The Thesis Committee for Victoria Anugrah Lestari Certifies
that this is the approved version of the following thesis:

Building Effective Representations for Domain Adaptation in Coreference Resolution

APPROVED BY

SUPERVISING COMMITTEE:

Greg Durrett, Supervisor

Katrin Erk

Building Effective Representations for Domain Adaptation in Coreference Resolution

by

Victoria Anugrah Lestari

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2018

Dedicated to my family.

Acknowledgments

First of all, I would like to thank my supervisor Greg Durrett for his valuable guidance and motivation during the time I am working on this thesis. Thank you for spending time for our weekly discussion, for your help in debugging codes whenever I am stuck in trouble, and finally for your feedback on the writing of this thesis. I would also like to thank my reader Katrin Erk, whose support and advice are especially important. Without them, I could not have finished this work.

Next, I would like to thank my co-supervisees in Greg's reading group for their help and support, especially when using the GPUs.

I would like to thank my family: my mom and dad, as well as my sisters and brother Shirley, Iman, Adeline, Patricia, and Elizabeth, for their continual love and support and for listening to my stories.

I would like to thank all my friends, from Indonesian Student Association (Permias) Austin and IFGF church, for a lot of great times we have, and my friends from UT Austin, for making learning here fun. Also my friends from online writing group, who read the stories I write just for refreshing my mind after studying here and working on this thesis.

I would like to thank my sponsor, the Indonesia Endowment Fund for Education (LPDP), without which I would not have the opportunity to study

at UT Austin.

Last but not least, I would like to thank Jesus, my Lord, Creator, and Savior, Who has given me so much blessings in my life. Without Him I would not be able to complete everything.

Building Effective Representations for Domain Adaptation in Coreference Resolution

Victoria Anugrah Lestari, M.S.Comp.Sci.
The University of Texas at Austin, 2018

Supervisor: Greg Durrett

Over the past few years, research in coreference resolution, one of the core tasks in Natural Language processing, has displayed significant improvement. However, the field of domain adaptation in coreference resolution is yet to be explored; Moosavi and Strube [2017] have shown that the performance of state-of-the-art coreference resolution systems drop when the systems are tested on datasets from different domains.

We modify *e2e-coref* [Lee et al., 2017], a state-of-the-art coreference resolution system, to perform well on new domains by adding sparse linguistic features, incorporating information from Wikipedia, and implementing a domain adversarial network to the system. Our experiments show that each modification improves the precision of the system. We train the model on CoNLL-2012 datasets and test it on several datasets: WikiCoref, the *pt* documents, and the *wb* documents from CoNLL-2012. Our best results gains 0.50, 0.52, and 1.14 F1 improvements over the baselines of the respective test sets.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This work focuses on domain adaptation in coreference resolution. We evaluate how well state-of-the-art coreference resolution systems perform in a domain adaptation setting. In other words, we observe how the systems suffer from *shift* between the domain they are trained on and the domain they are tested on.

We use the end-to-end coreference resolution system (*e2e-coref*) by Lee et al. [2017] as our base model. Our evaluation method emulates that of Moosavi and Strube [2017]. In one configuration, we train the model on CoNLL-2012 datasets and test it on WikiCoref; in two other configurations, we hold out a genre from CoNLL-2012 from the training documents, and then we test the model on the test set of that particular genre.

As experiments conducted by Moosavi and Strube demonstrate that the performance of coreference resolution systems drop when tested on a new domain, we want to investigate whether we can improve them by implementing three things to our model: (1) adding sparse linguistic features, (2) incorporating Wikipedia information, and (3) integrating a domain adversarial network to the model.

1.1 Coreference Resolution

Coreference resolution, which is the process of identifying entities in a text and finding all expressions that refer to the same entities[1], is a crucial task in understanding natural language. Even humans perform this whether they are read articles or talking to each other. Similarly, for machines to perform well in tasks involving natural language understanding, such as text summarization and question answering, they should gain the ability to resolve coreference links first.

However, unlike basic natural language (NLP) tasks, such as part-of-speech (POS) tagging or named entity recognition (NER), which have achieved almost 100 % accuracy and, thus, can be considered as "solved", coreference resolution still has a long way to achieve that number. Results from the best state-of-the-art coreference resolution systems [Lee et al., 2011, Durrett and Klein, 2013, Wiseman et al., 2015, 2016, Clark and Manning, 2016a,b, Martschat and Strube, 2015, Lee et al., 2017, 2018] show that their F1 scores range in 60-70.

1.2 Domain Adaptation

Domain adaption is the process of adapting a model that is trained on a specific domain, the *source domain*, to perform well on another domain, also known as the *target domain*. For example, a model that is trained on newswire

[1] Definition taken from https://nlp.stanford.edu/projects/coref.shtml

domain is expected to yield good results when tested on biomedical domain [Zhao and Ng, 2014]. In NLP, domain adaptation is favorable because some documents that belong to a certain topic are easy to collect, while documents of another topic are hard to amass. The difference between source domain and target domain is called *shift*. The shift between source domain and target domain can be small (the documents from both domains have many things in common) or large (the documents from source domain are very dissimilar from those from target domain). Multiple approaches to domain adaptation include feature augmentation [Daumé, 2007] (extended to neural networks by Kim et al. [2016]) and domain adversarial training for neural networks [Ganin and Lempitsky, 2015, Ganin et al., 2016].

This work implements domain adversarial network by Ganin and Lempitsky for *e2e-coref*. However, while previous works mostly specify two domains only, we extend the number of domains to seven, corresponding to the seven genres in CoNLL-2012.

1.3 Contributions

The main contribution of this work is that we modify a coreference resolution system to adapt to domains that it is not trained on. We add sparse linguistic features, incorporate information from Wikipedia, and implement domain adversarial network to the model. After we conduct experiments, we observe how each approach influences the behavior of the model and what kind of errors are mitigated by these approaches.

1.4 Thesis Outline

The outline of the rest of this document is organized as follows: Chapter 2 discusses the related works that serve as the foundation for this work. Chapter 3 elaborates our work and supporting theories in detail. Chapter 4 discusses experiments and analyzes results. Chapter 5 presents our conclusion and future work.

Chapter 2

Background and Related Work

In this chapter, we discuss the background and related work that serve as the basis for our work. First, we broadly cover prior work on coreference resolution and domain adaptation. Then we explain previous works that have the most significance on our work: the out-of-domain evaluation setup for multiple coreference resolution systems [Moosavi and Strube, 2017] and the coreference resolution system (*e2e-coref*) by Lee et al. [2017].

2.1 Related Work

We start by narrating the history and development of coreference resolution systems as well as the approaches that researchers used in their work. Then we describe several domain adaptation approaches, elaborating further about domain adversarial neural network (DANN) [Ganin and Lempitsky, 2015, Ganin et al., 2016], whose technique we implement in our modification of *e2e-coref*.

2.1.1 The History of Coreference Resolution

The history of coreference resolution can be traced back to the 1960s when heuristic approaches were the only option in coreference research. In the 1990s, with the rise of machine learning techniques, researchers gradually preferred "learning-based" approaches for coreference resolution. Ng [2010] summarized three classes of coreference models, namely, the mention-pair model, the entity-mention model (e.g. Daumé and Marcu [2005]), and ranking models (e.g. Denis and Baldridge [2008]).

Raghunathan et al. [2010] and Lee et al. [2011] from Stanford designed a rule-based system using multi-pass sieves, i.e. hand-crafted deterministic rules. Their approach initially favored high recall in their mention detection mechanism; then, spurious mentions were filtered out through the sieves. This multi-pass sieve coreference resolution system was the best in its time, ranked first in CoNLL-2011's open track and closed track. Afterwards, they improved their algorithm [Lee et al., 2013] through numerous steps of deterministic rules. Other works following this approach include Ratinov and Roth [2012] and Hajishirzi et al. [2013].

In 2013, Durrett and Klein developed a mention-ranking, purely learning-based model, which automatically extracted lexical features from data instead of using hand-crafted heuristics. Interestingly, their model proved to be more effective and less time-consuming than previous ones. Claiming their approach to win "easy victories", they detect mentions and determine whether two mentions are coreferent by looking at surface features, such as mention type, the complete string of a mention, and head word match. Like the multi-pass sieve approach, their system aims for highest recall possible and rejects spurious mentions afterwards. The effectiveness of their method is due to the fact that surface features captured the same phenomena as rule-based one, though implicitly. However, while these features capture syntactic and discourse-level phenomena remarkably well, they fail to capture semantic phenomena like type compatibility.

This learning-based approach spurred a trend in coreference resolution research. Wiseman et al. from Harvard and Clark and Manning from Stanford published five papers between 2015-2016, competing to develop the best-performing state-of-the-art system. Wiseman et al. [2015] focused on anaphoricity and antecedent ranking features; they outperformed their own system by introducing global features [Wiseman et al., 2016]. While Clark and Manning [2015] designed an entity-centric system with model stacking and improved it with deep reinforcement learning [Clark and Manning, 2016a], they admitted to be inspired by Wiseman et al. [2015] to incorporate entity-level distributed representations into their model [Clark and Manning, 2016b].

Lee et al. [2017] took matters into a further extreme by developing an end-to-end coreference resolution model (*e2e-coref*) that did not require a syntactic parser or a manually crafted mention detector. Surprisingly, their system outperformed all aforementioned systems. They applied attention mechanism in finding head words in mention spans. As we use this model as the basis for our work, we will elaborate details in Section 2.3.

Other approaches involve combining entity-mention model and mention-ranking model [Rahman and Ng, 2009], stacking [Björkelund and Farkas, 2012, Clark and Manning, 2015], and using latent structures [Martschat and Strube, 2015].

Some researchers showed that adding features (syntactic, semantic, lexical) improve coreference resolution systems [Bengtson and Roth, 2008, Björkelund and Nugues, 2011, Haghighi and Klein, 2009]. Occasionally, coreference resolution systems are joined with entity linking [Hajishirzi et al., 2013, Durrett and Klein, 2014]. Others demonstrated that incorporating information from external resources, e.g. semantic role labeling, Wikipedia, and WordNet, is beneficial not only for coreference resolution [Ponzetto and Strube, 2006, Kazama and Torisawa, 2007, Rahman and Ng, 2011, Ratinov and Roth, 2012], but also for other closely related NLP tasks, such as named entity recognition [Kazama and Torisawa, 2007] and entity linking [Hachey et al., 2013].

A less explored yet related area is integrating domain adaptation with coreference resolution. Yang et al. [2012], claiming to be the first to develop a domain adaptation algorithm for coreference resolution, used an adaptive ensemble method to train models to learn cross-domain knowledge. Zhao and Ng [2014] used active learning, feature augmentation technique [Daumé, 2007], and target weighting for domain adaptation.

2.1.2 Related Work on Domain Adaptation

Building a machine learning system by training the model on a specific domain may cause it to overfit the training data and perform poorly on a new dataset from a different domain [Kim et al., 2017]. Domain adaptation is intended to solve the problem of overfitting the model to the training domain, as well as allowing the system to adapt to a new domain that lacks labeled data.

