Zhao and Ng, 2007]	40.5	28.4	40.1	43.1	44.7	42.8	41.5
[Chen and Ng, 2015]	46.4	39.0	51.8	53.8	49.4	52.7	50.2
[Chen and Ng, 2016]	48.8	41.5	56.3	55.4	50.8	53.1	52.2
[Yin et al., 2016]	50.0	45.0	55.9	53.3	55.3	54.4	53.6
[Yin et al., 2017]	48.8	46.3	59.8	58.4	43.2	54.8	54.9
[Liu et al., 2017]	59.2	51.3	60.5	53.9	55.5	52.9	55.3
[Yin et al., 2018]	**64.3**	52.5	62.0	58.5	57.6	53.2	57.3
BERT (feature extraction)	59.3	48.7	66.0	64.9	57.9	59.5	60.4
BERT (fine-tuning)	61.8	51.8	67.9	66.7	58.7	61.6	62.1
Our model	63.4	**54.4**	**69.4**	**68.5**	**60.8**	**62.1**	**63.5**

Table 5.3: Our proposed model F scores on Chinese ZPs compared with BERT two modes and other models.

(even in Arabic) imply that even though fine-tuning can improve ZP resolution; however, defining more task-related features with BERT feature extraction mode can enhance AZP resolution.

Other versions of BERT were pretrained specifically for English and Chinese. Chinese-only BERT performs better than BERT Multilingual on Chinese texts in some NLP tasks, according to BERT authors' Github page[5]. Therefore, it might also improve the results we obtain with Chinese, although of course adopting that model would defeat the purpose of developing a cross-lingual model.

BERT Layers

Numerous studies show that BERT layers encode rich information about language structure [Aken et al., 2019, Goldberg, 2019, Hewitt and Manning, 2019, Jawahar et al., 2019, Kovaleva et al., 2019]. For a specific NLP task, some layers may carry more useful information than others. In fact, layers that contain indirect information may not lead to the optimal performance. Therefore, it is important to investigate the internal layers and find the most transferable representation. We examined every BERT layer's weights for our model, and report their behaviour on Arabic and Chinese in Figure 5.3. We can see that higher layers produce better F-scores than the lower ones. ZP context and true candidates usually share similar morphological characteristics and semantic relationships and higher layers seem to carry such information.

Therefore, the layers in the last half tend to be more relevant to our task than the layers in the lower half. Generally, F-scores increase as we employ higher layers except when we reach the last two layers. Their slight drops of F-scores might be attributed to BERT training objectives. BERT was trained on masked language modeling (MLM) and next sentence prediction (NSP). Since we are using BERT feature extraction mode, the

[5]https://github.com/google-research/bert/blob/master/multilingual.md

	F score on test set	
BERT Layer(s)	Chinese	Arabic
Third-to-last layer	**63.5**	**57.4**
Last layer	60.9	55.2
First layer	51.2	40.7
Weighted sum of the last 4 layers	63.1	55.2
Weighted sum of all 12 layers	62.4	53.1

Table 5.4: F-scores results when we use different BERT layer(s) for token representations.

last layers were optimized on these pretrained tasks. Even though MLM and NSP helped BERT model learn linguistic aspects in the internal and middle layers, it might have made the last layers biased and specific to their objective goals. The third-to-last (10th layer) and fourth-to-last (9th layer) layers achieve almost equal high F-scores in Arabic and Chinese, but we find the third-to-last to provide more stable states. In our model, we set the third-to-last as the base to produce embeddings for AZPs and their candidates.
We also tried combinations of layers to see if they can produce better representations for the task. Table 5.4 reports the first, last, and third-to-last layer F-scores. We compare their F-scores with two more settings: the weighted sum of the last 4 layers and all of 12 layers. The weighted sum of the 4 layers results in 63.1% for Chinese and 55.2% for Arabic. Chinese F-score decreases only 0.4% and Arabic 2.2% compared to their corresponding third-to-last F-scores. When we calculate the mean of all 12 layers, we get 62.4% and 53.1% for Chinese and Arabic respectively. F-scores drop 1.1% for Chinese and 4.7% for Arabic. The weighted sum of multiple layers did not seem to help improve the ZP resolution task. In both settings, Arabic seems to be more sensitive when several layers involved. Arabic morphology is complex and BERT layers might encode its morpheme interactions in some parts of its layers. Some of these interactions might get lost when multiple layers are weighted sum.

5.5 Discussion

AZP resolution was not considered in the CoNLL-2012 shared task. The main reason is English does not have AZPs while Chinese and Arabic have; so considering AZPs would decrease Arabic and Chinese overall performance [Pradhan et al., 2012]. However, we believe it would have been interesting if there were tracks for coreference resolution with and without AZPs. AZPs are common in pro-drop languages. They are also important to other NLP tasks and reflect real-world applications. While it is possible that AZPs would have posed a challenge to the participants, but it would have broadened our knowledge of AZPs and paved the way for more approaches.

The existing BERTology literature focuses on sentences and words, but little research has been dedicated to null arguments (e.g. AZPs and subject movement). Our results, in

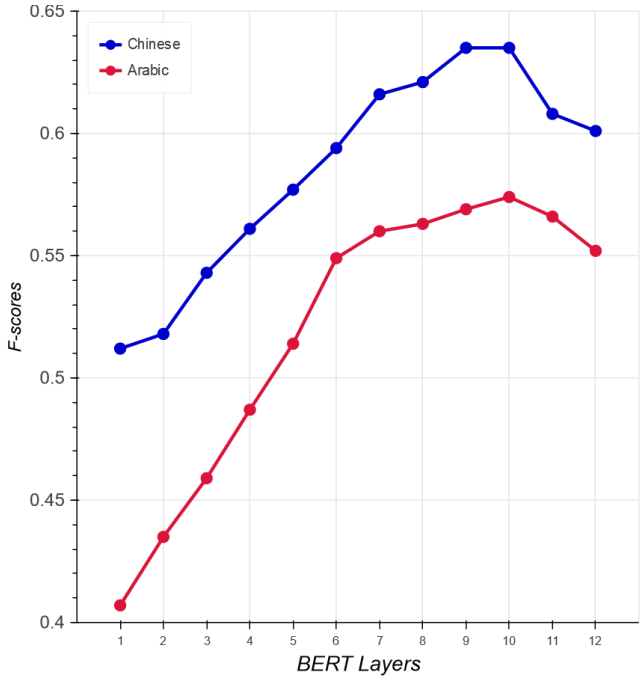

Figure 5.3: Arabic and Chinese F-scores when we use each of BERT layers to produce mention embeddings. Overall, higher layers produce better representations than the lower layers. The 10th layer led to the highest F-scores in both languages.

this and previous Chapters, showed that BERT encodes information about AZPs within its layers; however, more investigation can help us understand how language models learn about AZPs. In the future, we intend to study other parts of language models that can uncover more about AZP learning. For example, attention weights capture the word relations and dependency, and it would have been interesting to see which contexts encode information the most about AZPs. Also, the effect of increasing or decreasing the number of attention heads and Transfomers. Another interesting part to study is the output vectors. The output vectors can tell us the overall representation of a specific word, which then we can use to compare contexts of AZPs and non-AZPs (with overt pronouns).

5.6 Summary

In this Chapter, we proposed a cross-lingual method for AZP resolution and applied it to Arabic and Chinese. This is the first proposal to consider Arabic AZPs and it achieves state-of-the-art results on Chinese. Just like in the previous Chapter, we noticed that BERT encodes information about AZPs and further BERTology investigation on AZPs and their antecedent can shed light on BERT learning and training.

In Chapter 7, we present two methods that resolve AZPs and non-AZPs together. The AZP resolution and identification models are essential components of the two models.

Chapter 6

Data Augmentation for Zero Pronoun Resolution

One of the main obstacles for Arabic coreference resolution is the size of the available corpora. Data availability is a particular problem for Anaphoric Zero Pronouns. There has been various works on pronominal and general-mention anaphora in Arabic; however, there has been little for AZPs. This is because although AZPs are common [Chen and Ng, 2016], they are not always annotated in NLP corpora. There are two reasons for this. First, AZPs have no surface realization and the focus in coreference is usually on arguments realized on the surface [Lee et al., 2005]. Second, pro-drop languages have several types of ZPs, not all of which are anaphoric, which can make it challenging to identify AZPs. Therefore, the number of datasets with annotated AZPs is generally small [Konno et al., 2020]. Human annotation is expensive, cumbersome and time-consuming. To overcome these challenges, we investigate therefore an alternative way to create AZP samples which is using data augmentation methods. These methods for augmenting data existing are automatic, cheap and beneficial. In this Chapter, we use five data augmentation methods to generate and detect anaphoric zero pronouns using open text. We use the augmented data as additional training materials for two anaphoric zero pronoun systems for Arabic. Our experimental results show that data augmentation improves the performance of the two systems, surpassing the state-of-the-art results.

6.1 Motivation

As we have seen in previous Chapters, Arabic corpus size is one of the main obstacles in coreference resolution. To mitigate this problem, there has been various proposals for anaphora annotation, such as, QurAna [Sharaf and Atwell, 2012], AnATAr [Hammami et al., 2009] and others [Seddik et al., 2015]; however, they focus primarily on pronomi-

nal anaphora. There are other types of anaphora that has not received as much attention, one of which is AZPs. As far as we know, OntoNotes is the only corpus that includes annotated AZPs, but their total number is small. Another problem is the expensive process of manually annotating Arabic samples. Motivated by these two reasons, we decided to explore five methods for automatically annotating AZP samples. Another motivation is the optional property of AZPs [Altamimi, 2015] which means we can replace the null argument with a suitable pronoun. This feature permits to expand the augmented AZP samples to work for pronominal anaphora or coreference resolution. Consider the following example from the Arabic portion of OntoNotes (in Arabic OntoNotes 5.0, ZPs are denoted as * in Arabic text).

... عن بوش عدم حماسة للمؤتمر الدولي، اذ انه من البداية، يريد * اجتماعا مختلفا ...

...Bush did not show any enthusiasm for the international conference, because since the beginning, (he) wanted to attend another conference ...

We can convert the AZP into a pronominal anaphora as the following:

... عن بوش عدم حماسة للمؤتمر الدولي، اذ انه من البداية، يريد هو اجتماعا مختلفا ...

...Bush did not show any enthusiasm for the international conference, because since the beginning, he wanted to attend another conference ...

We replaced the AZP gap position with the pronoun he which refers to *Bush*. To know about the antecedent pronoun, we can apply the morphological analyzer of Obeid et al. [2020] which extracts the gender, number and person of the preceding verb, in the example is يريد . After knowing the verb proprieties (as we discussed in Chapter 2), we know which pronoun to replace the AZP with. We can generate many samples for the pronominal anaphora using this method. It is also possible to use the same antecedent mention to replace the AZP gap (e.g. replacing *he* with *Bush*) which can be used for coreference resolution.

6.2 Related Work

Data augmentation is an active research topic and has been applied in different areas of research of NLP [Chen et al., 2021a, Feng et al., 2021]. However, there has been very limited proposals for AZP data augmentation. Zhang et al. [2015] augmented data by replacing a word with its synonyms to improve text classification. Sennrich et al. [2015] augmented data by translating a sequence from one language to another, and then translating the sequence back into the original language. The new data were used to enhance the performance of neural machine translation models. Wang et al. [2018] examined various methods for data augmentation and proposed to randomly replace words in source

and target languages to improve neural machine translation. Şahin and Steedman [2019] removed dependency links and modified tree nodes to create an augmented dataset for Part-of-Speech tagging. Gulordava et al. [2018] replaced words with other words that share the same Part-of-Speech, morphological features, and dependency labels, to improve subject-verb agreement models. [Feng et al., 2019] introduced a pipeline called SMERTI which combines various data augmentation methods, such as, entity replacement, similarity masking, and text in-filling. Grundkiewicz et al. [2019] used a spellchecker to augment training data which are then used to pre-train sequence-to-sequence models. Singh et al. [2019] introduced a cross-lingual data augmentation called XLDA, evaluated on 14 languages on a natural language inference (XNLI) benchmark and question-answering task. XLDA replaces segments of an inputs text with its translation in other languages. [Kumar et al., 2019] proposed DiPS, a model that generates various paraphrased sentences used to train conversational agents and in text summarization tasks. [Andreas, 2019] proposed rule-based data augmentation, which replaces segments of inputs that share similar context to improve the training of n-gram and sequence-to-sequence language models. [Guo et al., 2020] proposed a statistical approach called SeqMix to decide which token to use at each position of an input, and they also provided a framework that combines several data augmentation approaches for several NLP tasks. [Chen et al., 2020] represented datasets as graphs and proposed methods to augment data based on graph theory for paraphrasing. [Ding et al., 2020] trained a language model on the linearized version of the input to synthesize data for low-resource sequence-tagging. [Feng et al., 2020] inserted character-level synthetic noise and word hypernyms to augment data for text generation. [Louvan and Magnini, 2020] used a set of augmentation methods that span words and modify sentences for slot filling and intent detection. [Ri et al., 2021] proposed a method to augment ZPs (not AZPs). They delete personal overt pronouns which results in extra sentences with no pronouns, and use them in Japanese-English translation. Other proposals investigated data augmentation for neural machine translation [Gao et al., 2019, Nguyen et al., 2019, Vaibhav et al., 2019], text classification [Anaby-Tavor et al., 2020, Jindal et al., 2020, Kobayashi, 2018, Wei and Zou, 2019], and question-answering [Kafle et al., 2017, Yang et al., 2019].

