Robustness in Coreference Resolution

vorgelegt von

Nafise Sadat Moosavi

aus Iran

Referent: Prof. Dr. Michael Strube
Korreferent: Prof. Dr. Simone Paolo Ponzetto
Einreichung: 16 Februar 2018

Abstract

Coreference resolution is the task of determining different expressions of a text that refer to the same entity. The resolution of coreferring expressions is an essential step for automatic interpretation of the text. While coreference information is beneficial for various NLP tasks like summarization, question answering, and information extraction, state-of-the-art coreference resolvers are barely used in any of these tasks. The problem is the lack of robustness in coreference resolution systems. A coreference resolver that gets higher scores on the standard evaluation set does not necessarily perform better than the others on a new test set.

In this thesis, we introduce robustness in coreference resolution by (1) introducing a reliable evaluation framework for recognizing robust improvements, and (2) proposing a solution that results in robust coreference resolvers.

As the first step of setting up the evaluation framework, we introduce a reliable evaluation metric, called LEA, that overcomes the drawbacks of the existing metrics. We analyze LEA based on various types of errors in coreference outputs and show that it results in reliable scores. In addition to an evaluation metric, we also introduce an evaluation setting in which we disentangle coreference evaluations from parsing complexities. Coreference resolution is affected by parsing complexities for detecting the boundaries of expressions that have complex syntactic structures. We reduce the effect of parsing errors in coreference evaluation by automatically extracting a minimum span for each expression. We then emphasize the importance of out-of-domain evaluations and generalization in coreference resolution and discuss the reasons behind the poor generalization of state-of-the-art coreference resolvers.

Finally, we show that enhancing state-of-the-art coreference resolvers with linguistic features is a promising approach for making coreference resolvers robust across domains. The incorporation of linguistic features with all their values does not improve the performance. However, we introduce an efficient pattern mining approach, called EPM, that mines all feature-value combinations that are discriminative for coreference relations. We then only incorporate feature-values that are discriminative for coreference relations. By employing EPM feature-values, performance improves significantly across various domains.

Zusammenfassung

Koreferenz-Resolution ist die Aufgabe, diejenigen Erwähnungen in einem Text zu finden, die sich auf dieselbe Entität beziehen. Diese Aufgabe ist essenziell für die automatische Interpretation von Texten. Obwohl Koreferenz-Resolution so wichtig für viele NLP-Bereiche wie Automatische Zusammenfassung, Frage-Antwort-Systeme und Informationsextraktions-Systeme ist, werden die neusten Koreferenz-Systeme selten für diese Aufgaben verwendet. Das Problem ist Zuverlässigkeit: Ein Koreferenz-System, welches besser auf dem Standard-Datenset operiert, ist nicht zwangsläufig auch besser auf neuen Datensets. In dieser Arbeit erhöhen wir die Zuverlässigkeit von Verbesserungen indem wir (1) ein Evaluations-Framework präsentieren, das zuverlässige Verbesserungen auch erkennt und (2) einen Weg aufzeigen, zuverlässige Verbesserungen zu erreichen.

Als erster Schritt hin zu einem neuen Evaluations-Framework präsentieren wir ein neues Evaluations-Maß (LEA) welches die Nachteile bisheriger Maße überwindet. Wir analysieren unser neues Maß auf verschiedenen Fehlerarten, die Koreferenz-Systeme ausgeben und zeigen, dass es immer zuverlässige Werte liefert. Zusätzlich präsentieren wir ein Evaluations-Setting, welches Koreferenz-Evaluierung von Parsing-Schwierigkeiten trennt – Koreferenz-Resolution wird von solchen Schwierigkeiten stark beeinflusst, da sie dazu führen, dass koreferente Ausdrücke nicht oder nur unvollständig erkannt werden. Wir reduzieren die genannten Schwierigkeiten, indem wir automatisch minimale Spannweiten für jede Erwähnung extrahieren. Dann gehen wir auf die Wichtigkeit von Out-of-domain-Evaluierungen ein und zeigen auf, warum aktuelle Koreferenz-Systeme so schlecht generalisieren. Schließlich zeigen wir auf, dass linguistische Features einen wichtigen Beitrag leisten, Koreferenz-Systeme robust bezüglich der Domäne zu machen. Jedes Feature ungeprüft mit allen Instanziierungen zu verwendet, erhöht jedoch nicht die Performanz. Deswegen führen wir einen effizienten Pattern-Mining-Algorithmus ein, der alle Feature-Instanzen erkennt, die tatsächlich diskriminativ für Koreferenz-Resolution sind. Wenn man nur diese Instanzen verwendet (zusätzlich zu den vom jeweiligen Koreferenz-System schon verwendeten Features und deren Instanzen), wird sowohl die Performanz, als auch die Zuverlässigkeit von Koreferenz-System auch domänenübergreifend signifikant erhöht.