Researchers have tackled the topic of domain adaptation with various approaches: Daumé [2007] introduced feature augmentation, which was extended for neural networks by Kim et al. [2016]; Ganin and Lempitsky [2015] proposed a domain adversarial neural network (DANN) by adding a gradient reversal layer to the regular deep feed-forward neural network; this work is further developed by Ganin et al. [2016]. Zhang et al. [2017] demonstrated that aspect-augmented adversarial networks improved the performance of their model for transfer learning.

Our work relies most heavily on the domain adversarial neural network (DANN) [Ganin and Lempitsky, 2015, Ganin et al., 2016]. Defining domain adaptation as "learning a discriminative classifier or other predictor in the presence of a shift between training and test distributions", they want to train the model to learn discriminative and domain-invariant features. Their approach is unsupervised: thus, no labeled data from the target domain are required.

Ganin and Lempitsky's DANN consists of three parts: (1) the feature extractor G_f, which converts the input data $\mathbf{x}$ into feature vectors $\mathbf{f}$; (2) the label predictor G_y, which maps the feature vectors $\mathbf{f}$ to their respective class labels y; and (3) the domain classifier G_d, which incorporates an additional task of predicting domain labels d for the feature vectors $\mathbf{f}$. During training, the model is expected to learn parameters that minimize the prediction loss L_y and maximize the domain loss L_d.

This is where the **gradient reversal layer** comes in. Inserted between G_f and G_d, it acts as an identity transform to the feature vector $\mathbf{f}$ during forward propagation; however, during backpropagation, the gradient reversal layer multiplies the gradient with a negative scalar. Consequently, as the iteration increases, the model is no longer able to predict the domains. Features that are associated with the domains are "drowned out", while features that predict class labels but are unrelated to domains are amplified.

2.1.3 Dataset

This work utilizes the English dataset from the CoNLL-2012 shared task [Pradhan et al., 2012] and the WikiCoref dataset [Ghaddar and Langlais, 2016]. The CoNLL-2012 dataset consists of seven genres (*bc, bn, mz, nw, pt, tc, wb*) and is divided into three sets: training, development, and test. The genres are described as follows:

- Broadcast Conversation (*bc*)

- Broadcast News (*bn*)
- Magazine (*mz*)
- Newswire (*nw*)
- Telephone Conversation (*tc*)
- Weblogs and Newsgroups (*wb*)
- Pivot text/New Testament (*pt*)

The WikiCoref dataset consists of 30 documents, all taken from the English version of Wikipedia, annotated for anaphoric relations following the OntoNotes guideline. The annotations were done first using Stanford CoreNLP tool [Manning et al., 2014] and then manually by humans with a Kappa coefficient of 0.78. Articles in WikiCoref vary in topic, from the biography to Barrack Obama to the description of Harry Potter film series. The length of the articles ranges from less than 1000 words to more than 5000 words.

2.2 Comparing Coreference Resolution Systems

While research in coreference resolution has extended throughout decades and utilizes multiple techniques, the systems are trained and tested on the same dataset. As Moosavi and Strube [2017] observed, state-of-the-art systems mostly used the CoNLL dataset, causing them to overfit to this dataset.

Thus, they argued that it is rather invalid to claim improvements on coreference resolution. Furthermore, they showed that the performance of these systems dropped when tested on a new domain.

2.2.1 Evaluation Setup

For their experiment, Moosavi and Strube collected four state-of-the-art coreference resolution systems:

- *rule-based*: Stanford's rule-based, multi-pass sieve coreference resolution system [Lee et al., 2011].
- *berkeley*: The Berkeley Coreference Resolution System [Durrett and Klein, 2013]. The variant *berkeley-final* is the same as *berkeley*, using the FINAL feature set described in Section 3, while the variant *berkeley-surface* only uses surface features.
- *cort*: The system using latent structures [Martschat and Strube, 2015]. A variant of this system, *cort–lexical*, does not use any lexical feature at all.
- *deep-coref*: Stanford's coreference resolution system that uses deep reinforcement learning [Clark and Manning, 2016a].

  For this system, they provide three configurations: *deep-coref [conll]* chooses the model with the highest CoNLL score; *deep-coref [lea]* chooses the best model based on LEA metrics; *deep-coref*$^-$ does not incorporate WikiCoref words into the dictionary.

Table 2.1: Comparison of in-domain and out-of-domain performance of coreference resolution systems. All systems are trained on the training set of CoNLL-2012. In the in-domain setup, the systems are tested on the test set of CoNLL-2012. In the out-of-domain setup, the systems are tested on WikiCoref. The results are taken from Moosavi and Strube [2017].

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	55.60	51.77	3.83
berkeley	61.24	51.01	10.23
cort	63.37	49.94	13.43
deep-coref [conll]	65.39	52.65	12.74
deep-coref [lea]	65.60	53.14	12.46

Moosavi and Strube designed three evaluation setups, each consisting of in-domain and out-of-domain testing. (1) In the first setup, the models are trained on the training set of CoNLL-2012. The in-domain evaluation for this setup utilizes the test set of CoNLL-2012, while the out-of-domain evaluation utilizes the WikiCoref dataset. (2) In the second setup, the models are trained on the whole CoNLL-2012 training set for the in-domain setting, while the out-of-domain setting excludes *pt* from the initial training set. The test set of this configuration is the CoNLL-2012 *pt* test set. We call this setup *no-pt*. (3) The third setup is similar to the second one; however, *wb* is excluded instead of *pt*. We call this setup *no-wb*. These two genres are selected because *pt* has the highest degree of overlap between mentions in the training set and the test set, while *wb* has a low degree of overlap.[1]

Table 2.1 shows the comparison of the systems' performance in the in-

[1] Actually, *tc* has the lowest degree of mention overlap, but it is not selected because it contains a large number of pronouns.

Table 2.2: Comparison of in-domain and out-of-domain performance of coreference resolution systems in the *no-pt* configuration. In the in-domain setup, the systems are trained on the whole training set of CoNLL-2012. In the out-of-domain setup, the systems are trained on CoNLL-2012 except *pt*. All systems are tested on the *pt* test set of CoNLL-2012. The results are taken from Moosavi and Strube [2017].

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	-	65.01	-
berkeley-surface	69.15	63.01	6.14
berkeley-final	70.71	64.24	6.47
cort	72.56	64.60	7.96
cort–lexical	69.48	64.32	5.16
deep-coref	75.61	66.06	9.55

domain setting and the out-of-domain setting. While all systems suffer drop in F1 scores, the performance of *rule-based* is relatively stable, losing 3.83 points, compared to learning-based systems that suffer more than 10 points[2].

Next, we discuss the performance of the systems when we hold out one genre from the CoNLL-2012 dataset. Table 2.2 displays the performance of the systems in the *no-pt* configuration. While the drop is not as severe as on WikiCoref, all systems obtain significantly lower F1 scores. Table 2.3 shows interesting results, as some systems actually perform better in the out-of-domain setup.

Moosavi and Strube explained that since *pt* has a large number of overlapping mentions, excluding it from the training set causes learning-based

[2] All results that we report are taken from the experiments conducted by Moosavi and Strube [2017]. We do not replicate their experiments, but we calculate the difference between in-domain and out-of-domain results.

Table 2.3: Comparison of in-domain and out-of-domain performance of coreference resolution systems in the *no-wb* configuration. In the in-domain setup, the systems are trained on the whole training set of CoNLL-2012. In the out-of-domain setup, the systems are trained on CoNLL-2012 except *wb*. All systems are tested on the *wb* test set of CoNLL-2012. The results are taken from Moosavi and Strube [2017].

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	-	53.80	-
berkeley-surface	56.37	55.14	1.23
berkeley-final	56.08	57.31	-1.23
cort	59.29	58.87	0.42
cort–lexical	56.83	57.10	-0.27
deep-coref	61.46	57.17	4.29

systems to decrease their performance. On the other hand, *wb* has few overlapping mentions, so including or excluding it to the training set does not greatly affect the performance of the systems.

2.3 End-to-end neural coreference resolution

Our work is based on the end-to-end neural coreference resolution (*e2e-coref*) developed by Lee et al. [2017], which is the state-of-the-art coref system during the time this work was started. The system does not rely on a syntactic parser or hand-engineered mention detected, yet it outperforms the previous state-of-the-art work [Clark and Manning, 2016a] by 1.5 F1 for their single model and 3.1 F1 for their ensemble models. The idea is to consider all spans as potential mentions and then decide which spans are actually mentions by learning from gold mentions.

The task of end-to-end coreference resolution is to determine which spans are actually mentions and link the mentions to potential antecedents. The input is a document D containing T words. Let $N = \frac{T(T+1)}{2}$ be the number of possible spans in D, and let the start and end of span i be represented by $\textsc{start}(i)$ and $\textsc{end}(i)$ respectively. The spans are ordered by $\textsc{start}(i)$; if two spans have the same start index, they are ordered by $\textsc{end}(i)$. It is the system's objective to assign to each span i an antecedent y_i from the set of possible antecedents $\mathcal{Y}(i)$. If a span does not have an antecedent, it is assigned a dummy antecedent ϵ.

2.3.1 Model

The objective of the model is to find the most likely cluster of antecedents given document D. Hence, it learns the probability distribution $P(y_1, ..., y_N|D)$ by calculating the product of multinomials for each span:

$$\begin{aligned} P(y_1, ..., y_N|D) &= \prod_{i=1}^{N} P(y_i|D) && (2.1) \\ &= \prod_{i=1}^{N} \frac{\exp s(i, y_i)}{\sum_{y' \in \mathcal{Y}(i)} \exp s(i, y')} && (2.2) \end{aligned}$$

where $s(i, j)$ is a pairwise score between span i and span j in D. The following three factors determine this pairwise score: (1) whether span i is a mention, (2) whether span j is a mention, and (3) whether j is an antecedent of i:

$$s(i, j) = \begin{cases} 0 & \text{if } j = \epsilon \\ s_{\mathrm{m}}(i) + s_{\mathrm{m}}(j) + s_{\mathrm{a}}(i, j) & \text{if } j \neq \epsilon \end{cases} \qquad (2.3)$$

$s_{\mathrm{m}}(i)$ is the score of span i being a mention and $s_{\mathrm{a}}(i,j)$ is the pairwise score of i and j being coreferent. These scoring functions are computed using feed-forward neural network:

$$s_{\mathrm{m}}(i) = \boldsymbol{w}_{\mathrm{m}} \cdot \text{FFNN}_{\mathrm{m}}(\boldsymbol{g}_i) \tag{2.4}$$

$$s_{\mathrm{a}}(i,j) = \boldsymbol{w}_{\mathrm{a}} \cdot \text{FFNN}_{\mathrm{a}}([\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j, \phi(i,j)]) \tag{2.5}$$

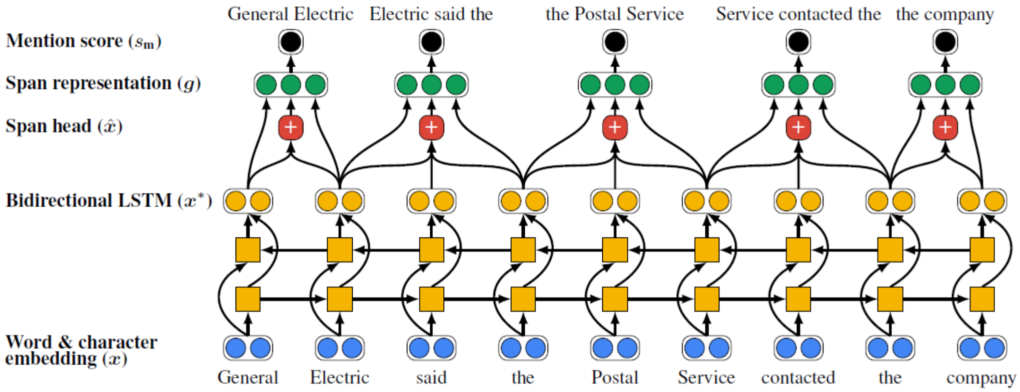
where $\boldsymbol{g}_i$ is the vector representation for span i and $\phi(i,j)$ is a feature vector that encodes speaker and genre information from the metadata and the distance between the two spans.