Data augmentation methods helped to improve various NLP tasks; however, very few proposals [Konno et al., 2020] considered the methods for AZP tasks which was focused on generating AZPs for resolution. We explore the direction of exploiting several data augmentation methods for AZP resolution and identification and how they can benefit both tasks.

6.3 Methodology

We applied the following five methods to generate AZPs:

1. **OntoNotes Patterns (ONP):** AZPs may occur more frequently in the company of certain verbs. To find the most frequent collocations, we apply the t-test on the Part-of-Speech sequences of AZP sentences. We tried a window of one, two, and three and we found empirically a window of two to detect many correct AZP collocations.

2. **Removing Subject Mention (RSM):** AZPs are dropped subjects of verbs. When an (explicit) mention is the subject of a verb phrase, we remove the mention to obtain an AZP sentence.

3. **Masking Candidate Mentions (MCM):** Also known as contextual data augmentation (CDA). We mask the true antecedents of AZPs and use a language model to find the semantically most similar words, which are then used to produce new sentences by replacing the original word.

4. **Back Translation (BT):** Also known as round-trip translation [Lau et al., 2015]. The AZP examples in OntoNotes are in Arabic. We translate them into English using the GoogleTrans API,[1] and then translate them back to Arabic.

5. **Changing Subject–verb Agreement (CSA):** the verb in an AZP construction and its true antecedent agree on number which can be in singular, dual, or plural form. We change their agreement number from one form to another. For example, if the AZP verb and its reference are in singular form, we change the verb and its reference to dual or plural form.

Table 6.1 shows the number of collected data of each method. AZPs interpretation involves two steps: AZP detection, and AZP resolution. The detection step finds if a sentence has AZPs while the resolution resolves AZPs to their true antecedent. The true antecedent might be present in the same AZP sentence or any previous sentences. In our experiments, we decided to collect AZP samples such that each sample has two sentences: the first has the true antecedent and the second has its AZP. While it is possible to automatically create AZP samples of one sentence only (the AZP position and its true antecedent in the same sentence), but two-sentence samples provide more context and information about AZPs. The augmented dataset will be available at `https://github.com/amaloraini/`

6.3.1 OntoNotes Patterns

AZP sentences may occur more frequently in certain constructions. To find these constructions, we apply a window of one, two, and three tokens on the Part-of-Speeches of AZP samples. We found a window of two (two words before an AZP and two words after as shown in Figure 6.1) to provide many correct collocations. We count the frequencies

[1] https://pypi.org/project/googletrans/

Method	Training
OntoNotes Patterns (ONP)	369
Removing Subject (RSM)	1,196
Masking Candidate (MCM)	736
Back Translation (BT)	501
Changing Subject-Verb Agr. (CSA)	104
Total	2,906

Table 6.1: The number of the augmented data of each method, and their total.

of the constructions and apply the t-test score as in [Manning and Schutze, 1999] to see how probable a construction is:

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}}$$

Where $\bar{x}$ is the sample mean of a Part-of-Speech, and s^2 is its sample variance, N is its sample size, and μ is its distribution mean. We choose the highest five t-test scores and apply them to large public text datasets i.e Wikipedia. We tag the sentences in Wikipedia summary sections with their Part-of-Speech using CAMeL tools [Obeid et al., 2020] and apply the OntoNotes constructions on each sentence except the first. When one sentence matches a pattern, it suggests that the sentence might be an AZP. The AZP true antecedent is usually in the first sentence of the summary section. We join the first sentence and the detected sentence together to represent one AZP sample, as shown in Figure 6.3. In Section 6.3.6, we discuss in details why we follow this approach and the challenges of finding the true antecedent of an AZP.

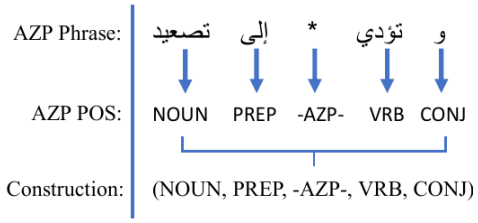

Figure 6.1: An example of one construction of an AZP from OntoNotes. For every AZP position, we extract the POS of two words before and two after, in order to create one construction.

6.3.2 Removing Subject Mentions

Subject nouns or pronouns of a verb phrase might be optional. Removing these subjects transforms a sentence into a null-subject, as shown in the following example:

تقع لندن على نهر التامز

London is located on the Thames river.

In the example, removing 'London' from the Arabic sentence changes it into an AZP sentence and grammatically correct. However, when the subject is removed, readers might not know what the AZP refers unless they have access to more context i.e previous sentences. Therefore, in our experiments each AZP sample has two sentences. The first sentence contains the mention that the AZP refers to, and the second contains the AZP gap.

6.3.3 Masking Candidate Mentions

We use the multilingual BERT[2] to mask antecedents of AZPs and replace them with their most similar semantic token. We replace the antecedent token with [MASK][3], and predict its most similar semantic word. We replace the original mention with the predicted mention from BERT. However, even though the predicted token might be semantically similar to the original token, it might not agree in number or gender with the AZP verb. For example, the antecedent in the original sample can be *student* and BERT replaces it with *students*. Such samples can distort the training for machine learning algorithms because it associates the morphemes with wrong morphological features. We address this problem in Section 6.3.7.

6.3.4 Back Translation

Back (or round-trip) translation is used to introduce variants into the text data [Lau et al., 2015]. It has been shown that translating a sentence from one language to another, and then translating it back to the original language to be beneficial to some NLP tasks [Vaibhav et al., 2019, Xie et al., 2018, Zhang et al., 2019]. Back Translation generates a paraphrased version of the original input adding noise, such as, semantic and syntactical changes. Therefore, we translate the Arabic samples to English, and translate them back to Arabic.

6.3.5 Changing Subject–verb Agreement

AZP verb and its antecedent usually agree on number, gender, and person. We change the verb and its antecedent number agreement from one form to another. For example, if the number of an AZP sample is singular, we change it to dual or plural. To find and generate verb inflections and mention number, we use CAMeL which consists of a set of NLP tools for Arabic.

The five methods we discussed are used create AZP samples, but they are applied differently. The first two (ONP and RSM) are used to detect AZPs on text while the

[2]https://github.com/google-research/bert

[3][MASK] is a special token in BERT which is used for prediction.

other three (CSA, MCM, and BT) are used to generate AZP from existing AZP samples. Thus, we apply the two methods on the Wikipedia summary sections to initially collect AZPs, and then use the other three to generate extra samples. An illustration of all data augmentation methods and steps in Figure 6.2.

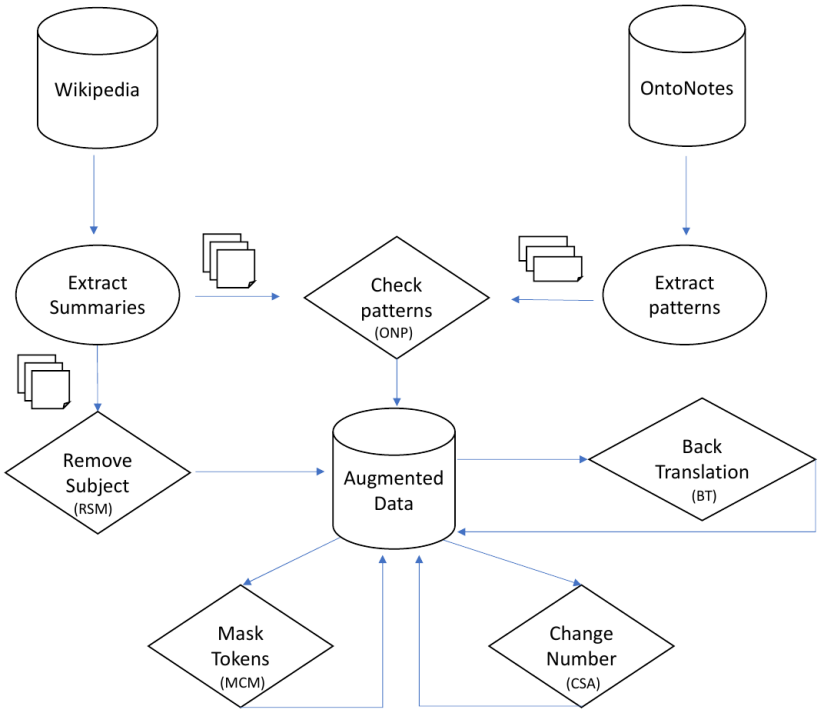

Figure 6.2: Overall picture of all data augmentation methods for Anaphoric Zero Pronouns. We extract patterns from OntoNotes and check them on Wikipedia summary sentences (ONP method). Removing Subject (RSM) is also applied to the summary sentences. after collecting AZP samples, We create extra data applying MCM, BT, and CSA methods.

6.3.6 Finding the True Antecedent

An AZP refers to a preceding mention in a text. The mention can be in the same sentence as the AZP or in previous sentences. Finding the true mention in a sentence using patterns is not trivial because resolving AZPs involves reasoning, context, background knowledge of real world, and deep understanding of a language characteristic, such as, its morphology [Alnajadat, 2017]. At the beginning of the experiment, we apply the two methods (ONP and RSM) on Arabic newspaper articles of Abu El-Khair corpus [El-Khair, 2017] to detect AZP locations. For resolving AZPs to their antecedents, we used AZP context features to find the true antecedent. The main features are the agreement morphemes between the AZP verb and its antecedent. The result was we managed to identify many AZP locations successfully; however, we encountered two problems when we tried finding their true antecedents. First, there were many mentions that share similar features as the true

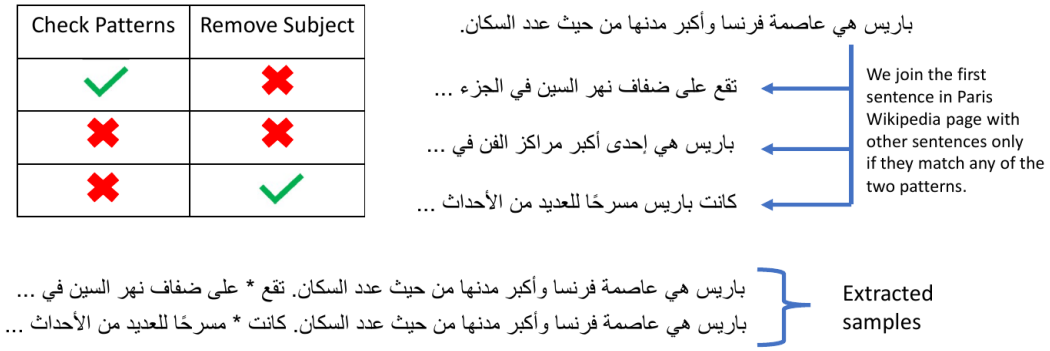

Figure 6.3: Illustration of finding AZPs using OntoNotes Patterns (ONP) and Remvoing Subject Mnetion (RSM) on Paris, from Arabic Wikipedia. Given a Wikipedia page, we join the first sentence with any other sentence in the summary only if they match any of the two data augmentation methods. The extracted samples are made of the first sentence and the matched sentences. The first sentence contains the true antecedent and the second the AZP.

antecedent which led to many incorrect antecedent reference. Second, some locations of the true antecedents were very far from the AZP gap.

To alleviate the above-mentioned limitations of AZP resolution, we apply the two methods on Wikipedia articles. Every Wikipedia articles focuses on a single topic especially in the summary section. Summary sentences can contain AZPs and they usually refer to the Wikipedia title. Therefore, we apply the five methods on the summary sentences, and we consider the title to be the true antecedent of every AZP. The true antecedent always appears in the first sentence i.e the title of Wikipedia page or part of it. So for every AZP in the other sentences of the same page, they refer to the mention in the first sentence. So each AZP sample consists of the first sentence of the Wikipedia page and the sentence where it has the AZP. The reason why we apply the methods only on Wikipedia summary sections because their sentences usually focus on briefly describing the entity of the page. The sentences of other sections might have AZPs but they also include other mentions which can make it challenging to resolve the AZPs to their true antecedent. An illustration of creating one sample from Wikipedia is in Figure 6.3. After collecting the samples using ONP and RSM methods, we apply the other methods MCM, BT, CSA to generate extra samples.

6.3.7 Filtering Samples

After data creation and generation, we found some samples to be grammatically incorrect or not AZPs. For example, sometimes MCM method replaces the original antecedent of an AZP with a mention of a different number or gender. BT method can be problematic in some cases because it predicts the dropped subject which converts an AZP sample into a non-AZP sample. Another problem of BT method is that the AZP verb in the original

Category	Training	Dev	Test
Documents	359	44	44
Sentences	7,422	950	1,003
Words	264,589	30,942	30,935
AZPs	3,495	474	412

Table 6.2: Basic statistics about AZPs in Arabic OntoNotes. The total number of AZPs is 3495, 474, and 412 for training, development, and test respectively.

sample is used to track the AZP location. If the AZP verb is translated back into another similar verb, the AZP location can not be retrieved back. To enhance the quality of the augmented data produced by the MCM method, we use Camel tools [4] which has a morphological analyser to find the gender and number of mentions. We use the analyser to see if the AZP verb and its antecedent agree on number and gender. For the grammatically incorrect cases, we remove samples that do not have complete agreements. For the added subject after the BackTranslation step, we remove the subjects in a similar way as we do in RSM method. If the verb of the original sample is absence after the BackTranslation, we remove the sample.

6.4 Evaluation

6.4.1 Dataset

Most existing Arabic corpora do not have AZP annotated. As far as we know, OntoNotes 5.0 is the only source that has Arabic AZPs annotated. The distribution of documents, sentences, words and AZPs in OntoNotes are in Table 6.2.