Contents

Part I

Introduction

Chapter 1

Introduction

Coreference resolution is the task of finding different expressions of a text that refer to the same entity. For instance, in Example 1.1, "his" and "the 40 year old Mr. Murakami" refer to the same entity. Similarly, "it" corefers with the noun phrase "his 1987 novel, Norwegian Wood" in this example.

(1.1) [The 40 year old Mr. Murakami]$_{(1)}$ is a publishing sensation in Japan. [[His]$_{(1)}$ 1987 novel, Norwegian Wood]$_{(2)}$, sold more than four million copies since Kodansha published [it]$_{(2)}$ in 1987.

The availability of coreference information benefits various Natural Language Processing (NLP) tasks including automatic summarization, question answering, machine translation and information extraction. For instance, we need to resolve the coreference relations of Example 1.1 in order to answer the question "who published Murakami's 1987 novel?".

The importance of using coreference information in various applications is receiving more attention recently, e.g. in machine translation (Hardmeier et al., 2015; Guillou et al., 2016), question answering (Choi et al., 2017), text compression (Dhingra et al., 2017), text summarization (Durrett et al., 2016), slot-filling (Yu & Ji, 2016), math problem solving (Matsuzaki et al., 2017), or named entity linking (Sil & Florian, 2017). However, despite the fact that there have been remarkable improvements in the performance of coreference resolvers, the use of coreference resolution in higher-level applications is either limited to the use of simple rule-based systems, e.g. Yu & Ji (2016), Elsner & Charniak (2008), it has a very small effect on the overall performance, e.g. Dhingra et al. (2017), Durrett et al. (2016), or it introduces a major source of error, e.g. Matsuzaki et al. (2017), Sil & Florian (2017).

This is due to the fact that coreference developments are not robust. The performance of state-of-the-art coreference resolvers, which are trained on the standard training set and significantly outperform rule-based systems on the standard test set, drops significantly, i.e.

to a level on-par or worse than rule-based systems, if we evaluate them on a slightly different coreference corpus (Ghaddar & Langlais, 2016a).

In this thesis, we introduce robustness in coreference resolution. We first introduce a reliable evaluation framework for recognizing robust improvements in coreference resolution. We then propose an approach to achieve robust coreference resolvers that generalize across domains.

Overall, we address the following research questions in this work:

1. **Are the existing evaluation metrics reliable?**

   Coreference developments heavily rely on evaluation metrics. The success or failure of various coreference models, e.g. mention-pair vs. entity-based, and various feature sets, e.g. syntactic or semantic features, is solely determined based on the resulting scores of evaluation metrics. By comparing the evaluation scores, we determine which system performs best, which model suits coreference resolution better, and which feature set is useful for improving the recall or precision of a coreference resolver. Therefore, evaluation metrics play an important role in the advancement of the underlying technology, and it is imperative for the evaluation metrics to be reliable. In order to ensure robustness in coreference resolution, the first step is to investigate the reliability of the coreference evaluation metrics.