Using bidirectional LSTMs (biLSTM) [Hochreiter and Schmidhuber, 1997, Gers et al., 2000], the system encodes every word in its context to compute the vector representations for the spans. Then, in contrast to previous systems that incorporated syntactic heads as features [Durrett and Klein, 2013, Clark and Manning, 2016a], *e2e-coref* utilizes attention mechanism over words to learn about headedness [Bahdanau et al., 2014], forming the weighted vector $\hat{\boldsymbol{x}}_i$ that represents a mention span. Each element of the vector denotes the headedness of the mention. Finally, $\hat{\boldsymbol{x}}_i$ is integrated into the final span representation $\boldsymbol{g}_i$:

$$\boldsymbol{g}_i = [\boldsymbol{x}^*_{\text{START}(i)}, \boldsymbol{x}^*_{\text{END}(i)}, \hat{\boldsymbol{x}}_i, \phi(i)] \tag{2.6}$$

Figure 2.1 shows the first step of the *e2e-coref* model. The system uses GloVe and Turian embeddings [Pennington et al., 2014, Turian et al., 2010] to represent words ($\boldsymbol{x}$), which are fed into the biLSTM. After concatenating the

Figure 2.1: The first part of the *e2e-coref* model architecture. The objective of this part is to determine which spans are actually mentions. (This figure is taken from Lee et al. [2017].)



start, end, and head of each span to form $\boldsymbol{g}$, the system computes the score s_{m} to determine whether $\boldsymbol{g}$ is a mention. Low-scoring spans are pruned.

In the second step, the model has identified mentions, and the task is to find the antecedent of a mention or determine if it should start a new cluster. Here, the model computes the antecedent score s_{a} and sum it with the mention score s_{m} to get the final coreference score s. At the final layer, the system calculates the probability distribution $P(y_i|D)$ and applies the softmax function to normalize the values.

The model uses 0.2 dropout rate [Gal and Ghahramani, 2016] for its FFNN and applies Adam optimizer [Kingma and Ba, 2014] for optimization.

2.3.2 Performance

The *e2e-coref* model was trained using the CoNLL-2012 dataset, which contains 2802 training documents, 343 development documents, and 348 test documents. On average, there are 454 words in the training document. The longest training document contains 4009 words.

Lee et al. experimented with feature ablations, including distance and width features, GloVe embeddings, speaker and genre metadata, head-finding attention, character CNN, and Turian embeddings. They reported that removing one of these features resulted in a lower F1 score. The *e2e-coref* model can also be ensembled, which yields a higher F1 score (69.0 against the single model's 67.7). However, since our work only depends on the single model, we do not discuss the ensemble models further.

Chapter 3

Implementation

This chapter discusses the implementation of our work. First, we talk about the performance of *e2e-coref* in an out-of-domain configuration. Next, we explain the technical details, such file preprocessing for the system input and extracting Wikipedia information. Finally, we go over in detail some additional features which we integrate into *e2e-coref* in hopes to improve its performance. Three things we have done are (1) adding part-of-speech tags and named entity tags, (2) incorporating Wikipedia information, and (3) applying domain adversarial neural network (DANN) to the model.

3.1 Initial Experiment

Since *e2e-coref* [Lee et al., 2017] was released after Moosavi and Strube [2017] conducted their evaluation, it is not included in their analysis. Treading in their footsteps, we perform similar experiments to discover how *e2e-coref* ranks against other systems in an out-of-domain context. We use the single model of *e2e-coref* and train it from scratch. Since Lee et al. reported that their model was trained for 400,000 iterations[1], we also train our model for

[1] https://github.com/kentonl/e2e-coref#other-quirks

Table 3.1: Comparison of *e2e-coref* against previous state-of-the-art systems as reported by Moosavi and Strube [2017]. The models are trained on CoNLL-2012 training sets. In-domain results are on the CoNLL-2012 test sets, out-of-domain on WikiCoref. Blue rows indicate our experiments.

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	55.60	51.77	3.83
berkeley	61.24	51.01	10.23
cort	63.37	49.94	13.43
deep-coref [conll]	65.39	52.65	12.74
deep-coref [lea]	65.60	53.14	12.46
e2e-coref (200k)	**66.93**	**50.67**	**16.26**
e2e-coref (400k)	**67.27**	**51.04**	**16.23**

the same amount. Nevertheless, we prepare another model that is trained for 200,000 iterations. Since we have to train several variants of *e2e-coref*, we determine that 200,000 is a reasonable amount of iterations; the model has converged well enough while it takes significantly less time to train.

Table 3.1 shows the performance of *e2e-coref* compared to previous systems. In the CoNLL-2012 test set, our 400k model achieves 67.27 F1 score, similar to the number reported by Lee et al.. On WikiCoref, *e2e-coref*'s F1 score of 51.04 is below *rule-based*, *deep-coref [conll]*, and *deep-coref [lea]*. The performance drop of *e2e-coref* is the greatest among other systems, indicating that *e2e-coref* does not adapt well to a new domain.

Meanwhile, our 200k *e2e-coref* achieves 66.93 F1 in-domain and 50.97 out-of-domain (less 0.34 and 0.47 than the F1 scores of the 400k model respectively). The performance drop is about the same. We decide that these numbers are good enough for future experiments.

Table 3.2: Comparison of *e2e-coref* against previous state-of-the-art systems as reported by Moosavi and Strube [2017]. Top part shows the *no-pt* configuration. The in-domain model is trained on the whole training set of CoNLL-2012; *pt* documents are removed for the out-of-domain model. Bottom part shows the *no-wb* configuration, which is similar to *no-pt* but with *wb* removed. Blue rows indicate our experiments. While *e2e-coref* outperforms previous systems, it also suffers the greatest drop between domains.

no-pt

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	-	65.01	-
berkeley-surface	69.15	63.01	6.14
berkeley-final	70.71	64.24	6.47
cort	72.56	64.60	7.96
cort-lexical	69.48	64.32	5.16
deep-coref	75.61	66.06	9.55
e2e-coref (200k)	**76.68**	**66.16**	**10.52**

no-wb

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	-	53.80	-
berkeley-surface	56.37	55.14	1.23
berkeley-final	56.08	57.31	-1.23
cort	59.29	58.87	0.42
cort-lexical	56.83	57.10	-0.27
deep-coref	61.46	57.17	4.29
e2e-coref (200k)	**62.83**	**59.51**	**3.32**

Next, we replicate Moosavi and Strube's *no-pt* and *no-wb* experiments. As described in Section 2.2.1, the model in the in-domain setting of both configurations is trained on the whole training set of CoNLL-2012. On the other hand, in the out-of-domain context, documents from the *pt* genre are removed from the training set for *no-pt* and from *wb* genre for *no-wb*. For this experiment, we train models for 200,000 iterations only.

Tables 3.2 show the performance of *e2e-coref* compared to previous systems. While *e2e-coref* outperforms other systems in all configurations, it yields the greatest and second greatest decrease in the *no-pt* and *no-wb* configurations respectively. The results confirm that this system does not port well to another domain.

3.2 Preprocessing

The *e2e-coref* system accepts inputs in JSON format. As documents in WikiCoref are provided in XML files, we have to preprocess them into JSON files. We also describe how we extract information from Wikipedia before incorporating it to span representations.

3.2.1 System Input

The *e2e-coref* system extracts speaker information, document number, words in sentences, and coreferent clusters from the `.conll` files. It converts this information into a JSON object and writes each object into a `.jsonlines` file. In other words, a line in a `.jsonlines` file represents a document. For regular training and evaluation, the system requires three files: `train.english.jsonlines` for training, `dev.english.jsonlines` for development, and `test.english.jsonlines` for testing. Each object contains speaker information, document number, list of sentences, and coreference clusters. When speaker information is missing, it is filled with dashes ("-").

Since *e2e-coref* requires both the `.conll` and `.jsonlines` files, we have to convert the WikiCoref documents into CoNLL format (*e2e-coref* can convert `.conll` files into `.jsonlines` files). Using the OntoNotesScheme version, we extract the coreferent clusters and mention spans from the XML files. However, since POS tags and NER tags are not available, we tag them automatically with NLTK[2] [Bird and Loper, 2002]. Speaker information is not provided either; we just fill them with dashes ("-").

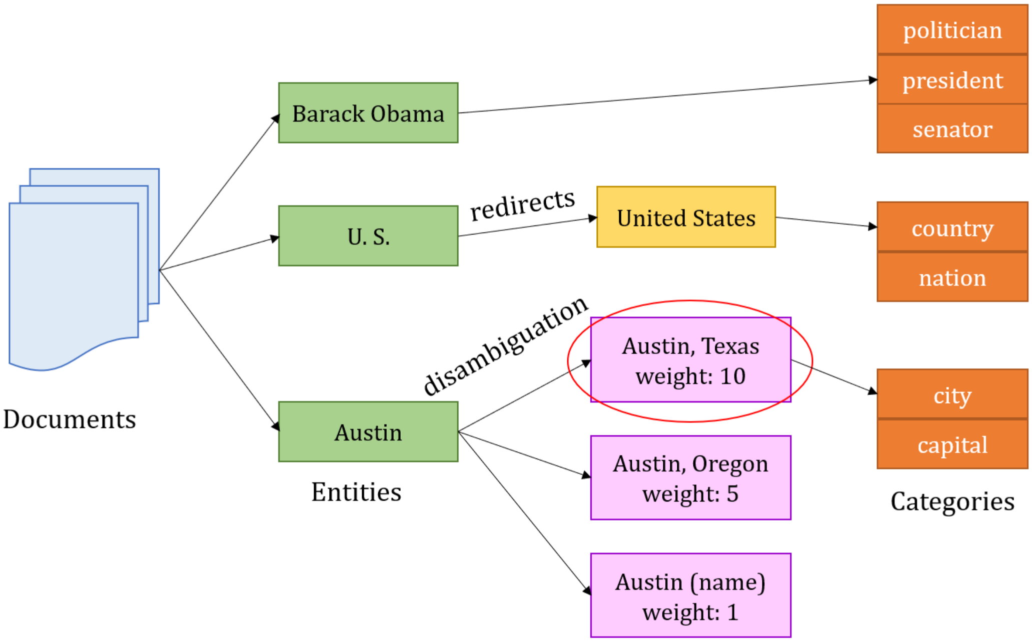
3.2.2 Extracting Information from Wikipedia

To obtain Wikipedia information for entities, we should first perform entity linking, i.e. connecting each entity with its corresponding Wikipedia page, if it has one [Bentivogli et al., 2010]. Entity linking is not an easy task; it has to go through multiple stages, such as entity disambiguation and mapping surface mentions to the exact wording as it appears on Wikipedia Milne and Witten [2008], Ratinov et al. [2011].