6.4.2 AZP systems

For AZP identification, we use the model by [Aloraini and Poesio, 2020a] which is a binary classifier that takes a candidate ZP location as input, and classifies whether it as an AZP or not. For AZP resolution, we use the model by [Aloraini and Poesio, 2020b] which combines BERT representations and additional task-related features of AZPS to learn their true antecedent. We adopt their systems and apply the same settings on Arabic AZPs. We then evaluate each method of our data augmentation for both systems to see its effect. We then try using all the collected data, and try various binary combinations as we will see in Section 6.5.

[4] https://github.com/CAMeL-Lab/camel_tools

Original text	لا بَأْسَ أَن تُقالَ
pre-processed text	لا باس ان تقال

Table 6.3: An example of how we pre-process Arabic text. We normalize all 'alif' letters forms, and remove all diacritics.

6.4.3 Data Preprocessing

Arabic is a morphologically rich language with a large set of morphological properties. Arabic text can suffer from sparsity (different forms for the same word) and ambiguity (same form for numerous words) if the text is not pre-processed. There are two reasons for these problems. First, certain letters can have different forms which are usually mis-spelled, such as the various forms of the letter "alif". Second, same word can have a fully diacritized, partially diacritized or undiacritized forms. Retaining diacritics can complex word representation and model training [Habash and Sadat, 2006]. Pre-processing Arabic text improves the overall performance. For example, [Aloraini et al., 2020] followed the pre-processing steps by [Althobaiti et al., 2014a] which led to a significant improvement in the performance of Arabic coreference resolution with an increase of 7% F1 score more than the baseline. Therefore, we follow their steps for pre-processing text which are:

- We normalize the various forms of the letter "alif" (آ,أ,إ,ٱ) to the letter "ا".
- We remove all diacritic marks on words, such that each word has its undiacritized form.

We show an example of an original and pre-processed sentence from OntoNotes in Table 6.3.

6.5 Results

We use the systems from [Aloraini and Poesio, 2020a] and [Aloraini and Poesio, 2020b] as baselines for AZP identification and resolution respectively. To see the effect of the collected samples, we use their systems following the same settings. We evaluate the augmented data of each method with different settings, and see their performance on the precision, recall, and F1 scores.

AZP identification in Table 6.4, we train the baseline system with every data augmentation method and evaluate on the test portion. We see the results vary. RSM and BT improve the scores with an increase of +0.4 and +0.1 compared to the baseline F1 score. ONP and CSA hurt the scores with a decrease of -0.3 and -0.7 compared to the baseline F1 score. When we train the system with all the augmented data, it outperforms the baseline

with an increase of +0.3, but still less than when we use RSM by itself. RSM method provides the best performance and outperforms the baseline and all other settings scores.

AZP resolution we evaluate each augmented-data method and the results are shown in Table 6.5. Almost every data augmentation method increases the overall performance when it is trained with CoNLL-2012 except for CSA method. ONP, RSM, MCM, and BT increase the F1 score with +0.5, +1.6, +0.8, and +1.0 F1 scores respectively while CSA decreases the score by -0.2, compared with the baseline's F1 score. The best settings when we train all the augmented data with CoNLL-2012 which results an increase of +1.8.

Combinations we also try combining two methods together to see if they can outperform when we use every method separately. We show the AZP identification results in Table 6.6 and Table 6.7 for AZP resolution. As shown in the tables, some combinations can improve the scores; however, no single combination outperforms the highest scores in Tables 6.4 and 6.5.

6.6 Discussion

The augmented data improved the results for both AZP identification and resolution. The improvements in scores for AZP resolution are more than AZP identification. To understand the reasons, we investigated the Arabic portion of CoNLL-2012 and the outputs of the two systems. In CoNLL-2012, we found that many AZP samples to be duplicates but they refer to different types of mentions. The augmented data has more variance of AZP verbs; however, many were not present in the test part which might have led the model to associate these newly seen AZP verbs to cases that are not AZPs in the test set. We also examined the identified and resolved AZPs, and we found several cases to be verbs or mentions that are not subset of the training or the augmented data. Even though the data augmentation methods we showed helped to alleviate this problem, we still need more annotated data and explore more data augmentation methods.

The main obstacle of Arabic AZPs is its size and quality. As far as we know, OntoNotes is the only existing dataset that has AZPs and their total number is small, as we can see in Table 6.2. While data augmentation methods can mitigate the problem and improve the results, but not by a very significant margin. We believe a large-scale and high quality dataset can improve the results of AZP tasks.

In the future, we intend to automatically extend the augmented dataset to other NLP tasks (e.g. pronominal resolution) by replacing the AZP with the antecedent pronoun or the noun itself.

Training settings	Test Evaluation			
	P	R	F1	diff
CoNLL-2012 [b]	**60.0**	78.9	68.2	-
+ ONP	59.4	79.3	67.9	-0.3
+ RSM	59.8	**80.4**	**68.6**	**+0.4**
+ MCM	59.6	79.8	68.2	0
+ BT	59.6	80.2	68.3	+0.1
+ CSA	59.3	78.5	67.5	-0.7
CoNLL-2012 + all	59.8	80.2	68.5	+0.3

Table 6.4: AZP identification training settings with each data augmentation technique, and their results on the test set. [b] is the baseline scores [Aloraini and Poesio, 2020a]. *diff* is the difference between a method's F1 score and the baseline.

Training settings	Test Evaluation			
	P	R	F1	diff
CoNLL-2012 [b]	64.4	51.8	57.4	-
+ ONP	64.8	52.4	57.9	+0.5
+ RSM	65.6	53.7	59.0	+1.6
+ MCM	65.3	52.6	58.2	+0.8
+ BT	65.3	52.9	58.4	+1.0
+ CSA	64.4	51.6	57.2	-0.2
CoNLL-2012 + all	**65.8**	**53.9**	**59.2**	**+1.8**

Table 6.5: AZP resolution training settings with each data augmentation technique, and their results on the test set. [b] is the baseline scores [Aloraini and Poesio, 2020b]. *diff* is the difference between a method's F1 score and the baseline.

Training settings	Test Evaluation			
	P	R	F1	diff
CoNLL-2012 [b]	**60.0**	78.9	68.2	-
+ ONP + RSM	59.6	79.7	68.2	0.0
+ ONP + MCM	59.5	79.7	68.1	-0.1
+ ONP + BT	59.8	79.8	68.3	+0.1
+ ONP + CSA	59.7	78.8	67.9	-0.3
+ RSM + MCM	59.8	79.0	68.1	-0.1
+ RSM + BT	59.8	79.8	68.4	+0.2
+ RSM + CSA	59.8	**80.2**	**68.5**	**+0.3**
+ MCM + BT	59.8	79.7	68.3	+0.1
+ MCM + CSA	59.7	79.7	68.2	0.0
+ BT + CSA	59.7	79.0	68.0	-0.2

Table 6.6: AZP identification training settings with different combinations of data augmentation methods, and their results on the test set.

Training settings	Test Evaluation			
	P	R	F1	diff
CoNLL-2012 [b]	64.4	51.8	57.4	-
+ ONP + RSM	65.3	53.7	58.9	+1.5
+ ONP + MCM	65.0	53.2	58.5	+1.1
+ ONP + BT	64.9	53.5	58.6	+1.1
+ ONP + CSA	64.6	52.7	58.0	+0.6
+ RSM + MCM	65.1	53.6	58.7	+1.4
+ RSM + BT	**65.4**	**53.8**	59.0	**+1.6**
+ MCM + BT	64.9	52.9	58.2	+0.8
+ MCM + CSA	65.1	53.4	58.6	+1.2
+ BT + CSA	64.6	52.4	57.8	+0.4

Table 6.7: AZP resolution training settings with different combinations of data augmentation methods, and their results on the test set.

6.7 Summary

In summary, the Chapter contributes the following:

- Various data augmentation methods to detect potential AZPs in unannotated sentences, and to generate sentences containing AZPs.
- A method to automatically find the true antecedent of AZPs.
- The augmented data improve AZP identification and resolution for Arabic, and surpass the current state-of-the-art results.

We also analyzed the errors of the AZP models and explained why they had failed to resolve some samples. We also discussed a few ideas on how to expand the augmented dataset so it works with other NLP tasks, such as, pronominal and coreference resolution.

Chapter 7

Resolving AZPs and non-AZPs jointly

In previous chapters, we proposed methods that cluster mentions and anaphoric zero-pronouns (AZP) separately. Early works suggested that learning both tasks can be challenging [Iida and Poesio, 2011] because AZPs cannot carry features similar to overt mentions, such as Part-of-Speech, word size and other surface features. Recently, however, it has been shown that BERT can encode mention features within its representations [Jawahar et al., 2019]. We have also shown in Chapters 4 and 5 that BERT can encode information about AZPs within its layers. This suggests it may be possible to address the challenges raised in earlier work. However, there have been few proposals that discussed incorporating both types using language models in a single learning framework. In this Chapter, we discuss two methods for clustering AZPs and non-AZPs jointly. We investigate how to cluster AZPs and non-AZPs in *pipeline* and *joint learning* settings. In pipeline fashion, we first learn how to cluster and identify mentions. Next, the pipeline learns how to detect AZPs, which are then linked to their corresponded clusters. In a joint learning setting, AZPs have an explicit representation (using the distinguished marker *pro*). This way, coreference resolution models would treat AZPs as any other mention when they have a surface realization. We apply that only for the training phase. For the test phase, we use an AZP identification model to tag AZP gaps which prepares the input for clustering. We compare these two proposed models with a recent proposal.

7.1 An Extended Version of the CoNLL Arabic dataset with Anaphoric Zero Pronouns

The goal of the CoNLL-2012 coreference shared task is to learn coreference resolution for three languages (English, Chinese and Arabic). However, AZPs were excluded from the task even though they are annotated in OntoNotes for Arabic and Chinese. This was because considering AZPs decreased the overall performance on Arabic and Chinese, but not on English [Pradhan et al., 2012] (As already mentioned, English is not a pro-drop language). So in order to study joint coreference resolution for explicit mentions and zero anaphors, we had to create a novel version of the CoNLL-2012 dataset in which AZPs and all related information were included.

The CoNLL-2012 annotation layers consists of the following [Pradhan et al., 2012]:

1. Document ID: Contains the file name.
2. Part number: Some files are divided into several files and this number shows the sentence number.
3. Word number: Word position in the sentence.
4. Word itself: This represents the tokenized token.
5. Part-of-Speech: The Part-of-speech of the word.
6. Parse bit: This is the bracketed structure broken before the first open parenthesis in the parse, and the word/part-of-speech leaf is replaced with a *.
7. Lemma: Used to show the gold and predicate lemma.
8. Predicate Frameset ID: This is the PropBank frameset ID of the predicate in Column 7.
9. Word sense: The word sense.
10. Speaker/Author: The speaker or author name, where available. Mostly in broadcast conversation and web log data. However, this is not available for Arabic because all texts are extracted from newspapers.
11. Named Entities: Named entity for the word.
12. Arguments: Predicted and gold arguments.
13. Coreference: Coreference chain which can be single or multiple tokens.

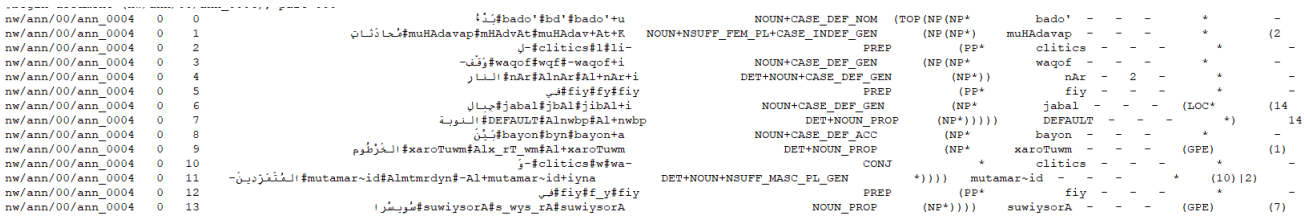

Figure 7.1: A screenshot of one of the CoNLL-2012 files, ann_0004. The 13 annotation layers are the columns which are separated by multiple spaces.

An example of one of the CoNLL-2012 files is in Figure 7.1.

In Chapters 3 and 4, we applied AZP systems on OntoNotes Normal Forms (ONF). They are annotated with AZPs and other mentions; however, they are not as well-prepared as CoNLL-2012. To create a CoNLL-like dataset with AZPs, we extract AZPs from ONF and add them to the already-existing CoNLL files. The goal of the new dataset is to be suitable for clustering AZPs and overt mentions, and can be compared with previous proposals that did not consider AZPs and as well as with future works that consider them.

To include AZPs and their information (e.g., Part-of-Speech and parse tree) to CoNLL-2012, we can use ONF. However, while adding AZPs to the clusters, we realized that some coreference chains only exist in ONF, but not in CoNLL-2012. These are clusters consisting of only one mention and one AZP, as in the example illustrated in Figure 7.2. Chain 71 has two mentions, an AZP (denoted with *) and a mention. Since CoNLL-2012 does not consider AZPs in coreference chains, this cluster would only have a single mention (i.e. singletons) and OntoNotes is only annotated with multi-mention entities. Our new dataset includes AZPs and become a cluster with multi-mentions; therefore, such clusters should be included. To add them to the existing CoNLL-2012, we have to assign them a new cluster. We thus wrote a script that automatically extracts AZPs from ONF and adds them in CoNLL-2012 following these steps:

1. Finds all clusters that have AZPs in ONF and extracts them.

2. Each extracted AZP is either:

   (a) Clustered with two or more mentions: For this case, CoNLL has already assigned a coreference-chain number and we assign the AZP to the same number.