2. **Why do state-of-the-art coreference resolvers generalize poorly?**

   As mentioned above, there have been remarkable improvements in coreference resolution, i.e. more than ten percent based on various evaluation metrics from 2011 to 2017. For instance, Example 1.2 shows a sample output of the state-of-the-art coreference resolver (Lee et al., 2017) on the development set of the CoNLL corpus, i.e. the standard dataset for coreference evaluation. The system correctly recognizes all coreference relations of the given text, i.e. "the country" refers to "El Salvador" and "the guerrillas" refers to "the country's leftist rebels", none of which are trivial to resolve.

   (1.2) [El Salvador's]$_{(1)}$ government opened a new round of talks with [[the country's]$_{(1)}$ leftist rebels]$_{(2)}$ in an effort to end a decade-long civil war. A spokesman said [the guerrillas]$_{(2)}$ would present a cease-fire proposal during the negotiations in Costa Rica that includes constitutional and economic changes.

   On the other hand, Example 1.3 shows two very simple sentences, which do not belong to the CoNLL data. In this example, there are only two candidate antecedents for the pronoun "it", among which "that brown table" is the only one that has compatible animacy with "it", i.e. both "that brown table" and "it" are inanimate. However, the

state-of-the-art system of Lee et al. (2017) does not detect any coreference relation for this example.

(1.3) I don't want to buy that brown table. It is very big for the living room.

This indicates that there is a critical issue with the generalization of coreference resolvers. This generalization problem becomes more critical considering the fact that coreference resolution is not an end-task and it is going to be employed in tasks and domains for which we do not have coreference annotated corpora.

3. **How to improve the generalization and develop robust coreference models?**

   By addressing the first two questions, we set up a reliable evaluation framework to recognize robust improvements. The next question is then how to improve generalization so the improvements will not be limited to the standard evaluation sets and to have consistent improvements across domains.

1.1 Thesis Contributions

In order to answer the first question, we analyze all current evaluation metrics to explore their drawbacks. Apart from the known issues of the current evaluation metrics, we discover a new problem, namely the mention identification effect, that leads to counterintuitive recall and precision values for all evaluation metrics except for MUC (Vilain et al., 1995). The MUC metric, on the other hand, is the least discriminative metric for coreference evaluation.

As a result, we introduce a new evaluation metric, called LEA, that overcomes all the drawbacks of the existing metrics. We perform thorough analyses on the LEA metric based on various types of errors in the coreference output and show that LEA is a reliable metric for coreference evaluation.

In the standard setting, mentions are annotated and evaluated using their maximum span[1]. The use of maximum spans entangles coreference resolution with parsing complexities like prepositional phrase attachments. Therefore, by using maximum spans in coreference evaluation, we directly penalize coreference resolvers because of parsing errors. An existing solution to this problem is to manually annotate the corresponding minimum span of each mention. However, this solution is costly and does not scale to large corpora. We introduce an approach to automatically extract minimum spans of both key and system mentions during evaluation. Our approach does not require any manual annotation, and therefore can be applied on any coreference corpus. Based on the analyses on the corpora that include manually annotated minimum spans, we show that the automatically extracted minimum spans are compatible

[1] A span is the begin and end offsets of a mention in a text.

with the ones annotated by human experts. We provide an open source implementation of all evaluation metrics which evaluates coreference outputs using both maximum and minimum spans.

Regarding the second question, we show that (1) there is a considerable overlap between the training and test sets of the CoNLL data that rewards the systems that memorize more from the training data, and (2) relying on lexical features as the main source of information, i.e. as the state-of-the-art coreference resolvers do, creates a strong bias towards resolving mentions that are seen during training. Therefore, a coreference resolver that mainly uses lexical features performs poorly on unseen mentions. As a result, an improvement on the CoNLL test set does not imply a better coreference model. The improvements may be due to better memorization of the training data. We argue that performing out-of-domain evaluations is a must in coreference evaluation in order to ensure meaningful improvements.