While a Wikipedia page contains plentiful information, we only extract

[2] https://www.nltk.org/

Figure 3.1: The process of identifying entities from the dataset and mapping the entities to their corresponding Wikipedia categories.



categories from the page and map each entity to its corresponding categories. For example, *Anatole France* has the following categories: *bibliophile*, *novelist*, *poet*, and *writer*. We design a simple technique to extract information from Wikipedia and assign each mention to the correct categories:

1. We identify all named-entity mentions x from the dataset. We consider all text spans that have NER tags as named-entity mentions. Let X be the set of named-entity mentions retrieved from the dataset.

2. Using the source code from the Berkeley Entity Resolution System[3] [Durrett and Klein, 2014], we generate three JSON files that help link named entities in the CoNLL-2012 dataset to Wikipedia categories.

   (1) `categories.json` consists of actual titles of Wikipedia articles and the categories that those articles belong to. Let $\boldsymbol{c} = [c_1, ..., c_k]$ be a list of categories and C the set of list of categories with $\boldsymbol{c} \in C$. Then `categories.json` contains C.

   (2) `redirects.json` serves as an intermediary to link these entities to their respective categories. Let $\tilde{x}$ be the redirected Wikipedia title for the entity x. Then `redirects.json` contains a list of mappings: $\langle x \rightarrow \tilde{x} \rangle$.

   (3) `target_given_surface.json` provides the probability distributions for disambiguation of entities. Let $(\hat{x}_i, w_i)$ be a pair of disambiguation and weight for the entity x. Then `target_given_surface.json` con-

[3] https://github.com/gregdurrett/berkeley-entity

tains a list of mappings from surface spans to Wikipedia titles and their corresponding probabilities: $\langle x \rightarrow [(\hat{x}_1, w_1)..., (\hat{x}_n, w_n)]\rangle$.

3. For each named-entity mention x extracted from the dataset, find its entry $\tilde{x}$ in the list of categories in `categories.json`. If found, map $\tilde{x}$ to its categories $\mathbf{c}$ directly.

4. If the mention x is not found in `categories.json`, look through the entities in the disambiguation file `target_given_surface.json`. If found, select the entry $\hat{x}_i$ with the highest weight w_i. If not found, just skip the mention.

5. Then, for each entity x that is not yet mapped to categories, look at the list of "redirects" in `redirects.json` and find its entry $\tilde{x}$. If found, return to step 3, i.e. map $\tilde{x}$ to its corresponding categories $\mathbf{c}$. After this stage, the entry should exist in C.

For example, suppose $x =$ *Barack Obama* is identified as a named entity in a CoNLL document. Then we look for this entry in `categories.json`. Since it exists, we map *Barack Obama* to its corresponding categories $c =$ [*politician*, *president*, *senator*].

Now, suppose we look for another entry, $x =$ *Barack Hussein Obama II*. Since it does not exist in `categories.json`, we look for it in `target_given_surface.json`. The result displays two entries: $x \rightarrow [(\hat{x}_1, w_1), (\hat{x}_2, w_2)]$, with the following values:

$\hat{x}_1 =$ *Barack Obama*, $w_1 = 7.0$

$\hat{x}_2 =$ *Barack Hussein Obama II*, $w_2 = 102.0$

The entry *Barack Hussein Obama II* will still be selected as it has the higher weight (102.0 as opposed to 7.0). Now x is updated with the value of $\hat{x}_2$, so $x =$ *Barack Hussein Obama II* (in this case, x is equal $\hat{x}_2$). We go over yet another step, using `redirects.json` to obtain the actual Wikipedia title, $\tilde{x} =$ *Barack Obama*. We update x with $\tilde{x}$ and return to find the categories for the entry *Barack Obama*.

Named entities that are not found in any of these files are skipped and considered not belonging to any category. After filtering out such entities, we discover 10,702 named entities from all documents in CoNLL-2012 and WikiCoref corpora. On average, each entity has 4.2 categories. The maximum number of categories for an entity is 32, belonging to *Winston Churchill*.

Using this technique, we are aware of possible errors in linking entities to categories, such as the following examples:

- *13th District*, a district in Illinois which Barack Obama represented as a senator, is wrongly redirected to *District 13*, a film. The actual *13th District* in Illinois does not have its own Wikipedia page but is tagged as a named entity.

- Selecting the highest weight in `target_given_surface.json` may result in a wrong entity. *Abbas* is wrongly categorized as an actor, even though it actually refers to *Mahmoud Abbas*, a Palestinian leader.

- A random person named *Ali* is wrongly tagged as an Arab ruler. *Ali* is a common name, and the actual person (who is a source in an article in the *wb* genre) does not have a Wikipedia page.

However, as we consider that these errors only comprise a small percentage in the overall corpora, we proceed with the preprocessing results. We have manually inspected 100 entities and their associated categories in 10 documents from *pt*, *wb*, and WikiCoref combined: we discovered that out of 100 entities, about 10 are incorrectly linked.

3.3 Sparse Linguistic Features

As *e2e-coref* does not rely on syntactic and semantic features, we want to know whether adding these features help improve its overall performance, both in an in-domain setting and an out-of-domain setting.

3.3.1 Part-of-speech Tags

The part-of-speech (POS) tag of a word denotes its class, whether it is a noun, verb, adjective, adverb, or numeric. Since the POS tags are provided by the `.conll` files, we treat them as gold POS tags and incorporate them into the `.jsonlines` files, adding a key ”pos_tags”, which is a list of lists containing POS tags corresponding to the words in sentences.

To serve as inputs to the *e2e-coref* model, the list of POS tags is converted into a one-hot vector of length 36 (since the Penn Treebank[4] lists 36 POS tags). Only one entry in this vector will be 1, denoting the POS tag of the word, while the rest are 0's. Then this vector is appended to the end of $\boldsymbol{x}_i$ before it is processed to form $\hat{\boldsymbol{x}}_i$, which is concatenated to $\boldsymbol{g}_i$:

$$\boldsymbol{g}_i = [\boldsymbol{x}^*_{\text{START}(i)}, \boldsymbol{x}^*_{\text{END}(i)}, \hat{\boldsymbol{x}}_i, \phi(i)]$$

3.3.2 Named Entity Recognition

Named entity recognition (NER) is the classification of nouns and proper nouns into one of several categories that describe them. Depending on how coarse or fine-grained the categories are, the number of categories ranges from the most commonly used four (`PERSON, LOCATION, DATE, ORGANIZATION`) to nineteen. Our work uses nineteen categories, which are:

CARDINAL, DATE, EVENT, FAC[5], GPE[6], LANGUAGE, LAW, LOC[7], MONEY, NORP[8], ORDINAL, ORG[9], PERCENT, PERSON, PRODUCT, QUANTITY, TIME, WORK_OF_ART, O[10]

The preprocessing of NER tags is similar to that of POS tags. The `.conll` files contain NER tags that serve as gold standards. However, two key

[4] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

[5] facility

[6] geo-political entity

[7] location

[8] nationality, or religious or political organization

[9] organization

[10] others (not a named entity)

differences are (1) not every word, or even noun, has NER tags, and (2) NER tags may span more than one word.

Incorporating NER tags into feature set. We compare the NER tags of two mentions and see whether there are matching tags. Matches are represented with 1's and differences are represented with 0's. This feature is concatenated to the pairwise mention features $\phi(i, j)$, which forms an input to the FFNN to compute antecedent scores $s_{\mathrm{a}}(i, j)$:

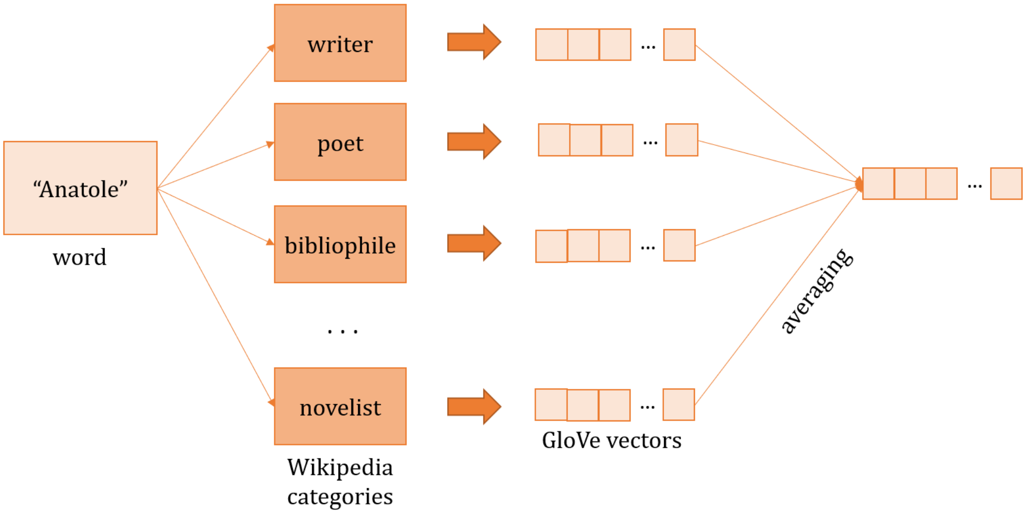
$$[\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j, \phi(i, j)]$$

3.4 Adding Wikipedia Information

We follow the work of Durrett and Klein [2014], who developed a joint model combining coreference, typing, and linking of entities by utilizing the categories section from Wikipedia. They argued that adding external knowledge would help a learning-based system to correct errors in co-referring entities. For example, a system may predict that *Freddie Mac* is a person and link it with a pronoun *his*. However, when information about *Freddie Mac* from Wikipedia is available to the system, it might link *Freddie Mac* to the pronoun *it* instead of *his*.

Since an entity can have more than one category, we decide to use all categories as features and incorporating them into the system input by adding another key in the `.jsonlines` files: `"categories"`, which is a three-dimensional list containing a list of categories. Words that do not have cate-

Figure 3.2: The process of converting Wikipedia categories into vector representations. The categories of *Anatole* are mapped into GloVe vectors; then the vectors are averaged and concatenated into text embeddings input for *e2e-coref*.



gories are assigned dummy categories, i.e. themselves. For example, the word `"novelist"`, which is a noun but not a named entity, is assigned to the category `["novelist"]`. Here we hope that the system can capture the feature *novelist* from one of the categories for *Anatole France* and match it to word *novelist*.

In total, there are over 9,500 categories used by all entities in the corpora; therefore, it is impossible to convert them into one-hot vectors as the size of the input will be too large. As GloVe embeddings help improve the performance of *e2e-coref*, we convert categories into GloVe embeddings and average them before concatenating them to $\boldsymbol{x}_i$, which later forms $\hat{\boldsymbol{x}}_i$ that is concatenated to $\boldsymbol{g}_i$.