   (b) Clustered with only one mention: We create a new cluster that includes the single mention and the AZP.

3. Writes other relevant information, such as, Part-of-Speech, syntax, and all other layers.

Adding AZPs to CoNL-2012 is beneficial to learn how to resolve them with other mentions or can be useful for future CoNLL-shared tasks and any other related NLP task. After

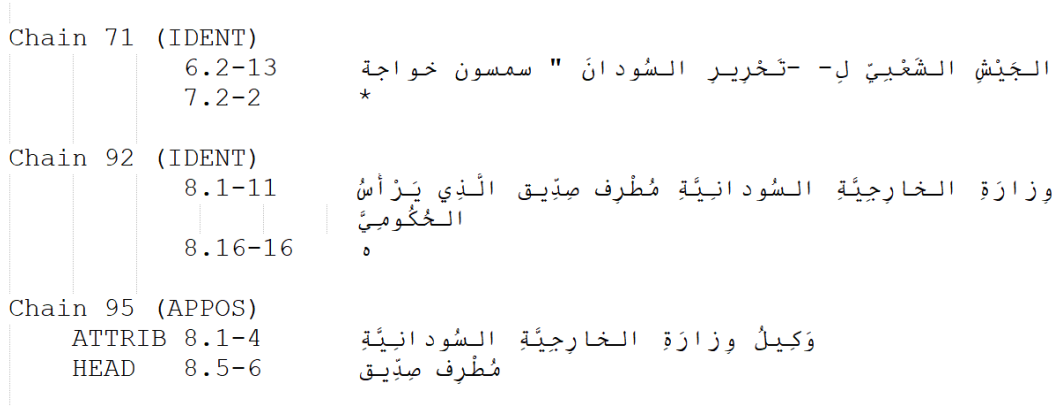

Figure 7.2: A screenshot of OntoNotes Normal Forms (onf). Chain 71 is not considered part of a CoNLL-2012 shared task because the cluster would become a singleton when we remove the AZP (denoted as *).

preparing the new CoNLL dataset as discussed, we used it to train our methods. This new version of Arabic Ontonotes will be made available with the next release of Ontonotes.[1]

7.2 Models

In this section, we show our methods for resolving AZPs and non-AZPs for coreference resolution and they are advantageous for AZP resolution. Earlier proposals resolved AZPs based on the antecedents that are in the same sentence as the AZP or two sentences away [Aloraini and Poesio, 2020b, Chen and Ng, 2015, 2016, Liu et al., 2017, Yin et al., 2016, 2017, 2018]. However, it has been shown that learning mention coreference in the whole document is beneficial for AZP resolution [Chen et al., 2021b]. Therefore, we apply two different methods for resolving AZPs using clusters and coreference chains. The *pipeline* resolves AZPs based on the output clusters from the coreference resolution model while the *joint learning* learns how to resolve AZPs from the coreference chains, we show an example of these two in Figure 7.3. In the example, the *pipeline* resolves AZPs to clusters, instead of mentions and the *joint learning* finds the coreference chains for mentions, including AZPs. Earlier proposals suffered from two main problems. First, they consider a limited number of candidates (i.e mentions in two sentences away from the AZP) as possible true antecedents; however, the true antecedent might be far away from the AZP. Second, other mentions can share salient context as the true antecedent which can introduce more noise to the learning. Our methods mitigate these problems by considering all mentions in the document and employing more relevant information. The *pipeline* resolves AZPs based on clusters which decreases dramatically the number of AZP candidates. The *joint learning* resolves AZPs using coreference chains which incorporates broader context for AZPs, insufficient contexts results in many errors [Chen and Ng, 2016].

[1] Sameer Pradhan, p.c.

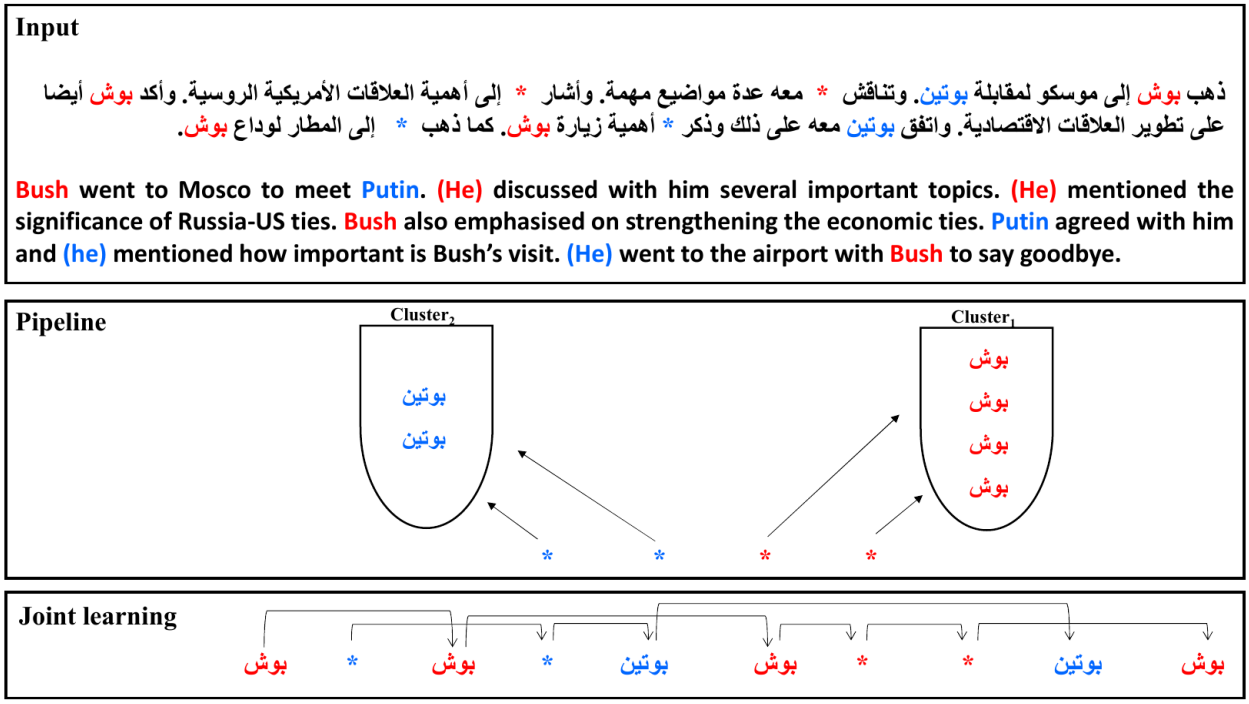

Figure 7.3: The input is a document with mentions. The asterisk * represents the AZPs in the text. For AZP resolution, The *pipeline* resolves AZPs with the output clusters and the *joint learning* resolves AZPs based on coreference chains.

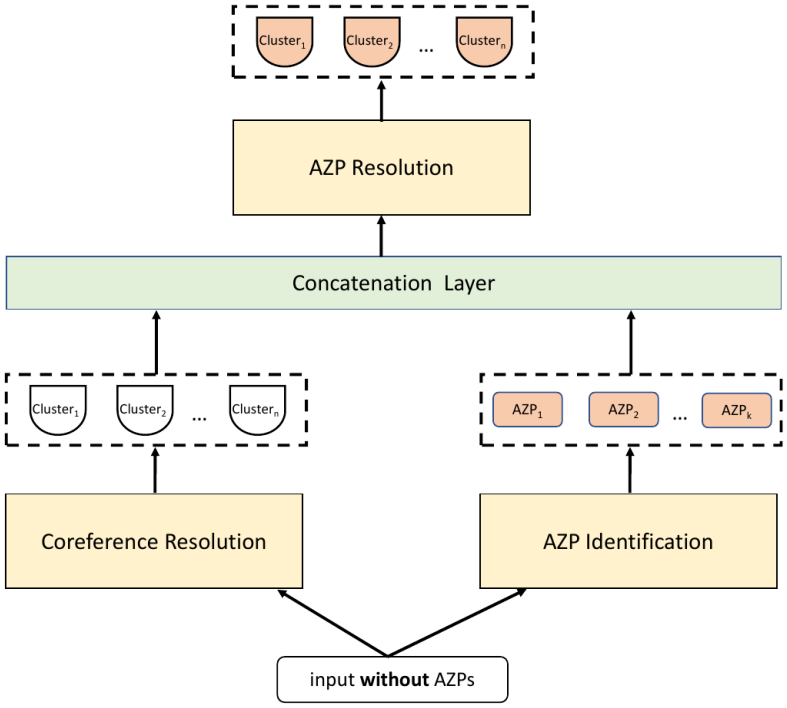

Figure 7.4: The input without AZPs is fed into the *Coreference Resolution* and *AZP identification* models. The outputs of the two models are clusters and AZPs respectively. Their representations are concatenated, and then their coreference information is learned through the *AZP Resolution* model.

7.2.1 Pipeline

In a *pipeline* setting, the inputs are the extended version of CoNLL, the one we described in Section 7.1. Each file consists of multiple sentences and we follow the same splits in CoNLL-2012 [Pradhan et al., 2012] for train, development and test. We initially fed the documents for training into two models: *coreference resolution* and *AZP identification*. We used the Arabic coreference resolution by Aloraini et al. [2020] and the proposed AZP identification by Aloraini and Poesio [2020a]. The outputs of coreference resolutions are clusters and each one has its own mentions. The outputs of the AZP identification are the predicted gap positions of AZPs. The *AZP resolution* model by Aloraini and Poesio [2020b] learns how to resolve the identified AZPs with their clusters. We show how we represent the input in the following:

The *input* is a document with multiple sentences separated with periods, and has a total of *n* words. The *input* does not consider AZPs initially, they are masked.

$$input = (w_1, w_2, w_3, ..., w_n) \tag{7.1}$$

We first feed the *input* into the *coreference resolution* model which outputs the mention clusters, c_1, c_2, to the last cluster index, *k*.

$$output\_clusters = coref\_res(input) \tag{7.2}$$

$$output\_clusters = (c_1, c_2, ..., c_k) \tag{7.3}$$

After finding the coreference clusters, the *AZP Identification* model predicts the AZP positions in two steps [2]. First, the *AZP identification* uses a Part-of-Speech tool to tag words and mark gaps before verbs as potential AZPs. Second, the *AZP identification* classifies these marked gaps as AZPs or not. Therefore, not every gap between words has an AZP. For example, in (7.5) there is no AZP between the words w_2 and w_3, but there is one between w_1 and w_2 (i.e. a_j). We find AZP locations and extract their positions.

[2] We explained these steps in details in Chapter 4.

$$input\_with\_azp = AZP\_Id(input) \tag{7.4}$$

$$input\_with\_azp = (w_1, a_i, w_2, w_3, ..., a_k, w_n) \tag{7.5}$$

$$AZPs = (a_i, ..., a_k) \tag{7.6}$$

$$ss_{a_i} = same\_sentence(a_i, c_j) \tag{7.7}$$

$$cd_{a_i} = cluster\_distance(a_i, c_j) \tag{7.8}$$

$$azp_i = (a_i\_previous, a_i\_next, ss_{a_i}, cd_{a_i}) \tag{7.9}$$

We follow the same representation for AZPs as Aloraini and Poesio [2020b]:

- embeddings for previous word to AZP.
- embeddings for next word to AZP.
- Whether the AZP and the candidate entity (represented either as the last mention or first mention) are in the same sentence or not.
- The distance between the AZP and its cluster representation.

All four AZP features are concatenated, as shown in 7.8.

Clusters can be represented in different ways, including, e.g, the representation of the first mention or the last mention. We found empirically that representing clusters with the nearest mention to the AZP (the last added mention to the cluster) produces the best results.

$$c_j = \{m_1, m_2, ..., m_z\} \tag{7.10}$$

$$c_j = \begin{cases} m_1 & \text{the first mention to represent } c_j \\ m_z & \text{the last mention to represent } c_j \end{cases} \tag{7.11}$$

Next, the AZP and cluster representations are joined together through a concatenation layer. The variable *input* contains the concatenated representation of a mention pair - the AZP and its corresponding cluster. The binary variable *AZP res* receives *input* and is 1 if the AZP and the cluster corefer. The model also outputs the final clusters.

The following equations specify how the output of the network is computed:

$$input = concat(azp_i, c_j) \tag{7.12}$$

$$input = [azp_i, c_j] \tag{7.13}$$

$$results = AZP\_Res(input) \tag{7.14}$$

$$results = (r_1, r_2, ..., r_s) \tag{7.15}$$

The variable *results* consists of the final clusters of the resolved AZPs and non-AZPs. We show the model architecture in Figure 7.4.

Original CoNLL-2012 sentence	كانا في الوضع نفسه
Extended CoNLL-2012 sentence	كانا *pro* في الوضع نفسه

Table 7.1: An example of how we explicitly represent AZPs.