Let $\textsc{words}(i) = [z_1, ..., z_n]$ be the list of words that belong to span i and $\textsc{categories}(z_j) = [c_1, ..., c_m]$ be the set of categories that belong to the word z_j. Let n and m be the length of span i and the number of categories for z respectively. Let $\textsc{GloVe}(c_k)$ be a function that returns the word embedding vectors of a word c_k. Then, for word z_j in span i, the vector for categories $\phi_{cat}(z_j)$ is computed as follows:

$$\phi_{cat}(z_j) = \frac{\sum_{k=1}^{m} \textsc{GloVe}(c_k)}{m} \tag{3.1}$$

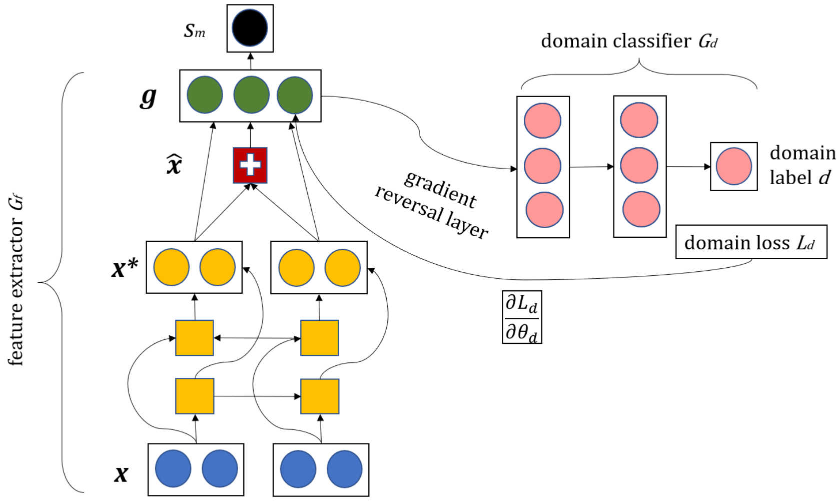
Then, like POS tags, $\phi_{cat}(z_j)$ is appended into $\hat{\boldsymbol{x}}_i$, which is then concatenated into $\boldsymbol{g}_i$. Figure 3.2 illustrates the process of converting Wikipedia categories into GloVe vectors and then averaging them.

3.5 Adding a Domain Adversarial Layer

Another modification we perform to *e2e-coref* is to incorporate domain adversarial neural network (DANN) by Ganin and Lempitsky [2015]. As described in Section 2.1.2, DANN is composed of three parts: the feature extractor G_f, the label predictor G_y, and the domain classifier G_d. While *e2e-coref* already consists of a feature extractor (which converts text input into vector representation) and a label predictor (determining coreference clusters), it does not have a domain classifier.

In the context of coreference resolution, *domain* means the genres of documents. It is no longer binary classification; we extend the number of

Figure 3.3: First implementation of DANN in *e2e-coref*. The domain classifier G_d accepts the span representation $\boldsymbol{g}$ as the input vector.



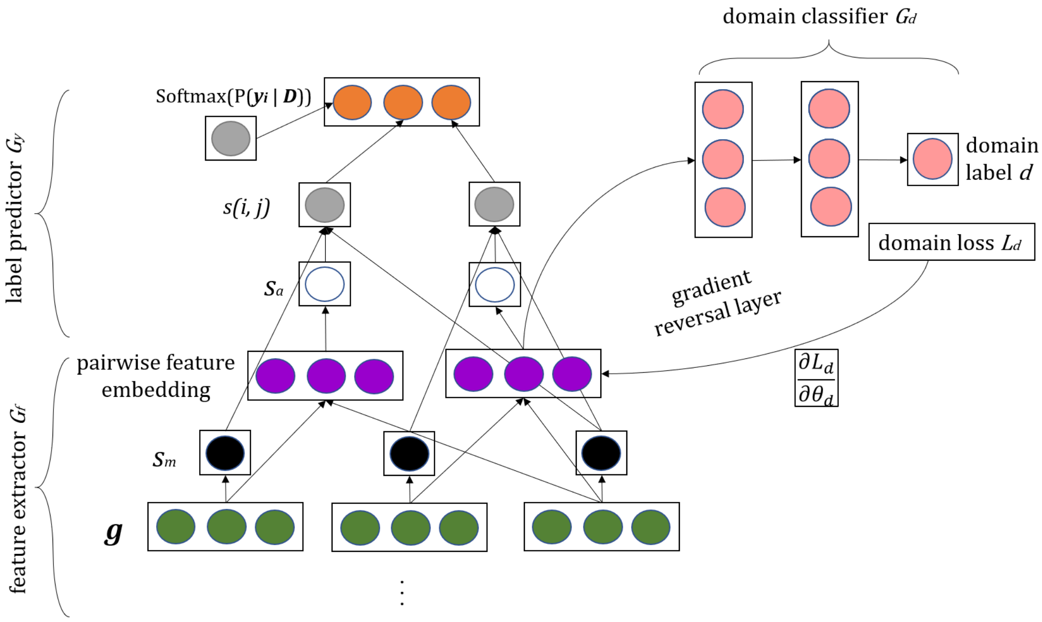
domains to include all seven genres in the CoNLL-2012, plus WikiCoref. We implement DANN to *e2e-coref* by adding **gradient reversal layer**, using the source code that Ganin and Lempitsky provided in their repository.[11]. Our implementation of DANN is a feed-forward neural network with two hidden layers. Before going through the first hidden layer, the input vector passes through the gradient reversal layer

During training, the system learns to find parameters that maximize the domain loss L_d in order to "drown out" features that are specific to the genres. L_d is added to the label predictor loss L_y to get the total loss L, which is then used to compute the gradient for parameter tuning.

We consider two places to insert the gradient reversal layer. In the first

[11] https://github.com/pumpikano/tf-dann/blob/master/flip_gradient.py

Figure 3.4: Second implementation of DANN in *e2e-coref*. The domain classifier G_d accepts the pairwise representation $[\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j, \phi(i, j)]$ as the input vector.



place, we regard the first part of *e2e-coref* as the feature extractor G_f. The domain classifier G_d accepts the span representation $\boldsymbol{g}$ as the input vector. This implementation is illustrated in Figure 3.3.

In the second place, the feature extractor G_f is more expansive, covering the first and second parts of *e2e-coref*. The input vector to the domain classifier G_d is the pairwise representation $[\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j, \phi(i, j)]$. Figure 3.4 depicts the second implementation.

To evaluate the performance of each implementation, we run a few mini experiments in two configurations: (1) We train the model on the training set of CoNLL-2012 excluding *pt* and evaluate it on the unused *pt* training set and development set; (2) the same as the first configuration, but excluding *wb* instead of *pt*. The model is trained for 100,000 iterations and evaluated by the average F1 results. In both configuration (*pt* and *wb*), placing DANN in the second position (accepting pairwise representation as input) yields the higher score. Thus, we proceed with the second implementation for our experiments and analysis.

Chapter 4

Experiments and Results

In this chapter, we discuss experiments and results with the modified *e2e-coref* system. We elaborate the configuration of each experiment and provide possible explanations about the results. Finally, we break down the errors into categories and analyze them.

4.1 Experimental Setup

Following the work of Moosavi and Strube [2017], we train models using three different datasets:

1. **The *original* configuration.** The model is trained on the whole training set of CoNLL-2012 and evaluated on the development set of CoNLL-2012. The model is ultimately tested on the WikiCoref dataset for the out-of-domain setting.

2. **The *no-pt* configuration.** The model is trained on the CoNLL-2012 training set, excluding the *pt* genre. This setting has two development sets: the **in-domain** set from the CoNLL-2012 development set, excluding the *pt* genre, and the **out-of-domain** set constructed from unused

Table 4.1: The number of documents for training sets, two development sets, and test sets of the three configurations.

Configuration	# train	# dev	# dev2	# test
original	2802	343	-	30
no-pt	2483	319	342	26
no-wb	2629	318	198	24

pt documents of the CoNLL-2012 training and development sets. We call the second development set *dev2*. The model is ultimately tested on *pt* documents from CoNLL-2012's test set.

3. **The *no-wb* configuration.** The model is trained on the CoNLL-2012 training set, excluding the *wb* genre. Like the previous configuration, *no-wb* also has two development sets: the in-domain set (*dev*) constructed from the development set of CoNLL-2012 (*wb* genre only) and the out-of-domain set (*dev2*) constructed from unused *wb* documents of the CoNLL-2012 training and development sets. The model is ultimately tested on *wb* documents from the CoNLL-2012's test set.

Table 4.1 shows the numbers of documents in each set for all three configurations. Since the *original* configuration does not have an out-of-domain development set, we evaluate the performance of modified *e2e-coref* mostly on the *no-pt* and *no-wb* configurations.

4.1.1 Baseline

Our baseline is the unmodified version of *e2e-coref*, built from scratch using Lee et al.'s source code. We train three models, one for each configuration, and evaluate them on both in-domain and out-of-domain development sets. However, instead of training the models for 400,000 iterations, we decide to train them for 200,000 iterations only.

4.2 Performance with Sparse Linguistic Features

We incorporate POS tags and NER tags to the model separately at first, and then we combine both of them. We create three models and evaluate their performance both in an in-domain setting and an out-of-domain setting, comparing their performance to the baseline. The model with POS tags features is called +POS, the model with NER tags features is called +NER, and the model with both POS and NER tags features is called +POS+NER. As Lee et al. [2017] did, we judge the performance using three metrics: the MUC, B^3, and $\text{CEAF}_{\phi 4}$ [Luo, 2005].

First, we evaluate the performance of the models with the addition of lexical features on the CoNLL-2012 test set. The baseline result is comparable to that of Lee et al.'s in Table 3.1[1]. While this is still an in-domain evaluation, we observe that the addition of lexical features improve the performance of *e2e-coref*. The combination of POS tags and NER tags slightly boosts the

[1] Our F1 here is 66.93 instead of 67.27 because we run the model for 200,000 iterations instead of 400,000.

Table 4.2: Top part of the table shows the results of *e2e-coref* trained on the whole CoNLL-2012 training set and tested on CoNLL-2012 test set (in-domain). Bottom part shows the same models but tested on WikiCoref (out-of-domain). POS and NER give small boost to the baseline.

CoNLL										
	MUC			B^3			CEAF$_{\phi 4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline (200k)	77.80	73.63	75.66	67.28	61.78	64.61	62.79	58.73	60.69	66.93
+POS	77.47	**74.38**	75.90	67.14	**62.84**	64.92	62.18	**59.78**	60.96	67.26
+NER	77.71	73.56	75.58	67.68	62.18	64.81	62.94	59.10	60.96	67.12
+POS+NER	**78.74**	73.66	**76.11**	**68.80**	61.91	**65.17**	**63.27**	59.46	**61.31**	**67.54**
WikiCoref										
	MUC			B^3			CEAF$_{\phi 4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline (200k)	68.01	**57.66**	62.41	56.30	**42.90**	48.70	45.63	36.76	40.72	50.67
+POS	69.10	57.28	62.64	58.59	42.08	48.98	46.27	**38.21**	**41.86**	51.21
+NER	**69.96**	57.43	**63.08**	**58.66**	42.21	**49.09**	**47.58**	36.76	41.47	**51.28**
+POS+NER	68.72	57.16	62.41	57.03	42.71	48.84	45.54	37.06	40.86	50.78

F1 score by 0.61. Next, we run the models on WikiCoref. The results are slightly different than the previous in-domain results, as +NER outperforms the others by 0.61 increase over the baseline. However, looking at the numbers more closely, we can see that adding lexical features generally increase the performance of *e2e-coref*. Table 4.2 shows the results of both in-domain and out-of-domain experiments.

Table 4.3 displays the results of +POS, +NER, and +POS+NER in the *no-pt* and *no-wb* configurations. We evaluate the models on out-of-domain datasets: the CoNLL-2012 `dev-out`[2] set. In both configurations, +POS outperforms the others by gaining 0.93 F1 increase on *pt* and 1.51 F1 increase on *wb*. While +NER fares the worst, even unable to outperform the baseline

[2] the unused *pt* documents from CoNLL-2012 training and dev sets

Table 4.3: Top part of the table shows the results of *e2e-coref* in the *no-pt* configuration. The models are trained on the CoNLL-2012 except *pt* and tested on *pt*'s out-of-domain development set. Bottom part shows results in the *no-wb* configuration. The models are trained on the CoNLL-2012 except *wb* and tested on *wb*'s out-of-domain development set. +POS wins in both rounds.

no-pt										
	MUC			B^3			$\text{CEAF}_{\phi 4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline	83.49	**74.70**	78.85	65.73	**57.10**	61.11	57.21	53.63	55.36	65.11
+POS	85.24	73.36	**78.86**	**70.05**	55.83	**62.14**	**59.15**	**55.23**	**57.12**	**66.04**
+NER	84.68	71.26	77.39	68.22	54.69	60.71	58.30	52.59	55.30	64.47
+POS+NER	**85.27**	73.33	78.85	69.49	56.16	62.12	59.13	54.39	56.66	65.88
no-wb										
	MUC			B^3			$\text{CEAF}_{\phi 4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline	66.05	**70.30**	68.11	56.40	**59.55**	57.93	51.19	57.08	53.97	60.01
+POS	69.22	69.71	**69.47**	60.06	58.53	**59.28**	54.01	**57.69**	**55.79**	**61.52**
+NER	**70.40**	66.71	68.51	**61.92**	55.53	58.55	**54.10**	55.11	54.60	60.56
+POS+NER	69.45	68.17	68.80	59.94	57.14	58.50	53.85	55.94	54.88	60.73

in *no-pt*, it actually generates a higher precision than the baseline. In fact, in the *no-wb* configuration, the recall of +NER is the highest. Overall, we conclude that adding POS tags and NER tags improves precision but occasionally lowers recall.

4.3 Performance with Wikipedia Information

As previously, we train separate models including features from Wikipedia categories, which we name +Wikipedia, and evaluate their performances by comparing them to the baseline and the best-performing model with lexical features. As each configuration has different best-performing model (+NER for WikiCoref, +POS and +POS+NER for *no-pt*, and +POS for *no-wb*), we also include different models to compare with +Wikipedia.

Table 4.4: Results of *e2e-coref*, plus Wikipedia features, trained on CoNLL-2012 and tested on WikiCoref (out-of-domain). While +Wikipedia does not outperform +NER, it still gains 0.37 F1 over the baseline.

	MUC			B^3			$CEAF_{\phi4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline (200k)	68.01	57.66	62.41	56.30	**42.90**	48.70	45.63	**36.76**	40.72	50.67
+Wikipedia	69.53	**57.72**	**63.08**	57.53	42.82	**49.10**	46.19	36.50	**40.78**	51.04
+NER	**69.96**	57.43	**63.08**	**58.66**	42.21	49.09	**47.58**	36.76	41.47	**51.28**

Table 4.5: Performance of *e2e-coref* plus Wikipedia features. Top part of the table shows results in the *no-pt* configuration. Bottom part shows results in the *no-wb* configuration. +Wikipedia still struggles to beat the baseline, but it generates better precision.

no-pt										
	MUC			B^3			$CEAF_{\phi4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline	83.49	**74.70**	78.85	65.73	**57.10**	61.11	57.21	**53.63**	55.36	65.11
+Wikipedia	84.21	73.27	78.36	66.47	56.22	60.91	57.49	53.04	55.18	64.82
+POS	**85.24**	73.36	**78.86**	**70.05**	55.83	**62.14**	**59.15**	55.23	**57.12**	**66.04**
no-wb										
	MUC			B^3			$CEAF_{\phi4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline	66.05	**70.30**	68.11	56.40	59.55	57.93	51.19	57.08	53.97	60.01
+Wikipedia	67.00	70.23	68.58	57.05	**59.68**	58.34	52.26	56.13	54.13	60.35
+POS	**69.22**	69.71	**69.47**	**60.06**	58.53	**59.28**	**54.01**	**57.69**	**55.79**	**61.52**

First of all, we train the model on all training data of CoNLL-2012. Since this configuration does not have an out-of-domain development set, we only run them on the WikiCoref dataset. Table 4.4 shows the evaluation metrics of this model against the baseline. We can see that the numbers from +Wikipedia consistently exceed the numbers from the baseline, except recall in B^3 and $CEAF_{\phi4}$. While +Wikipedia has not outperformed +NER, we demonstrate that adding Wikipedia information slightly improves the performance of *e2e-coref* on WikiCoref.

Table 4.5 displays the performance of +Wikipedia in the *no-pt* and *no-*

wb configurations. In both cases it fails to top +POS. While Wikipedia features do not improve the F1 scores in *no-pt*, the precision values of +Wikipedia always beat those of the baseline, confirming our statement that Wikipedia features increase precision at the cost of lowering the recall. This is also consistent with the performance of the +Wikipedia model in the original configuration (trained on CoNLL-2012, tested on WikiCoref), as well as the results from Section 4.2, in which sparse linguistic features also improve the performance of *e2e-coref*. On the other hand, +Wikipedia still gives a slight boost on *no-wb*, gaining a 0.34 F1 increase over the baseline.

Our conclusion here is the same as before; like adding lexical features, adding Wikipedia information also helps increase precision. However, the decline in recall is sharp enough to lower F1 scores. Hence, +Wikipedia barely outperforms the baseline.

4.4 Performance with Domain Adversarial Layer

We train *e2e-coref* models with a domain adversarial layer, which we call +DANN, in the same way we did for lexical features and Wikipedia information. As previously, we evaluate the models on three configurations and compare the performances to the baseline and the best-performing models of the previous two evaluations. Since models with lexical features outperform models with Wikipedia information, we only compare +DANN with models with lexical features.

Because of memory issues, we can only train the +DANN models using

Table 4.6: Results of *e2e-coref*, plus domain adversarial layer, trained on CoNLL-2012 and tested on WikiCoref (out-of-domain). +DANN outperforms the baseline and +NER and yields a high recall as well.

	MUC			B^3			$\text{CEAF}_{\phi 4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline (200k)	68.01	57.66	62.41	56.30	42.90	48.70	45.63	36.76	40.72	50.67
+DANN	67.05	**58.86**	62.69	55.61	**44.90**	**49.68**	45.72	**38.40**	**41.74**	**51.42**
+NER	**69.96**	57.43	**63.08**	**58.66**	42.21	49.09	**47.58**	36.76	41.47	51.28

documents with no more than 30 sentences. If a document contains more than 30 sentences, it will be truncated into 30 sentences. However, there is no cut in the development and testing sets; +DANN is evaluated on the same datasets as *e2e-coref* with lexical features and Wikipedia information.

Table 4.6 shows the performance of +DANN trained on all documents of the CoNLL-2012 training set and evaluated on WikiCoref. Interestingly, +DANN outperforms both the baseline and +NER, achieving 0.75 F1 increase over the baseline. While +DANN's precision numbers are lower than either model, it makes up by obtaining high recall, getting a solid 1-2 increase over the baseline on the evaluation metrics.

Nevertheless, the results on WikiCoref are not duplicated on the *no-pt* and *no-wb* configurations, as shown in Table 4.7. While +POS once again outperforms the others, +DANN still manages to come in the second place, gaining 0.2 F1 on *pt* and 0.15 F1 on *wb*.

While +DANN mostly gains slight improvement over the baseline in all configurations, it is hard to draw a conclusion on its performance. It appears to perform best on WikiCoref. However, on *pt* and *wb*, it struggles to beat the

Table 4.7: Performance of *e2e-coref* plus domain adversarial layer. Top part of the table shows results in the *no-pt* configuration. Bottom part shows results in the *no-wb* configuration. +DANN outperforms the baseline but fails to beat +POS.

no-pt										
	MUC			B^3			CEAF$_{\phi4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline	83.49	74.70	78.85	65.73	57.10	61.11	57.21	**53.63**	55.36	65.11
+DANN	84.28	73.31	78.41	68.85	55.21	61.28	57.37	55.10	56.20	65.31
+POS	**85.24**	73.36	**78.86**	**70.05**	55.83	**62.14**	**59.15**	55.23	**57.12**	**66.04**
no-wb										
	MUC			B^3			CEAF$_{\phi4}$			
Configuration	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Baseline	66.05	**70.30**	68.11	56.40	**59.55**	57.93	51.19	57.08	53.97	60.01
+DANN	67.27	70.02	68.62	57.09	59.00	58.03	52.32	55.40	53.82	60.16
+POS	**69.22**	69.71	**69.47**	**60.06**	58.53	**59.28**	**54.01**	**57.69**	**55.79**	**61.52**

baseline. We can only argue that the performance of +DANN can improve *e2e-coref*, but it depends too heavily on the dataset.

4.5 Results on Test Sets

We return to Moosavi and Strube's experiments by choosing our best models for each configurations and comparing their in-domain and out-of-domain performance against previous results. For the *original* configuration, the in-domain test set is the CoNLL-2012 test set. We select +DANN, as it performs best on WikiCoref, and train it for 400,000 iterations to match the unmodified *e2e-coref* model.

Table 4.8 shows the performance of +DANN on WikiCoref as opposed to previous results. Although the in-domain F1 score decreases by 0.83, the out-of-domain F1 score actually increases by 0.50. This slight gain has not

Table 4.8: Comparison of *e2e-coref* and our best modification (+DANN) against previous state-of-the-art systems as reported by Moosavi and Strube [2017]. In-domain results are on the CoNLL-2012 test sets, out-of-domain on WikiCoref. Adding domain adversarial layer improves out-of-domain F1 by 0.5; however, it is still unable to beat *rule-based*.

System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	55.60	51.77	3.83
berkeley	61.24	51.01	10.23
cort	63.37	49.94	13.43
deep-coref [conll]	65.39	52.65	12.74
deep-coref [lea]	65.60	53.14	12.46
e2e-coref (400k)	67.27	51.04	16.23
e2e-coref+DANN (400k)	**66.44**	**51.54**	**14.90**

been able to outperform *rule-based*, but we have shown that applying DANN slightly improves the out-of-domain performance.