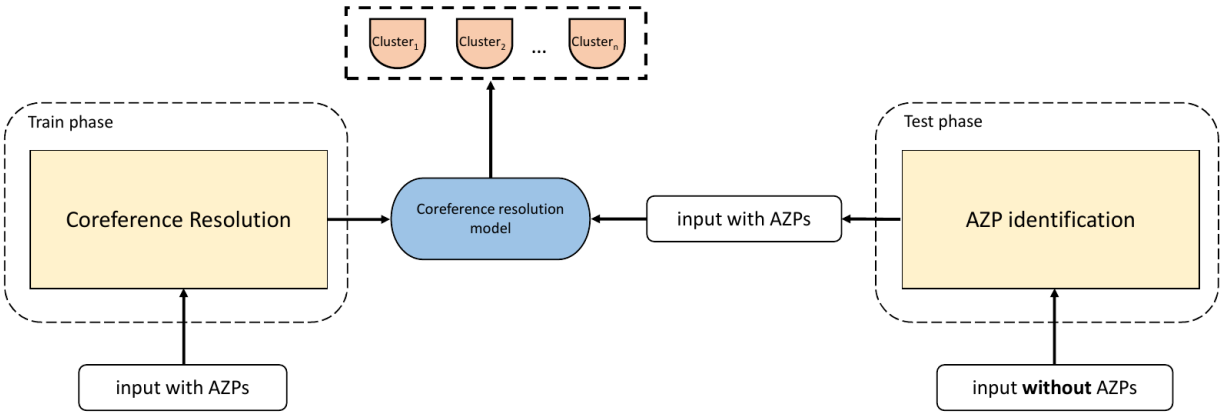

Figure 7.5: In the train phase, the model learns how to resolve mentions and AZPs. AZPs are represented with the *pro* tag and treated like any other mention. The test phase predicts and tags AZPs locations. We use the model proposed by Aloraini and Poesio [2020a] to find AZPs. The pretrained coreference resolution model is used in the test phase to cluster mentions and AZPs.

7.2.2 Joint Learning

In the *joint learning* setting, the model learns to resolve AZPs by using the explicitly represented AZP gaps. This way, AZPs would be learned as any other overt mention. In our extended CoNLL-2012 documents, AZPs have the special identified *pro*. Table 7.1 shows an example of a CoNLL-2012 original sentence and its extended version. However, we consider AZPs only in the training phase when we apply the coreference resolution model. At test time, AZPs are not considered, same as in a real life application. Instead, we use the *AZP identification* model by Aloraini and Poesio [2020b] to tag AZP gaps.

After tagging, the input is ready for clustering using the trained coreference resolution model. This is how we represent the inputs for both training and testing:

The *input* is a CoNLL-2012 document with many sentences that has a set of *n* mentions. A mention can be a word or an AZP tag (*pro*).

$$input = (m_1, m_2, m_3, ..., m_n) \tag{7.16}$$

The variable *input* is fed into the *coreference resolution* (coref_res) model which outputs clusters. The clusters contain mentions and AZPs that refer to the same entity.

$$output\_clusters = coref\_res(input) \tag{7.17}$$

$$output\_clusters = (c_1, c_2, ..., c_k) \tag{7.18}$$

For the test phase, we assume a document is not labeled with AZP tags, which reflects real-life applications. Therefore, we first feed *input* into the *AZP identification* (AZP_Id) which outputs *input_with_azp*, that is *input* but with tagged AZPs. The *AZP identification* is pre-trained on the train set of CoNLL-2012 to detect AZP locations.

$$input\_with\_azp = AZP\_Id(input) \tag{7.19}$$

$$input\_with\_azp = (w_1, a_i, w_2, w_3, ..., m_n) \tag{7.20}$$

After preparing *input_with_azp* which has words and AZP tags, we feed it into the *coreference resolution* model which outputs the final clusters.

$$results = pretrained\_coref\_res(input\_with\_azp) \tag{7.21}$$

$$results = (r_1, r_2, ..., r_s) \tag{7.22}$$

The variable *results* has the resolved AZPs and non-AZPs. We show the overall architecture in Figure 7.5.

7.3 Other Proposals

Most existing works regard coreference resolution and AZP resolution as two independent tasks. There have been few proposals on solving the two tasks jointly. Iida and Poesio [2011] integrated the AZP resolver with a coreference resolution system using an integer-linear-programming model. Kong and Ng [2013] employed AZPs to improve

the coreference resolution of non-AZPs using a syntactic parser. Shibata and Kurohashi [2018] proposed an entity-based joint coreference resolution and predicate argument structure analysis for Japanese. However, these works relied on language-specific features and some assumed the presence of AZPs.

There are two end-to-end neural proposals about learning AZPs and non-AZPs together. The first is by Yang et al. [2022] who created a framework called CorefDPR. The framework consists of four components: the input representation layer, coreference resolution layer, pronoun recovery layer and general CRF layer. The second proposal is by Chen et al. [2021b]. They represent tokens and gaps using an encoder. They also proposed a two-stage mechanism between AZPs and mentions. In our experiments, we only compare our results with the latter proposal because the former only evaluated the Chinese conversational speech [3] of OntoNotes. In addition, their model is also not publicly available which makes it difficult to compare our results with theirs.

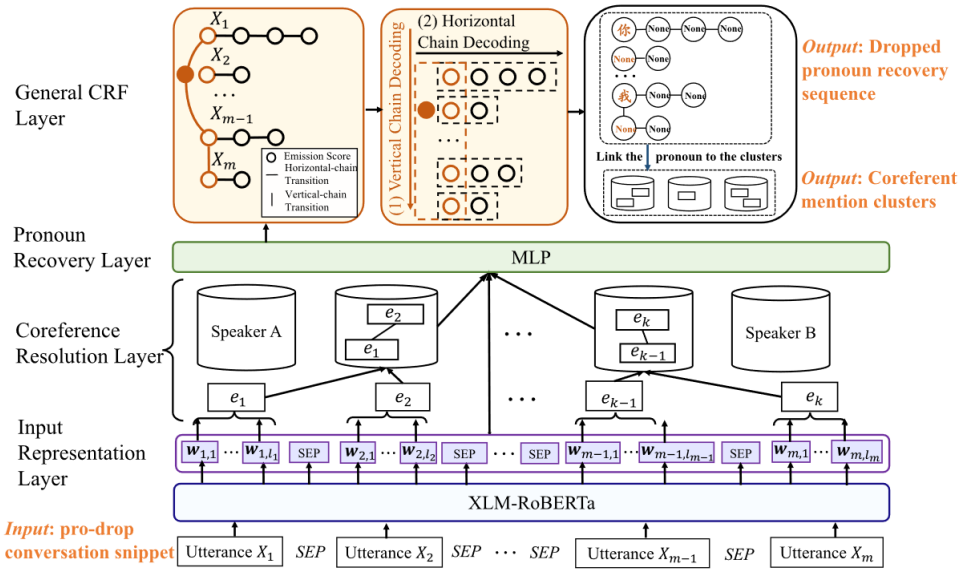

Figure 7.6: CorefDPR consists of four components: input representation layer, coreference resolution layer, pronoun recovery layer and general CRF layer [Yang et al., 2022].

7.3.1 Evaluation metrics

Coreference resolution

For our evaluation of the coreference system, we use the official CoNLL-2012 evaluation metrics to score the predicted clusters. We report recall, precision, and F1 scores for MUC, B^3 and $CEAF_{\phi_4}$ and the average F1 score of those three metrics.

[3] The TC part of the Chinese portion in OntoNotes.

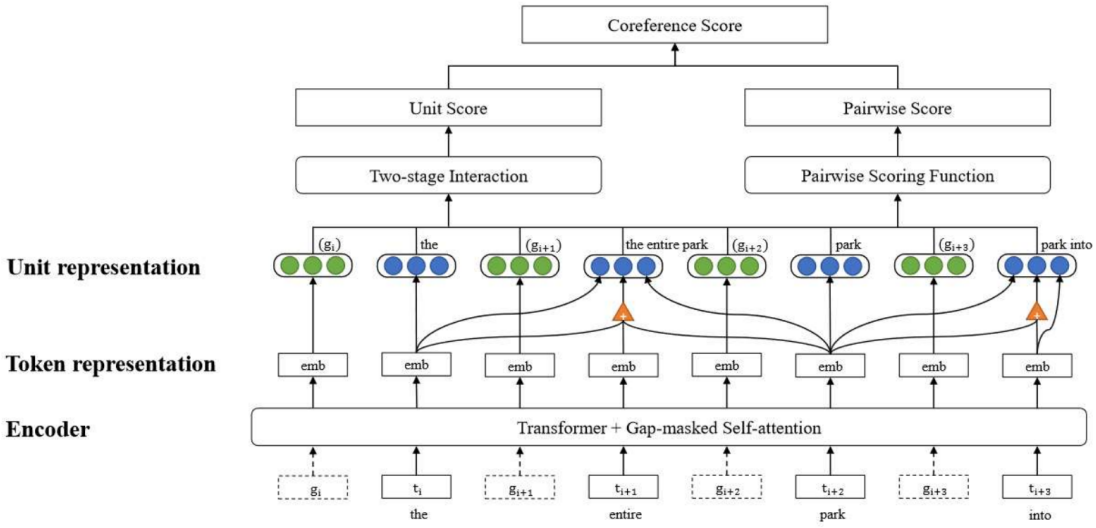

Figure 7.7: Resolving AZPs and non-AZPs in one model [Chen et al., 2021b].

AZP resolution

We evaluate AZP resolution in terms of recall, precision and F-score, following the same metrics in Zhao and Ng [2007]:

$$Recall = \frac{AZP\,hits}{Number\ of\ AZPs\ in\ Key}$$

$$Precision = \frac{AZP\,hits}{Number\ of\ AZPs\ in\ Response}$$

Key represents the gold set of AZP entities in the dataset, and *Response* represents the the predicted resolved AZPs. *AZP hits* are the reported resolved AZP positions in *Response* which occur in the same position as in *Key*.

7.3.2 Training Objectives

Pipeline

The training objective of the AZP identification is binary cross-entropy loss, same as in Aloraini and Poesio [2020a]:

$$AZP\_Id\_Loss(\theta) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] \tag{7.23}$$

θ is the set of learning parameters in the model. N is the number of training samples in the extended CoNLL-2012. y_i is the true label *i* and $\hat{y}_i$ is its predicted label.

For the AZP resolution, the goal is to minimize the cross entropy error between every

AZP and its antecedents, as defined in Aloraini and Poesio's [2020b] model; however, we resolve AZPs with clusters, instead of mentions:

$$\textit{AZP\_Res\_Loss}(\theta) = -\sum_{t \in T}^{n} \sum_{c \in C}^{k} \delta(azp, c) \log(P(azp, c)) \tag{7.24}$$

θ denotes the set of learning parameters. *T* consists of the *n* training instances of AZPs, and *C* represents the *k* candidate clusters from the coreference resolution. *δ(azp, c)* returns whether a candidate cluster *c* is the correct one for the *azp*, or not. *log(P(azp, c)* is the predicted log probability of the (*azp*, *c*) pair.

The training objective of the coreference resolution is to optimize the log-likelihood of all correct mentions [Lee et al., 2017], as the following :

$$\textit{Coreference\_Resolution\_Loss}(\theta) = \log \prod_{i=1}^{N} \sum_{\hat{y} \in \mathcal{Y}(i) \cap \mathrm{G}(i)} P(\hat{y}) \tag{7.25}$$

G represents the spans in the gold cluster that includes *i*.

Joint Learning

In the *joint learning*, we only use the (7.25) for training. AZPs are treated as any other mention; therefore, they become part of the coreference resolution learning objective. We also do not have to train the AZP identification model because we only use the AZP identification in the test phase and we use the pre-trained one on the original CoNLL-2012 from Aloraini and Poesio [2020a].

7.4 Results

We compare the results of the *pipeline* and *joint learning* models with the results of Chen et al. [2021b]. We followed Chen et al. [2021b] for hyperparameter tunning, but we changed the language model to AraBERT-base. We evaluate two tasks. First, we assess the results at joint coreference resolution of both AZPs and non-AZPs. Second, we evaluate AZP resolution only. Unlike previous proposals that resolve AZPs with their antecedents, the AZPs of our methods and the Chen et al.'s [2021b] model are resolved differently. The *pipeline* uses the output clusters, the *joint learning* uses the coreference chains and Chen et al. [2021b] uses two scoring components.

7.4.1 Resolving AZPs and non-AZPs

In Table 7.2, we see the results of resolving AZPs and non-AZPs. Chen et al.'s [2021b] model achieves 64.2% F1 score, which is 0.7% more than the *pipeline*, but less than the *joint learning* with 2.9%. Our *joint learning* approach outperforms our *pipeline* and Chen et al.'s [2021b] system, achieving the best F1 average score of 67.1%.

Models	MUC			B^3			$CEAF_{\phi_4}$			Avg.
	R	P	F1	R	P	F1	R	P	F1	F1
Pipeline	62.9	70.7	66.5	57.3	65.6	61.2	61.1	64.5	62.7	63.5
Joint learning	65.2	75.5	70.0	62.6	68.3	65.3	64.8	67.7	66.2	67.1
[Chen et al., 2021b]	62.7	71.1	66.6	58.5	65.7	61.6	61.4	67.2	64.2	64.2

Table 7.2: Resolving AZPs and non-AZPs together.

7.4.2 AZP resolution

Next, we compare the AZP resolution results. For the *pipeline*, we used two settings to represent clusters. First, we used the first mention in the cluster to be concatenated with the AZP representation. Second, we used the last-added mention. Consider the following example:

> Input: Bush met Putin at a nearby airport. He discussed several topics with him.
> Clusters: {Bush, He}

In the example, we can represent the George Bush cluster with Bush or He. We found that representing clusters with the last mention provided better results.

The *pipeline* approach achieves an F1 score of 58.08% when using the first mention as the cluster representation and 58.59% when using the last mention. The *joint learning* provided better results with an F1 score of 59.33%. Chen et al.'s [2021b] model resolved more AZPs correctly than the *pipeline* and *joint learning* methods, achieving an F1 score of 59.49% which is 0.19% more than the *joint learning* score. It seems the two components of Chen et al.'s [2021b] model, the Unit Score and Pairwise Score, are able to distinguish AZPs and mentions effectively for the AZP resolution; however, for coreference resolution, they have showed the performance is better when not considering AZPs.