For the *no-pt* and *no-wb* configurations, we choose +POS as the contender against previous systems as it generates the best results on the out-of-domain development set. However, +POS in the in-domain setting is trained on the whole CoNLL-2012 training set, while +POS in the out-of-domain setting is trained without *pt* or *wb*. On *pt*, +POS actually improves both in-domain and out-of-domain performance. On *wb*, +POS fares slightly worse in the in-domain setting, but achieves 1.14 F1 score increase in the out-of-domain setting. Detailed results can be seen in Table 4.9.

Compared to Stanford's *rule-based* system [Lee et al., 2011], even the unmodified *e2e-coref* has no problem outperforming it when both training and test sets originate from CoNLL-2012, as shown by the results in both *no-pt*

Table 4.9: Comparison of *e2e-coref* and our best modification (+POS) against previous state-of-the-art systems as reported by Moosavi and Strube [2017]. In-domain results use a model trained on the whole CoNLL-2012 training set. **Top:** In the out-of-domain setting of *no-pt*, *pt* documents are removed from the training set. Adding POS tags improves out-of-domain F1 by 0.52. **Bottom:** In the out-of-domain setting of *no-wb*, *wb* documents are removed from the training set. Adding POS tags improves out-of-domain F1 by 1.14.

no-pt			
System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	-	65.01	-
berkeley-surface	69.15	63.01	6.14
berkeley-final	70.71	64.24	6.47
cort	72.56	64.60	7.96
cort-lexical	69.48	64.32	5.16
deep-coref	75.61	66.06	9.55
e2e-coref (200k)	76.68	66.16	10.52
e2e-coref+POS (200k)	**77.83**	**66.64**	**11.19**

no-wb			
System name	F1 (in-domain)	F1 (out-of-domain)	Drop
rule-based	-	53.80	-
berkeley-surface	56.37	55.14	1.23
berkeley-final	56.08	57.31	-1.23
cort	59.29	58.87	0.42
cort-lexical	56.83	57.10	-0.27
deep-coref	61.46	57.17	4.29
e2e-coref (200k)	62.83	59.51	3.32
e2e-coref+POS (200k)	**62.39**	**60.65**	**1.74**

and *no-wb* configurations. However, on WikiCoref, *e2e-coref*+DANN's 51.54 F1 score is unable to beat *rule-based*'s 51.77, although it is certainly coming closer. While we argue that documents in WikiCoref are much longer than those in CoNLL-2012, and that the topics covered in WikiCoref are a great shift from those in CoNLL-2012 combined together (i.e. WikiCoref is a difficult corpus), we can only conclude that rule-based systems generalize on different domains better than learning-based systems.

4.6 Error Analysis

For the error analysis, we select all four models from the *no-pt* and *no-wb* configurations: (1) baseline, (2) +POS, (3) +Wikipedia, and (4) +DANN. First, we go in detail about the precision and recall of each model, and then we split mentions into categories and analyze the accuracy of antecedent linking for each category.

4.6.1 Precision and Recall

While we have already known the precision and recall of the models in any configuration, we want to observe how the features affect the model in deciding the coreference clustering. Therefore, we compare the coreference clusters predictions generated by the models on the out-of-domain development set (*dev2*)and analyze the error, particularly looking at the documents `pt/nt/40/nt_4001_0` and `wb/a2e/00/a2e_0000_0`.

> `pt/nt/40/nt_4001_0`
>
> This is the family history of Jesus Christ. He came from the family of David and from the family of Abraham. Abraham was the father of Isaac. Isaac was the father of Jacob. Jacob was the father of Judah and his brothers. Judah was the father of Perez and Zerah. -LRB- Their mother was Tamar. -RRB- Perez was the father of Hezron. Hezron was the father of Ram. Ram was the father of Amminadab. Amminadab was the father of Nahshon. ... But after Joseph thought about this, an angel from the Lord came to him in a dream. The angel said, “Joseph, son of David, don’t be afraid to accept Mary to be your wife. The baby inside her is from the Holy Spirit. She will give birth to a son. You will name him Jesus. Give him that name because he will save his people from their sins. ....

The document `pt/nt/40/nt_4001_0` is an excerpt from the New Testament and discusses the genealogy of Jesus Christ. The gold standard specifies 53 clusters. As previously discussed, adding features improves precision but lowers recall, since the baseline model discovers 33 clusters, +POS and +DANN find 23, and +Wikipedia identifies 28. However, this is not always the case. Sometimes both the unmodified and modified versions of *e2e-coref* identify more clusters than there actually exist in the gold standard.

Looking at the mentions in the cluster, we see a very interesting prediction from the baseline model, in which it puts *Jesus Christ*, *Abraham*, *Isaac*, and *Jacob* in the same cluster. Meanwhile, +POS, +Wikipedia, and +DANN are able to create a separate cluster for *Jesus Christ*. However, only +POS captures the span *”Jesus , who is called the Christ”*, which is specified as a mention in the gold standard.

Another interesting behavior is shown when the models (except +DANN) attempt to cluster a mention, *the prophet*. While this span is not classified as a mention by the gold standard, the baseline, +POS, and +Wikipedia consider it as a mention. The baseline model puts *the prophet* in the same cluster as *Jesus Christ*, *Abraham*, *Isaac*, and *Jacob*. On the other hand, +POS includes this mention in the same cluster as *Jesus Christ*, and +Wikipedia includes this mention in the same cluster as *Abraham*, *Isaac*, and *Jacob*.

While in Wikipedia all of them are categorized as prophets, *Jesus* has significantly more categories than *Abraham*, *Isaac*, and *Jacob*. Since the +Wikipedia model averages GloVe embeddings for the categories, it is possible that the prophet category for the entity *Jesus* is drowned out by other categories, thus leading the model to cluster *the prophet* with *Abraham*, *Isaac*, and *Jacob*.

+DANN, nevertheless, makes a mistake by clustering not only *Abraham*, *Isaac*, and *Jacob* together, but also putting all other ancestors of Jesus, such as *Hezekiah*, *Uzziah*, and *Eleazar*.

+DANN also displays a unique behavior not found in the other three while trying to cluster the span *your wife* in the context of Joseph taking Mary to be his wife. While the gold standard does not identify *your wife* as a mention since Mary is not yet Joseph's wife at that time, +DANN puts this phrase into the same cluster with *Mary*.

Features aside, the models seem to have a strong preference for head word match. All four puts mentions with the word *mother* in the same cluster, even though the *mother* refers to different people (Tamar, Rahab, Ruth, Uriah's wife, and Mary). However, we can further argue that this document is atypical; it contains so many people and so few actions or events.

> `wb/a2e/00/a2e_0000_0`
>
> Celebration Shooting Turns Wedding Into a Funeral in Southern Gaza Strip Asad 1/20/2007 Gaza - UPI The cheers and hails of happiness at a wedding in Khan Younes in the southern Gaza Strip turned into screams and moans of pain after one of the celebrators lost control of his weapon, from which a number of bullets were released that killed the groom's brother and hit three other relatives of his, turning the wedding into a funeral in moments. ... Mohamed Al Bashiti, 18 -LRB- years old -RRB-, a relative of the victim, mentioned that while the young men were performing dances and popular dabke dances Thursday evening, and while they were in a state of intense rejoicing for the wedding of our relative, Majed Al Bashiti, one of the armed men lost control of his weapon, which he was trying to use to fire to celebrate and greet the groom. The bullets went astray and hit a number of participants amid a state of panic and terror. ...

Turning to the document `wb/a2e/00/a2e_0000_0`, which is a news article talking about shooting in Gaza Strip, we can see that adding features also lowers recall here. The gold standard specifies 19 clusters, while the baseline and +DANN identify 14, +POS finds 13, and +Wikipedia discovers 12. The models are unable to identify mentions that look too general, such as *the warnings*, and that are foreign-sounding, such as *Nasser Hospital*. Some clusters also require a deeper understanding of the text, as the gold standard puts *one of the armed men* and *one of the celebrators* in the same cluster. On the other hand, the models group *the armed men* and *the young men* instead, creating a mistake in predicting the mention span and then propagating that error into clustering the mentions.

+DANN is the only model that can identify *Majed Al Bashiti, one of the armed men* as a separate mention and linking it to seemingly correct pronouns from the following phrase: "*... while they were in a state of intense rejoicing for the wedding of our relative,* ***Majed Al Bashiti, one of the armed men*** *lost control of* ***his*** *weapon, which* ***he*** *was trying to use to fire to celebrate and greet the groom.*" (Bold indicates +DANN's clustering.) Although the gold standard specifies that *Majed Al Bashiti* is the name of the relative who had the wedding instead of the name of the armed man, we can argue that this appositive might be considered ambiguous.

+Wikipedia shows yet another interesting behavior when it clusters *died* and *the tragic accident* together. While *died* is definitely not a mention, the model apparently understands the word meaning and thus puts them in the same cluster.

4.6.2 Based on Mention Types

We also analyze errors based on the mention types, as Durrett and Klein [2013] did. We break down mentions into several categories denoting their lexical types and characteristics in the clusters. First of all, we distinguish the word class of the mentions: Nominal/Proper, which comprises of regular nouns and proper nouns, and Pronominal, which comprises of pronouns. Secondly, we classify a mention by its role in the coreference clusters: whether it is the first word to refer to a specific entity (Starts Entity) or a subsequent referent (Anaphoric). Finally, we differentiate a mention further by the type of its head word: whether its head word is the first to appear in a mention (1^{st} w/heads) or its head word has appeared in a previous mention, not necessarily in the same cluster (2^{nd}+ w/heads).

A mention is said to be correctly linked to an antecedent if both mention and antecedent belong in the same cluster according to the gold standard. For example, if the system predicts that the mention i has an antecedent j, the link between them is considered correct if both i and j exist in the same cluster in the gold standard. We then compute the accuracy of the *e2e-coref* system as follows:

$$\text{accuracy} = \frac{\#\text{ correct antecedent links}}{\#\text{ total antecedent links}}$$

Table 4.10 shows the percentages of correct antecedent links for all models and all categories. As usual, the models are trained on the CoNLL-2012, except *pt* for *no-pt* configuration, and except *wb* for *no-wb* configuration. Overall, the *e2e-coref* system struggles to resolve anaphoric mentions when the head word first appears as a mention (bottom left) and pronominal mentions when it starts an entity (top right). This result is consistent with Durrett and Klein [2013]'s report; the biggest weakness of the Berkeley Coreference Resolution System is also resolving anaphoric nouns with first head appearances.

We argue two reasons to explain this phenomenon. First, instances in the bottom left and top right categories are the least (0.2-0.4K and 0.4-0.9K respectively). Thus, the models do not see enough data to resolve them. Second, it is unusual for an entity to first appear in a pronominal word, as

Table 4.10: Breakdown of errors for *e2e-coref* based on mention types. Results on *no-pt* are shown on top, *no-wb* at the bottom. The models are evaluated on out-of-domain development sets. In general, the system struggles in resolving coreference for anaphoric head words with first appearances (bottom left) and pronominal words that starts entities (top right). Overall results show +POS once again outperforming others in both *no-pt* and *no-wb* configurations.

no-pt		Nominal/Proper				Pronominal	
		1^{st} w/heads		2^{nd}+ w/heads			
Baseline	Starts Entity	53.75%	1K	**57.88%**	4.7K	20.75%	0.9K
	Anaphoric	**27.03%**	0.2K	65.60%	18.7K	70.87%	14.9K
+POS	Starts Entity	57.66%	0.9K	57.31%	4.7K	**22.90%**	0.9K
	Anaphoric	22.48%	0.2K	**67.90%**	17.9K	**72.79%**	14.6K
+Wikipedia	Starts Entity	55.28%	1K	57.03%	4.6K	20.38%	0.8K
	Anaphoric	21.96%	0.2K	66.24%	18.1K	71.48%	14.7K
+DANN	Starts Entity	**58.66%**	0.9K	57.20%	4.9K	21.25%	0.9K
	Anaphoric	18.14%	0.2K	66.95%	18.3K	71.69%	14.5K
no-wb		Nominal/Proper				Pronominal	
		1^{st} w/heads		2^{nd}+ w/heads			
Baseline	Starts Entity	40.78%	1.6K	47.52%	1.7K	**17.06%**	0.4K
	Anaphoric	22.72%	0.4K	46.77%	5.8K	51.47%	5K
+POS	Starts Entity	**44.73%**	1.6K	**49.63%**	1.6K	16.97%	0.4K
	Anaphoric	**28.57%**	0.3K	**50.70%**	5.4K	**51.61%**	4.9K
+Wikipedia	Starts Entity	40.91%	1.6K	48.84%	1.5K	14.54%	0.4K
	Anaphoric	25.88%	0.4K	48.39%	5.6K	50.60%	5K
+DANN	Starts Entity	42.53%	1.6K	48.38%	1.6K	16.27%	0.4K
	Anaphoric	26.05%	0.4K	49.11%	5.5K	50.77%	5.1K

well as to have its proper name appears later as an anaphor, e.g. introducing *Barack Obama* as “he” before actually mentioning his proper name.

Once again, +POS proves to be the best among other *e2e-coref* variants. One explanation is that most entities are nouns, whether proper nouns, regular nouns, or pronouns, and POS tags provide this information. Previous results demonstrate that +POS models generate high precision. In the second place, +DANN generally has higher accuracy across categories and datasets than +Wikipedia. This result is also consistent with previous results, in which +Wikipedia yields the smallest improvement.

We have several assumptions on why +Wikipedia is the weakest model. First, not every entity has its own Wikipedia page, meaning that they have no Wikipedia features. Sometimes this cause the mention to be redirected to wrong pages. For example, the location *13th District* is incorrectly redirected to the film *District 13*. Second, the entity linking mechanism might generate mistakes since, in the disambiguation process, it always selects the entity with the highest weight. Finally, converting Wikipedia categories into GloVe vectors and averaging them might not be the best method to gain the most information. In order to make Wikipedia features perform well, we have to develop meticulous heuristics to avoid mistakes.

Chapter 5

Conclusion and Future Work

While research in coreference resolution appears to be promising as new systems continually outperform older ones, the results reported in research papers and conferences are mostly based on popular datasets, such as CoNLL-2012. Few researchers heed the problem of overfitting their models to these datasets. As Moosavi and Strube [2017] have demonstrated in their experiments, most state-of-the-art coreference resolution systems do not adapt well to new domains which the systems are not trained on.

In this work, we modify a state-of-the-art system, *e2e-coref* developed by Lee et al. [2017], using three different methodologies: (1) we add sparse linguistic features, i.e. POS tags and NER tags, into text embeddings, (2) we include information from Wikipedia by extracting categories for named entities and converting them into GloVe vectors, and (3) we integrate a domain adversarial neural network [Ganin and Lempitsky, 2015] to the system.

As we have shown in our experiments, all methods mostly improve the precision of the baseline, though occasionally at the cost of recall, resulting in a lower F1. While, at first glance, adding POS tags to the model gives the greatest increase in F1 scores, our error analysis indicates that different features generate different behavior in the models.

We conclude that domain adaptation for coreference resolution is a challenging problem. However, there are a lot of methods that we have not attempted due to limited time and resources. (Training a model for 200,000 iterations consumes 10 GB of GPU memory and takes around 11 hours.)

In the future, we want to combine some of the features, e.g. sparse linguistic features with Wikipedia information, or all three methodologies together, and see which combination yields the best results for domain adaptation. Another idea for future work is to use a different method for GloVe embeddings of Wikipedia categories, such as element-wise multiplication instead of averaging the vectors, or including Turian embeddings.

A new paper that appears while our work is ongoing [Lee et al., 2018] applies deep contextual word representations [Peters et al., 2018] and gain 5.8 F1 score improvement over *e2e-coref* (it is, indeed, *e2e-coref* with that new type of word embeddings). Our future work is to see how well the new system performs in an out-of-domain setting, and whether our methods also helps improving it.

Bibliography

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Eric Bengtson and Dan Roth. Understanding the value of features for coreference resolution. In *EMNLP*, 2008.

Luisa Bentivogli, Pamela Forner, Claudio Giuliano, Alessandro Marchetti, Emanuele Pianta, and Kateryna Tymoshenko. Extending english ace 2005 corpus annotation with ground-truth links to wikipedia. 2010.

Steven Bird and Edward Loper. Nltk: The natural language toolkit. *CoRR*, cs.CL/0205028, 2002.

Anders Björkelund and Richárd Farkas. Data-driven multilingual coreference resolution using resolver stacking. In *Joint Conference on EMNLP and CoNLL - Shared Task*, CoNLL '12, pages 49–55, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2391181.2391185`.

Anders Björkelund and Pierre Nugues. Exploring lexicalized features for coreference resolution. In *CoNLL Shared Task*, 2011.

Kevin Clark and Christopher D. Manning. Entity-centric coreference resolution with model stacking. In *ACL*, 2015.

Kevin Clark and Christopher D. Manning. Deep reinforcement learning for mention-ranking coreference models. In *EMNLP*, 2016a.

Kevin Clark and Christopher D. Manning. Improving coreference resolution by learning entity-level distributed representations. *CoRR*, abs/1606.01323, 2016b.

Hal Daumé. Frustratingly easy domain adaptation. *CoRR*, abs/0907.1815, 2007.

Hal Daumé and Daniel Marcu. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *HLT/EMNLP*, 2005.

Pascal Denis and Jason Baldridge. Specialized models and ranking for coreference resolution. In *EMNLP*, 2008.

Greg Durrett and Dan Klein. Easy victories and uphill battles in coreference resolution. In *EMNLP*, 2013.

Greg Durrett and Dan Klein. A joint model for entity analysis: Coreference, typing, and linking. *TACL*, 2:477–490, 2014.

Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*, 2016.

Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, Franois Laviolette, Mario Marchand, and Victor S. Lempitsky. Domain-adversarial training of neural networks. In *Domain Adaptation in Computer Vision Applications*, 2016.

Felix A. Gers, Jürgen Schmidhuber, and Fred A. Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12 10:2451–71, 2000.

Abbas Ghaddar and Philippe Langlais. Wikicoref: An english coreference-annotated corpus of wikipedia articles. In *LREC*, 2016.

Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. Evaluating entity linking with wikipedia. *Artif. Intell.*, 194:130–150, 2013.

Aria Haghighi and Dan Klein. Simple coreference resolution with rich syntactic and semantic features. In *EMNLP*, 2009.

Aria Haghighi and Dan Klein. Coreference resolution in a modular, entity-centered model. In *HLT-NAACL*, 2010.

Hannaneh Hajishirzi, Leila Zilles, Daniel S. Weld, and Luke S. Zettlemoyer. Joint coreference resolution and named-entity linking with multi-pass sieves. In *EMNLP*, 2013.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 8:1735–80, 1997.

Jun'ichi Kazama and Kentaro Torisawa. Exploiting wikipedia as external knowledge for named entity recognition. In *EMNLP-CoNLL*, 2007.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. Frustratingly easy neural domain adaptation. In *COLING*, 2016.

Young-Bum Kim, Karl Stratos, and Dongchan Kim. Adversarial adaptation of synthetic or stale data. In *ACL*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Heeyoung Lee, Yves Peirsman, Angel X. Chang, Nathanael Chambers, Mihai Surdeanu, and Daniel Jurafsky. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *CoNLL Shared Task*, 2011.

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

Kenton Lee, Luheng He, Mike Lewis, and Luke S. Zettlemoyer. End-to-end neural coreference resolution. In *EMNLP*, 2017.

Kenton Lee, Luheng He, and Luke S. Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. 2018.

Xiaoqiang Luo. On coreference resolution performance metrics. In *HLT/EMNLP*, 2005.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014. URL `http://www.aclweb.org/anthology/P/P14/P14-5010`.

Sebastian Martschat and Michael Strube. Latent structures for coreference resolution. *TACL*, 3:405–418, 2015.

David N. Milne and Ian H. Witten. Learning to link with wikipedia. In *CIKM*, 2008.

Nafise Sadat Moosavi and Michael Strube. Lexical features in coreference resolution: To be used with caution. In *ACL*, 2017.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

Vincent Ng. Supervised noun phrase coreference research: The first fifteen years. In *ACL*, 2010.

Haoruo Peng, Kai-Wei Chang, and Dan Roth. A joint framework for coreference resolution and mention head detection. In *CoNLL*, 2015.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matthew Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018.

Simone Paolo Ponzetto and Michael Strube. Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *HLT-NAACL*, 2006.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *EMNLP-CoNLL Shared Task*, 2012.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nate Chambers, Mihai Surdeanu, Daniel Jurafsky, and Christopher D. Manning. A multi-pass sieve for coreference resolution. In *EMNLP*, 2010.

Altaf Rahman and Vincent Ng. Supervised models for coreference resolution. In *EMNLP*, 2009.

Altaf Rahman and Vincent Ng. Coreference resolution with world knowledge. In *ACL*, 2011.

Lev-Arie Ratinov and Dan Roth. Learning-based multi-sieve co-reference resolution with knowledge. In *EMNLP-CoNLL*, 2012.

Lev-Arie Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. In *ACL*, 2011.

Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *ACL/IJCNLP*, 2009.

Joseph P. Turian, Lev-Arie Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *ACL*, 2010.

Olga Uryupina and Massimo Poesio. Domain-specific vs. uniform modeling for coreference resolution. In *LREC*, 2012.

Sam Wiseman, Alexander M. Rush, Stuart M. Shieber, and Jason Weston. Learning anaphoricity and antecedent ranking features for coreference resolution. In *ACL*, 2015.

Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. Learning global features for coreference resolution. In *HLT-NAACL*, 2016.

Jian-Bo Yang, Qi Mao, Qiaoliang Xiang, Ivor W. Tsang, Kian Ming Adam Chai, and Hai Leong Chieu. Domain adaptation for coreference resolution: An adaptive ensemble approach. In *EMNLP-CoNLL*, 2012.

Yuan Zhang, Regina Barzilay, and Tommi S. Jaakkola. Aspect-augmented adversarial networks for domain adaptation. *TACL*, 5:515–528, 2017.

Shanheng Zhao and Hwee Tou Ng. Domain adaptation with active learning for coreference resolution. In *Louhi@EACL*, 2014.

Index