Training Settings	Test Evaluation		
	P	R	F1
Pipeline (CR: first mention)	60.34	55.98	58.08
Pipeline (CR: last mention)	60.97	56.39	58.59
Joint Learning	61.41	57.40	59.33
[Chen et al., 2021b]	61.67	57.45	59.49

Table 7.3: AZP resolution results of *pipeline*, *joint learning* and Chen et al. [2021b].

7.5 Discussion

The main difference between our *joint learning* approach and Chen et al. [2021b] is how AZPs are detected and learned. In our approach, we present AZPs explicitly and we cluster them with other mentions, while Chen et al.'s [2021b] model learns classifying AZPs and then clustering them with mentions in an end-to-end system. Our results appear to confirm earlier results that considering AZP identification with end-to-end in the coreference resolution task (jointly with overt mentions) can negatively affect the performance on the task [Chen et al., 2021b, Iida and Poesio, 2011] One possible explanation might be the overall performance of the mention detection on non-AZPs is better than AZPs [Chen et al., 2021b]. Chen et al. [2021b] consider every gap as a candidate AZP, which increases the space of possible candidates and affects their detection recall. To mitigate this problem, we use a different neural component for AZP detection. The AZP identification that we used in both the *joint learning* and *pipeline* settings only considers gaps that appear after verbs which limits the number of candidates. Moreover, the AZPs in the *joint learning* have explicit tags which might have resulted in their correct detection, which could be why the approach achieved better results. The main limitation of the our proposed approaches is if the AZP identification fails to detect many AZPs in the test phase, it might have dropped the evaluation of the coreference resolution and AZP resolution dramatically.

In our experiments, we applied the BERT-base architecture of AraBERT [Antoun et al., 2020]. We suspect that using BERT-large might improve the results even more. In addition, pre-training BERT with annotated AZPs can be beneficial. Existing language models (LMs) learn by masking words or perturbing their order [Qiu et al., 2020], but this is not applicable to AZPs. Konno et al. [2021] have shown two approaches to improve language models so they work for AZPs, first by introducing a new pre-training task and second by a new fine-tuning technique. They showed an increased performance for AZP resolution for Japanese. In future works, we intend to pre-train a large-scale language model using their methods and see if it can boost the performance.

7.6 Summary

In this Chapter, we discussed two ways to resolve AZPs and non-AZPs together. The first approach is in a *pipeline* setting, and the second in a *joint learning* representation. The two methods employ the proposed models in Chapters 3, 4 and 5. The *joint learning* outperformed the *pipeline* and another approach [Chen et al., 2021b] in the joint coreference resolution. We also discussed the challenges of these models and how we can improve the task.

Chapter 8

Conclusion

8.1 Summary of our Contributions

Coreference resolution is a difficult task, and even more so for languages that do not have as many resources as English does. This research was an attempt to improving performance on coreference resolution for Arabic, which has so far lagged behind. In this work we made several useful contributions to Arabic coreference resolution:

In Chapter 3, we present the first neural coreference resolver for Arabic. Evaluated on CoNLL-2012, it surpassed the previous state-of-the-art for Arabic coreference with a substantial F1 increase of 15%. We also explained the current challenges and how we can advance in the field.

In Chapter 4, we presented a multi-lingual approach for detecting anaphoric zero-pronouns (AZPs) using BERT-base. We were the first to apply BERT-base for AZP identification in Arabic texts, and our model also achieved state-of-the-art results on Chinese.

In Chapter 5, we proposed a cross-lingual method to resolve AZPs with their true (gold) antecedents. We showed that language models (e.g. BERT) learn about AZPs implicitly, and encode this information within their layers.

We discussed the challenges of creating large datasets for relatively rare phenomena such as AZPs in Chapter 6. We introduced five data augmentation methods that can generate samples based on open texts (e.g. Wikipedia). We apply the augmented dataset on AZP identification and resolution tasks.

In Chapter 7, we tackled the problem of learning AZPs and non-AZPs coreference resolution jointly. We proposed two methods. We showed that a joint-learning model with separate AZP identification outperforms a pipeline model and a pure end-to-end model.

8.2 Revisiting our Research Questions

We have dedicated one or more Chapters to each of the four research questions introduced at the beginning:

RQ1: Is end-to-end coreference resolution feasible with the corpora of more limited size that we have for Arabic?

In Chapter 3, we adapted the end-to-end system of Lee et al.'s [2018] with multilingual BERT [Devlin et al., 2018] to use on the Arabic portion of CoNLL-2012 [Pradhan et al., 2012]. We achieved results better than previous proposals and improved the results even more by using AraBERT [Antoun et al., 2020], a state-of-the-art mention detector [Yu et al., 2020], pre-processing Arabic text and applying an annealing training strategy. We achieved an average F1 score 63.9% while English and Chinese achieved 77.1% and 67.6% respectively using similar settings (i.e, using Lee et al.'s [2018] model and BERT-base). This is reasonable given that Arabic portion of OntoNotes is smaller than the English and Chinese ones [Pradhan et al., 2012].

RQ2: Can neural network models learn how to identify and resolve Arabic Anaphoric zero-pronouns (AZPs)? And do language models encode AZP information in their layers?

In Chapters 4 and 5, we proposed two neural models for AZP identification and resolution. As far as we know, we were the first to work on Arabic AZPs in OntoNotes 5.0 [Weischedel et al., 2011]. Our neural models are also applicable to other languages, as we showed by evaluating them on the Chinese section of OntoNotes, achieving state-of-the-art results.

We conducted more experiments on BERT layers and we found that BERT implicitly encode information about AZPs in their contexts. Some layers encode more information than others and provide better results for AZP tasks.

RQ3: Can we automatically generate data for AZPs without expert annotators?

While there are several datasets have been annotated for pronominal anaphora in Arabic, none has considered annotating AZPs. As far as we know, OntoNotes is the only publically available corpus that has annotated Arabic AZPs, and their total number is relatively small.

We showed five methods for automatically generating AZP samples using Wikipedia articles. Our augmented data helped to boost the performance of existing AZP models. We have also shown possible ways to extend the augmented dataset so it works for coreference resolution and pronominal anaphora.

RQ4: (Realized) Mentions and AZPs are different in nature: mentions have surface realizations while AZPs not (i.e. they are null arguments). Is it possible to cluster these two types of anaphoric expressions in one learning framework?

There have been a couple of proposals for joint AZP and non-AZPs resolution in Chinese, but none for Arabic. In Chapter 7 we showed that it is possible to cluster the two

types of mentions in one framework. We introduced two methods for resolving AZPs and non-AZPs using BERT and neural networks, in a *pipeline* and *joint learning* settings. One of our methods surpasses a previous proposal [Chen et al., 2021b] when evaluated on the Arabic portion of OntoNotes.

8.3 Future Directions

This work has addressed several challenges for coreference resolution in modern standard Arabic (MSA). We also indicated possible future directions for each of these challenges. We hope our work sets the stage for a larger research effort in closely related areas such as coreference resolution for Arabic dialects. Arabic dialects are based on MSA; however, they differ in many grammatical and lexical aspects. For example, the Saudi dialect is similar to MSA and shares many characteristics, but, Moroccan is very different (e.g. darejah). Investigating coreference resolution for these dialects would also have practical applications because they are widely used in social networks and on a daily basis. There are other Semitic and morphologically rich languages that have not been investigated, such as, Hebrew and Amharic. These languages might encounter the same obstacles and can benefit from our suggested solutions.

There are various applications for coreference resolution. It has been shown that coreference resolution is practical for real-time conversations in dialogue processing [Xu and Choi, 2022]. Another application is machine translation where coreference chains can help in translating nouns and pronouns correctly [Werlen and Popescu-Belis, 2017]. Coreference resolution has also been shown to be beneficial to other NLP areas [Krishna et al., 2017, Meged et al., 2020, Pakray et al., 2011, Steinberger et al., 2007]. One of our future plans is to add our coreference resolution and AZP systems to NLP tools, such as, CAMeL [Obeid et al., 2020] or Stanza [Qi et al., 2020]. CAMeL and Stanza are both widely used tools for Arabic NLP and adding our proposed models can help to combine them easily with other NLP areas.

Bibliography

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–16, 2016.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. 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 7088–7105, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.551. URL `https://aclanthology.org/2021.acl-long.551`.

Betty Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. How does bert answer questions? a layer-wise analysis of transformer representations. In *arXiv preprint arXiv:1908.08593*, 2019.

Mahmoud Al-Ayyoub, Aya Nuseir, Kholoud Alsmearat, Yaser Jararweh, and Brij Gupta. Deep learning for arabic nlp: A survey. *Journal of computational science*, 26:522–531, 2018.

Mahmoud Al-Ayyoub, Abed Allah Khamaiseh, Yaser Jararweh, and Mohammed N Al-Kabi. A comprehensive survey of arabic sentiment analysis. *Information processing & management*, 56(2):320–342, 2019.

Wateen Aliady, Abdulrahman Aloraini, Chris Madge, Juntao Yu, Richard Bartle, and Massimo Poesio. Coreference annotation of an arabic corpus using a virtual world game. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, 2022.

Manar Alkhatib, Azza Abdel Monem, and Khaled Shaalan. Deep learning for arabic error detection and correction. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(5):1–13, 2020.

Bashir M. Alnajadat. Pro-drop in standard arabic. In *International Journal of English Linguistics 7.1*, 2017.

Abdulrahman Aloraini and Massimo Poesio. Anaphoric zero pronoun identification: A multilingual approach. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, 2020a.

Abdulrahman Aloraini and Massimo Poesio. Cross-lingual zero pronoun resolution. In *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020b.

Abdulrahman Aloraini and Massimo Poesio. Data augmentation methods for anaphoric zero pronouns. *Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference*, 2021.

Abdulrahman Aloraini, Juntao Yu, and Massimo Poesio. Neural coreference resolution for arabic. *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, 2020.

Abdulrahman Aloraini, Sameer Pradhan, and Massimo Poesio. Joint coreference resolution for zeros and non-zeros in arabic. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, 2022.

Fahad Alotaiby, Ibrahim Alkharashi, and Salah Foda. Processing large arabic text corpora: Preliminary analysis and results. In *Proceedings of the second international conference on Arabic language resources and tools*, pages 78–82. Citeseer, 2009.

Mansour Ibrahim Altamimi. Arabic pro-drop. In *Eastern Michigan University*, 2015.

Maha Althobaiti, Udo Kruschwitz, and Massimo Poesio. Aranlp: A java-based library for the processing of arabic text. *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*, 2014a.

Maha Althobaiti, Udo Kruschwitz, and Massimo Poesio. Automatic creation of arabic named entity annotated corpus using wikipedia. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 106–115, 2014b.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7383–7390, 2020.

Jacob Andreas. Good-enough compositional data augmentation. *arXiv preprint arXiv:1904.09545*, 2019.

Wissam Antoun, Fady Baly, and Hazem Hajj. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*, 2020.

Chinatsu Aone and Scott William Bennett. Evaluating automated and manual acquisition of anaphora resolution strategies. In *ACL '95 Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 122–129, 1995.

Mohammed A Attia. *Handling Arabic morphological and syntactic ambiguity within the LFG framework with a view to machine translation*. The University of Manchester (United Kingdom), 2008.

Hitham M Abo Bakr, Khaled Shaalan, and Ibrahim Ziedan. A statistical method for detecting the arabic empty category. In *Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt*, 2009.

Yassine Benajiba and Imed Zitouni. Arabic mention detection: toward better unit of analysis. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 709–712, 2010.

Yassine Benajiba, Mona Diab, and Paolo Rosso. Arabic named entity recognition using optimized feature sets. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 284–293, 2008.

Rajae Bensoltane and Taher Zaki. Comparing word embedding models for arabic aspect category detection using a deep learning-based approach. In *E3S Web of Conferences*, volume 297. EDP Sciences, 2021.

Majdi Beseiso and Abdulkareem Al-Alwani. A coreference resolution approach using morphological features in arabic. *International Journal of Advanced Computer Science and Applications*, 7(10):107–113, 2016.

Thomas G Bever and Montserrat Sanz. Empty categories access their antecedents during comprehension: Unaccusatives in spanish. *Linguistic Inquiry*, pages 69–91, 1997.

Anders Björkelund and Pierre Nugues. Exploring lexicalized features for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 45–50, 2011.

Anders Björkelund and Jonas Kuhn. Learning structured perceptrons for coreference resolution with latent antecedents and non-local features. In *Proceedings of the 52nd Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 47–57, 2014.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Kaj Bostrom and Greg Durrett. Byte pair encoding is suboptimal for language model pretraining. *arXiv preprint arXiv:2004.03720*, 2020.

Saoussen Mathlouthi Bouzid, Fériel Ben Fraj Trabelsi, and Chiraz Ben Othmane Zribi. How to combine salience factors for arabic pronoun anaphora resolution. In *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, pages 929–936. IEEE, 2017.

Susan E Brennan, Marilyn W Friedman, and Carl Pollard. A centering approach to pronouns. In *25th Annual Meeting of the Association for Computational Linguistics*, pages 155–162, 1987.

Donna K. Byron, Whitney Gegg-Harrison, and Sun-Hee Lee. Resolving zero anaphors and pronouns in korean. In *Traitement Automatique des Langues 46.1*, pages 91–114, 2006.

Tao Chang, Shaohe Lv, Xiaodong Wang, and Dong Wang. Zero pronoun identification in chinese language with deep neural networks. In *2017 2nd International Conference on Control, Automation and Artificial Intelligence (CAAI 2017)*. Atlantis Press, 2017.

Chen Chen and Vincent Ng. Combining the best of two worlds: A hybrid approach to multilingual coreference resolution. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 56–63, 2012.

Chen Chen and Vincent Ng. Chinese zero pronoun resolution: Some recent advances. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1360–1365, 2013.

Chen Chen and Vincent Ng. Chinese zero pronoun resolution: An unsupervised probabilistic model rivaling supervised resolvers. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014.

Chen Chen and Vincent Ng. Chinese zero pronoun resolution: A joint unsupervised discourse-aware model rivaling state-of-the-art resolvers. 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 320–326, 2015.

Chen Chen and Vincent Ng. Chinese zero pronoun resolution with deep neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788, 2016.

Hannah Chen, Yangfeng Ji, and David Evans. Finding friends and flipping frenemies: Automatic paraphrase dataset augmentation using graph theory. *arXiv preprint arXiv:2011.01856*, 2020.

Jiaao Chen, Derek Tam, Colin Raffel, Mohit Bansal, and Diyi Yang. An empirical survey of data augmentation for limited data learning in nlp. *arXiv preprint arXiv:2106.07499*, 2021a.

Shisong Chen, Binbin Gu, Jianfeng Qu, Zhixu Li, An Liu, Lei Zhao, and Zhigang Chen. Tackling zero pronoun resolution and non-zero coreference resolution jointly. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 518–527, 2021b.

Kevin Clark and Christopher D. Manning. Entity-centric coreference resolution with model stacking. In *Association for Computational Linguistics (ACL)*, 2015.

Kevin Clark and Christopher D. Manning. Deep reinforcement learning for mention-ranking coreference models. In *Empirical Methods on Natural Language Processing (EMNLP)*, 2016a.

Kevin Clark and Christopher D. Manning. Improving coreference resolution by learning entity-level distributed representations. In *Association for Computational Linguistics (ACL)*, 2016b.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

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. *arXiv preprint arXiv:1911.02116*, 2019.

Susan Converse. Pronominal anaphora resolution in chinese. In *PhD Thesis, University of Pennsylvania*, 2006.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514, 2021.

Abdelghani Dahou, Shengwu Xiong, Junwei Zhou, and Mohamed Abd Elaziz. Multi-channel embedding convolutional neural network model for arabic sentiment classification. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 18(4):1–23, 2019.

Kareem Darwish and Hamdy Mubarak. Farasa: A new fast and accurate arabic word segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1070–1074, 2016.

H. Daume and D. Marcu. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proc. HLT/EMNLP*, Vancouver, 2005.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *arXiv preprint arXiv:1810.04805*, 2018.

B. Di Eugenio. Centering theory and the italian pronominal system. In *Proc. of the 13th COLING*, Helsinki, Finland, 1990.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. Daga: Data augmentation with a generation approach for low-resource tagging tasks. *arXiv preprint arXiv:2011.01549*, 2020.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, pages 837–840. Lisbon, 2004.

Timothy Dozat and Christopher Manning. Deep biaffine attention for neural dependency parsing. In *Proceedings of 5th International Conference on Learning Representations (ICLR)*, 2017.

Mushira Eid. On the communicative function of subject pronouns in arabic. In *Journal of Linguistics 19.2*, pages 287–303, 1983.

Ibrahim Abu El-Khair. Abu el-khair corpus: A modern standard arabic corpus. *International Journal of Recent Trends in Engineering & Research (IJRTER) Volume*, 2, 2017.

Mohamed Elmahdy, Rainer Gruhn, Wolfgang Minker, and Slim Abdennadher. Modern standard arabic based multilingual approach for dialectal arabic speech recognition. In *2009 Eighth International Symposium on Natural Language Processing*, pages 169–174. IEEE, 2009.

Hady ElSahar and Samhaa R El-Beltagy. Building large arabic multi-domain resources for sentiment analysis. In *International conference on intelligent text processing and computational linguistics*, pages 23–34. Springer, 2015.

Ali Farghaly and Khaled Shaalan. Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8 (4):1–22, 2009.

Hongliang Fei, Xu Li, Dingcheng Li, and Ping Li. End-to-end deep reinforcement learning based coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 660–665, 2019.

Steven Y Feng, Aaron W Li, and Jesse Hoey. Keep calm and switch on! preserving sentiment and fluency in semantic text exchange. *arXiv preprint arXiv:1909.00088*, 2019.

Steven Y Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. Genaug: Data augmentation for finetuning text generators. *arXiv preprint arXiv:2010.01794*, 2020.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.

Eraldo Fernandes, Cícero dos Santos, and Ruy Milidiú. Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 41–48, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

Eraldo R Fernandes and Ruy L Milidiú. Entropy-guided feature generation for structured learning of portuguese dependency parsing. In *International Conference on Computational Processing of the Portuguese Language*, pages 146–156. Springer, 2012.

Eraldo Rezende Fernandes, Cícero Nogueira dos Santos, and Ruy Luiz Milidiú. Latent trees for coreference resolution. *Computational Linguistics*, 40(4):801–835, 2014.

Antonio Ferrández and Jesús Peral. A computational approach to zero-pronouns in spanish. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 166–172, 2000.

Ryan Gabbard. Null element restoration. In *Ph.D Thesis, University of Pennsylvania*, 2010.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, 2019.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010.

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

Madhav Gopal and Girish Nath Jha. Zero pronouns and their resolution in sanskrit texts. In *The International Symposium on Intelligent Systems Technologies and Application*, pages 255–267, 2017.

Spence Green, Conal Sathi, and Christopher Manning. Np subject detection in verb-initial arabic clauses. In *Proceedings of the Third Workshop on Computational Approaches to Arabic Script-based Languages (CAASL3). Vol. 112.*, 2009.

Diana Grigorova. An algorithm for zero pronoun resolution in bulgarian. In *Proceedings of the 14th International Conference on Computer Systems and Technologies*, 2013.

Diana Grigorova. Hybrid approach to zero pronoun resolution in bulgarian. In *Proceedings of the 17th International Conference on Computer Systems and Technologies 2016*, pages 331–338, 2016.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 252–263, 2019.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. Colorless green recurrent networks dream hierarchically. *arXiv preprint arXiv:1803.11138*, 2018.

Demi Guo, Yoon Kim, and Alexander M Rush. Sequence-level mixed sample data augmentation. *arXiv preprint arXiv:2011.09039*, 2020.

Jin Guo. Critical tokenization and its properties. *Computational Linguistics*, 23(4): 569–596, 1997.

Nizar Habash and Fatiha Sadat. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52, 2006.

Nizar Y Habash. Introduction to arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187, 2010.

Souha Hammami, Lamia Belguith, and Abdelmajid Ben Hamadou. Arabic anaphora resolution: Corpora annotation with coreferential links. *International Arab Journal of Information Technology (IAJIT)*, 6(5), 2009.

Na-Rae Han. A korean null pronouns: Classification and annotation. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation. Association for Computational Linguistics, 2004.*, pages 33–40, 2004.

Masatsugu Hangyo, Daisuke Kawahara, and Sadao Kurohashi. Japanese zero reference resolution considering exophora and author/reader mentions. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 924–934, 2013.

Salima Harrat, Karima Meftouh, and Kamel Smaïli. Maghrebi arabic dialect processing: an overview. *Journal of International Science and General Applications*, 2018.

John Hewitt and Christopher 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)*, 2019.

Jerry Hobbs. Resolving pronoun references. In *Lingua*, pages 311–338, 1978.

C.-T. James Huang. On the distribution and reference of empty pronouns. In *Linguistic Inquiry, Vol. 15, No. 4*, pages 531–574, 1984.

Ryu Iida and Massimo Poesio. A cross-lingual ilp solution to zero anaphora resolution. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 804–813, 2011.

Ryu Iida, Kentaro Inui, and Yuji Matsumoto. Exploiting syntactic patterns as clues in zero-anaphora resolution. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistic*, pages 625–632, 2006.

Ryu Iida, Kentaro Inui, and Yuji Matsumoto. Zero-anaphora resolution by learning rich syntactic pattern features. In *ACM Transactions on Asian Language Information Processing, 6(4)*, 2007.

Ryu Iida, Kentaro Torisawa, Chikara Hashimoto, Jong-Hoon Oh, and Julien Kloetzer. Intra-sentential zero anaphora resolution using subject sharing recognition. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2179–2189, 2015.

Hideki Isozaki and Tsutomu Hirao. Japanese zero pronoun resolution based on ranking rules and machine learning. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing,*, pages 184–191, 2003.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *57th Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy*, 2019.

Fan Jiang and Trevor Cohn. Incorporating constituent syntax for coreference resolution. *arXiv preprint arXiv:2202.10710*, 2022.

Amit Jindal, Arijit Ghosh Chowdhury, Aniket Didolkar, Di Jin, Ramit Sawhney, and Rajiv Shah. Augmenting nlp models using latent feature interpolations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6931–6936, 2020.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*, 2019a.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. BERT for coreference resolution: Baselines and analysis. 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 5803–5808, Hong Kong, China, November 2019b. Association for Computational Linguistics. doi: 10.18653/v1/D19-1588.

Sangkeun Jung and Changki Lee. Deep neural architecture for recovering dropped pronouns in korean. *ETRI Journal*, 40(2):257–265, 2018.

Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. Data augmentation for visual question answering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 198–202, 2017.

Megumi Kameyama. *Zero Anaphora: The case of Japanese*. PhD thesis, Stanford University, Stanford, CA, 1985.

Dror Kamir, Naama Soreq, and Yoni Neeman. A comprehensive nlp system for modern standard arabic and modern hebrew. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, 2002.

Ben Kantor and Amir Globerson. Coreference resolution with entity equalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 673–677, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1066.

Sopan Khosla, Juntao Yu, Ramesh Manuvinakurike, Vincent Ng, Massimo Poesio, Michael Strube, and Carolyn Rosé. The codi-crac 2021 shared task on anaphora, bridging, and discourse deixis in dialogue. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 1–15, 2021.

Yeun-Bae Kim and Terumasa Ehara. Zero-subject resolution method based on probabilistic inference with evaluation function. In *Proceedings of the 3rd Natural Language Processing Pacific- Rim Symposium*, pages 721—727, 1995.

YOUNG-JOO Kim. Subject/object drop in the acquisition of korean: A cross-linguistic comparison. In *Journal of East Asian Linguistics 9.4*, pages 325–351, 2000.

Nikita Kitaev, Steven Cao, and Dan Klein. Multilingual constituency parsing with self-attention and pre-training. *arXiv preprint arXiv:1812.11760*, 2018.

Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*, 2018.

Vid Kocijan, Oana-Maria Camburu, Ana-Maria Cretu, Yordan Yordanov, Phil Blunsom, and Thomas Lukasiewicz. Wikicrem: A large unsupervised corpus for coreference resolution. *arXiv preprint arXiv:1908.08025*, 2019.

Fang Kong and Hwee Tou Ng. Exploiting zero pronouns to improve chinese coreference resolution. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 278–288, 2013.

Fang Kong and Guodong Zhou. A tree kernel-based unified framework for chinese zero anaphora resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics*, pages 882–891, 2010.

Fang Kong, Min Zhang, and Guodong Zhou. Chinese zero pronoun resolution: A chain-to-chain approach. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(1):1–21, 2019.

Ryuto Konno, Yuichiroh Matsubayashi, Shun Kiyono, Hiroki Ouchi, Ryo Takahashi, and Kentaro Inui. An empirical study of contextual data augmentation for japanese zero anaphora resolution. *arXiv preprint arXiv:2011.00948*, 2020.

Ryuto Konno, Shun Kiyono, Yuichiroh Matsubayashi, Hiroki Ouchi, and Kentaro Inui. Pseudo zero pronoun resolution improves zero anaphora resolution. *arXiv preprint arXiv:2104.07425*, 2021.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. Revealing the dark secrets of bert. In *arXiv preprint arXiv:1908.08593*, 2019.

M Hari Krishna, K Rahamathulla, and Ali Akbar. A feature based approach for sentiment analysis using svm and coreference resolution. In *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 397–399. IEEE, 2017.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

Ashutosh Kumar, Satwik Bhattamishra, Manik Bhandari, and Partha Talukdar. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3609–3619, 2019.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*, 2020.

Yuri Kuratov and Mikhail Arkhipov. Adaptation of deep bidirectional multilingual transformers for russian language. *arXiv preprint arXiv:1905.07213*, 2019.

Shalom Lappin and Herbert J Leass. An algorithm for pronominal anaphora resolution. *Computational linguistics*, 20(4):535–561, 1994.

Jey Han Lau, Alexander Clark, and Shalom Lappin. Predicting acceptability judgements with unsupervised language models. 2015.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *CONLL Shared Task '11 Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34, 2011.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916, 2013.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*, 2017.

Kenton Lee, Luheng He, and Luke Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. *arXiv preprint arXiv:1804.05392*, 2018.

Sangah Lee, Hansol Jang, Yunmee Baik, Suzi Park, and Hyopil Shin. Kr-bert: A small-scale korean-specific language model. *arXiv preprint arXiv:2008.03979*, 2020.

Sun-Hee Lee, Donna K Byron, and Seok Bae Jang. Why is zero marking important in korean? In *International Conference on Natural Language Processing*, pages 588–599. Springer, 2005.

Baoli li. Learning to model multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL2012-Shared Task*, 2012.

Charles N. Li and Sandra A. Thompson. Third person pronouns and zero anaphora in chinese discourse. In *Syntax and Semantics*, volume 12: Discourse and Syntax, pages 311–335. Academic Press, 1979.

Tyne Liang and Dian-Song Wu. Automatic pronominal anaphora resolution in english texts. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 9, Number 1, February 2004: Special Issue on Selected Papers from ROCLING XV*, pages 21–40, 2004.

Ting Liu, Yiming Cui, Qingyu Yin, Weinan Zhang, Shijin Wang, and Guoping Hu. Generating and exploiting large-scale pseudo training data for zero pronoun resolution. In *arXiv preprint arXiv:1606.01603*, 2017.

Samuel Louvan and Bernardo Magnini. Simple is better! lightweight data augmentation for low resource slot filling and intent classification. *arXiv preprint arXiv:2009.03695*, 2020.

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*, volume 27, pages 466–467. Cairo, 2004.

Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing*. MIT press, 1999.

Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Using Large Corpora*, page 273, 1994.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

Sebastian Martschat and Michael Strube. Latent structures for coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:405–418, 2015.

Yehudit Meged, Avi Caciularu, Vered Shwartz, and Ido Dagan. Paraphrasing vs coreferring: Two sides of the same coin. *arXiv preprint arXiv:2004.14979*, 2020.

Claudiu Mihăilă, Iustina Ilisei, , and Diana Inkpen. Zero pronominal anaphora resolution for the romanian language. In *Research Journal on Computer Science and Computer Engineering with Applications, POLIBITS, 42*, 2011.

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, 2013.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Kun Ming. Chinese coreference resolution via bidirectional lstms using word and token level representations. In *2020 16th International Conference on Computational Intelligence and Security (CIS)*, pages 73–76. IEEE, 2020.

Ruslan Mitkov. *Anaphora resolution: the state of the art*. Citeseer, 1999.

Ruslan Mitkov and Paul Schmidt. On the complexity of pronominal anaphora resolution in machine translation. *STUDIES IN FUNCTIONAL AND STRUCTURAL LINGUISTICS*, pages 207–222, 1998.

Ruslan Mitkov, Richard Evans, Constantin Orasan, Catalina Barbu, Lisa Jones, and Violeta Sotirova. Coreference and anaphora: developing annotating tools, annotated resources and annotation strategies. In *Proceedings of the Discourse, Anaphora and Reference Resolution Conference (DAARC2000)*, pages 49–58. Citeseer, 2000.

Mahmoud Nabil, Mohamed Aly, and Amir Atiya. Astd: Arabic sentiment tweets dataset. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2515–2519, 2015.

Vinod Nair and Geoffrey Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

Xuan-Phi Nguyen, Shafiq Joty, Wu Kui, and Ai Ti Aw. Data diversification: A simple strategy for neural machine translation. *arXiv preprint arXiv:1911.01986*, 2019.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. Camel tools: An open source python toolkit for arabic natural language processing. In *Proceedings of the 12th language resources and evaluation conference*, pages 7022–7032, 2020.

Partha Pakray, Snehasis Neogi, Pinaki Bhaskar, Soujanya Poria, Sivaji Bandyopadhyay, and Alexander F Gelbukh. A textual entailment system using anaphora resolution. In *TAC*, 2011.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *LREC*, volume 14, pages 1094–1101, 2014.

Thomas Payne. *Exploring language structure: A student's guide*. Cambridge University Press, 2006.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Matthew Peters, Sebastian Ruder, and Noah Smith. To tune or not to tune? adapting pretrained representations to diverse tasks. In *arXiv preprint arXiv:1903.05987*, 2019.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

M. Poesio, R. Stuckardt, and Y. Versley. *Anaphora Resolution: Algorithms, Resources and Applications*. Springer, Berlin, 2016.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets. In *CLiC-it*, 2019.

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. Association for Computational Linguistics, Association for Computational Linguistics.*, pages 1–40, 2012.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*, 2020.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Altaf Rahman and Vincent Ng. Resolving complex cases of definite pronouns: the winograd schema challenge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 777–789, 2012.

Guillaume Raille, Sandra Djambazovska, and Claudiu Musat. Fast cross-domain data augmentation through neural sentence editing. *arXiv preprint arXiv:2003.10254*, 2020.

Luz Rello and Iustina Ilisei. A rule-based approach to the identification of spanish zero pronouns. In *Proceedings of the Student Research Workshop*, pages 60–65, 2009.

Luz Rello, Gabriela Ferraro, and Iria Gayo. A first approach to the automatic detection of zero subjects and impersonal constructions in portuguese. *Procesamiento del lenguaje natural*, 49:163–170, 2012.

Ryokan Ri, Toshiaki Nakazawa, and Yoshimasa Tsuruoka. Zero-pronoun data augmentation for japanese-to-english translation. *arXiv preprint arXiv:2107.00318*, 2021.

Tom Ritchey. General morphological analysis. In *16th euro conference on operational analysis*, 1998.

Gözde Gül Şahin and Mark Steedman. Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*, 2019.

Ryohei Sasano and Sadao Kurohashi. discriminative approach to japanese zero anaphora resolution with large-scale lexicalized case frames. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 758–766, 2011.

Ryohei Sasano, Daisuke Kawahara, and Sadao Kurohashi. A fully-lexicalized probabilistic model for japanese zero anaphora resolution. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 769–776, 2008.

Ryohei Sasano, Daisuke Kawahara, and Sadao Kurohashi. The effect of corpus size on case frame acquisition for discourse analysis. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 521–529, 2009.

Khadiga M Seddik, Ali Farghaly, and Aly Aly Fahmy. Arabic anaphora resolution: Corpus of the holy qurâ [euro](tm) an annotated with anaphoric information. *International Journal of Computer Applications*, 124(15), 2015.

Kazuhiro Seki, Atsushi Fujii, and Tetsuya Ishikawa. A probabilistic method for analyzing japanese anaphora integrating zero pronoun detection and resolution. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, pages 1–7, 2002.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

Khaled Shaalan, Marwa Magdy, and Aly Fahmy. Analysis and feedback of erroneous arabic verbs. *Natural Language Engineering*, 21(2):271–323, 2015.

Khaled Shaalan, Sanjeera Siddiqui, Manar Alkhatib, and Azza Abdel Monem. Challenges in arabic natural language processing. In *Computational linguistics, speech and image processing for arabic language*, pages 59–83. World Scientific, 2019.

Abdul-Baquee Sharaf and Eric Atwell. Qurana: Corpus of the quran annotated with pronominal anaphora. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 130–137, 2012.

Tomohide Shibata and Sadao Kurohashi. Entity-centric joint modeling of japanese coreference resolution and predicate argument structure analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 579–589, 2018.

Sho Shimazu, Sho Takase, Toshiaki Nakazawa, and Naoaki Okazaki. Evaluation dataset for zero pronoun in japanese to english translation. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3630–3634, 2020.

Jasdeep Singh, Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. Xlda: Cross-lingual data augmentation for natural language inference and question answering. *arXiv preprint arXiv:1905.11471*, 2019.

Karan Singla, Kunal Sachdeva, Srinivas Bangalore, Dipti Misra Sharma, and Diksha Yadav. Reducing the impact of data sparsity in statistical machine translation. In *Pro-*

ceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 51–56, 2014.

Abu Bakr Soliman, Kareem Eissa, and Samhaa R El-Beltagy. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, 117:256–265, 2017.

Wee M. Soon, Daniel C. Y. Lim, and Hwee T. Ng. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4), December 2001.

Fabio Souza, Rodrigo Nogueira, and Roberto Lotufo. Portuguese named entity recognition using bert-crf. *arXiv preprint arXiv:1909.10649*, 2019.

Marcus Stamborg, Dennis Medved, Peter Exner, and Pierre Nugues. Using syntactic dependencies to solve coreferences. In *Joint Conference on EMNLP and CoNLL2012-Shared Task*, 2012.

Josef Steinberger, Massimo Poesio, Mijail A Kabadjov, and Karel Ježek. Two uses of anaphora resolution in summarization. *Information Processing & Management*, 43(6): 1663–1680, 2007.

Murat Tayli and Abdulah I Al-Salamah. Building bilingual microcomputer systems. *Communications of the ACM*, 33(5):495–504, 1990.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*, 2019.

Fériel Ben Fraj Trabelsi, Chiraz Ben Othmane Zribi, and Saoussen Mathlouthi. Arabic anaphora resolution using markov decision process. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 520–532. Springer, 2016.

Reut Tsarfaty, Djamé Seddah, Yoav Goldberg, Sandra Kübler, Yannick Versley, Marie Candito, Jennifer Foster, Ines Rehbein, and Lamia Tounsi. Statistical parsing of morphologically rich languages (spmrl) what, how and whither. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 1–12, 2010.

Olga Uryupina, Alessandro Moschitti, and Massimo Poesio. Bart goes multilingual: the unitn/essex submission to the conll-2012 shared task. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 122–128. Association for Computational Linguistics, 2012.

Vaibhav Vaibhav, Sumeet Singh, Craig Stewart, and Graham Neubig. Improving robustness of machine translation with synthetic noise. *arXiv preprint arXiv:1902.09508*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

Yannick Versley, Simone Paolo Ponzetto, Massimo Poesio, Vladimir Eidelman, Alan Jern, Jason Smith, Xiaofeng Yang, and Alessandro Moschitti. Bart: A modular toolkit for coreference resolution. In *Proceedings of the ACL-08: HLT Demo Session*, pages 9–12, 2008.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. Switchout: an efficient data augmentation algorithm for neural machine translation. *arXiv preprint arXiv:1808.07512*, 2018.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. Mind the gap: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617, 2018.

Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. Ontonotes: A large training corpus for enhanced processing. *Handbook of Natural Language Processing and Machine Translation. Springer*, 3(3):3–4, 2011.

Lesly Miculicich Werlen and Andrei Popescu-Belis. Using coreference links to improve spanish-to-english machine translation. In *Proceedings of the 2nd Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2017)*, pages 30–40, 2017.

Lydia White. The "pro-drop" parameter in adult second language acquisition. *Language learning*, 35(1):47–61, 1985.

Sam Wiseman, Alexander M Rush, Stuart Shieber, and Jason Weston. Learning anaphoricity and antecedent ranking features for coreference resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1416–1426, 2015.

Sam Wiseman, Alexander M Rush, and Stuart M Shieber. Learning global features for coreference resolution. In *Proceedings of the 2016 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 994–1004, 2016.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.622.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. In *arXiv preprint arXiv:1609.08144*, 2016.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. Generalized data augmentation for low-resource translation. *arXiv preprint arXiv:1906.03785*, 2019.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Y Ng, and Dan Jurafsky. Noising and denoising natural language: Diverse backtranslation for grammar correction. 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 619–628, 2018.

Liyan Xu and Jinho D Choi. Online coreference resolution for dialogue processing: Improving mention-linking on real-time conversations. *arXiv preprint arXiv:2205.10670*, 2022.

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, 2005.

Souta Yamashiro, Hitoshi Nishikawa, and Takenobu Tokunaga. Neural japanese zero anaphora resolution using smoothed large-scale case frames with word embedding. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, 2018.

Jingxuan Yang, Si Li, Sheng Gao, and Jun Guo. Corefdpr: A joint model for coreference resolution and dropped pronoun recovery in chinese conversations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:571–581, 2022.

Wei Yang, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. Data augmentation for bert fine-tuning in open-domain question answering. *arXiv preprint arXiv:1904.06652*, 2019.

Ching-Long Yeh and Yi-Chun Chen. Zero anaphora resolution in chinese with shallow parsing. In *Journal of Chinese Language and Computing 17 (1)*, pages 41–56, 2006.

Qingyu Yin, Yu Zhang, Weinan Zhang, and Ting Liu. A deep neural network for chinese zero pronoun resolution. In *arXiv preprint arXiv:1604.05800.*, 2016.

Qingyu Yin, Yu Zhang, Weinan Zhang, and Ting Liu. Chinese zero pronoun resolution with deep memory network. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1309–1318, 2017.

Qingyu Yin, Yu Zhang, Weinan Zhang, Ting Liu, and William Yang Wang. Zero pronoun resolution with attention-based neural network. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 13–23, 2018.

Qingyu Yin, Weinan Zhang, Yu Zhang, and Ting Liu. Chinese zero pronoun resolution: A collaborative filtering-based approach. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(1):1–20, 2019.

Katsumasa Yoshikawa, Masayuki Asahara, and Yuji Matsumoto. Jointly extracting japanese predicate-argument relation with markov logic. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1125–1133, 2011.

Kei Yoshimoto. Identifying zero pronouns in japanese dialogue. In *Coling Budapest 1988 Volume 2: International Conference on Computational Linguistics*, 1988.

Koichiro Yoshino, Shinsuke Mori, and Tatsuya Kawahara. Predicate argument structure analysis using partially annotated corpora. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 957–961, 2013.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. Neural mention detection. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1–10, Marseille, France, May 2020. European Language Resources Association.

Omar F Zaidan and Chris Callison-Burch. Arabic dialect identification. *Computational Linguistics*, 40(1):171–202, 2014.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657, 2015.

Yi Zhang, Tao Ge, Furu Wei, Ming Zhou, and Xu Sun. Sequence-to-sequence pre-training with data augmentation for sentence rewriting. *arXiv preprint arXiv:1909.06002*, 2019.

Shanheng Zhao and Hwee Tou Ng. Identification and resolution of chinese zero pronouns: A machine learning approach. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 541–550, 2007.

Desislava Zhekova and Sandra Kübler. Ubiu: A language-independent system for coreference resolution. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, page 96–99, 2010.

Haichao Zhu, Li Dong, Furu Wei, Bing Qin, and Ting Liu. Transforming wikipedia into augmented data for query-focused summarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2022.

Imed Zitouni, Jeffrey Sorensen, Xiaoqiang Luo, and Radu Florian. The impact of morphological stemming on arabic mention detection and coreference resolution. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 63–70, 2005.