By using the LEA metric, considering minimum instead of maximum spans, and performing out-of-domain evaluations, we establish a reliable framework for coreference evaluation. We then address the third question to propose an approach to make coreference resolvers robust across domains. We improve generalization by incorporating linguistic features. We propose a new approach, called EPM, that efficiently selects all feature-values that are useful for discriminating coreference relations. EPM casts the problem of finding discriminative features for coreference relations as a pattern mining approach. It efficiently mines all combinations of feature-values that are discriminative for coreference relations. We then incorporate the selected feature-values in a state-of-the-art coreference resolver. We show that the incorporation of EPM feature-values significantly improves the performance in both in-domain and out-of-domain evaluations, and therefore makes the baseline coreference resolver more robust across domains.

1.2 Thesis Structure

In Chapter 2, we review the common coreference resolution corpora, existing ways to model the coreference problem, and the coreference resolvers that we use as baselines throughout this thesis.

The second part of the thesis, including Chapters 3, 4 and 5, introduces a reliable framework for coreference evaluation. In Chapter 3, we first review existing evaluation metrics as well as their drawbacks. We then introduce LEA, i.e. the Link-Based Entity-Aware metric, that overcomes all drawbacks of the existing metrics. We perform thorough analyses on the LEA metric and show it is a reliable metric for coreference evaluation.

In Chapter 4, we first highlight the problem of using maximum spans in coreference evaluation, i.e. penalizing coreference resolvers directly because of parsing errors. We then intro-

duce an algorithm to automatically extract minimum spans and provide an evaluation package in which all evaluations are performed based on both maximum and minimum spans.

In Chapter 5, we discuss the importance of out-of-domain evaluations in coreference resolution and investigate the reasons that state-of-the-art coreference resolvers do not generalize well.

After establishing a reliable framework for coreference evaluation, in the third part of the thesis that includes Chapters 6, 7 and 8, we introduce a solution, i.e. reconciling coreference resolution with linguistic features, to make coreference resolvers robust across domains.

In Chapter 6, we introduce a new approach, i.e. EPM, for recognizing feature-values that are discriminative for coreference relations. EPM is a new pattern mining approach that efficiently mines the set of feature-value combinations that are discriminative for the class label.

In Chapter 7, we evaluate the efficacy and time efficiency of EPM compared to other discriminative pattern mining approaches on standard machine learning datasets. We show that in comparison to its counterparts, EPM is very efficient, and is therefore scalable to large datasets while resulting in patterns with on-par discriminative power.

In Chapter 8, we evaluate the feature-values that are selected by EPM in coreference resolution. We first show that it is important to use discriminative feature-values and not all the feature-values. We then show that the selected feature-values significantly improve the performance and result in robust improvements across domains.

Finally, in the last part of the thesis, i.e. Chapter 9, we summarize the contributions and conclusions of this thesis and discuss some future work.

1.3 Published Work

Most Research presented in this thesis is an extension of the published work by the author of this thesis. Chapters 3, 4, and 5 are extensions of Moosavi & Strube (2016b), Moosavi et al. (2019), and Moosavi & Strube (2017a), respectively. Chapters 6, 7 and 8 are extensions of Moosavi & Strube (2018). Moosavi & Strube (2017b), Heinzerling et al. (2017), and Moosavi & Strube (2016a) are also by-products of the research that is done for this thesis.

Chapter 2

Background

In this chapter, we first define the task of coreference resolution. We then briefly give an overview of the common corpora and coreference resolution models in the literature. We then describe the coreference resolution systems that are used in various experiments of this work.

2.1 Coreference Resolution

The relation between two expressions referring to the same entity is defined as coreference relation (Hirschman & Chinchor, 1997). In other words, assume m_1 and m_2 are two expressions that both have a unique referent in the text. Considering Referent(m_i) to be the entity that is referred to by m_i, the coreference relation is defined as:

Definition 1 m_1 *and* m_2 corefer *if and only if Referent*(m_1) = *Referent*(m_2).

The focus of the majority of existing work is on noun phrase coreference resolution, i.e. m_1 and m_2 are both noun phrases in Definition 1. For instance, the coreference relation of Example 2.1[1] is between two noun phrases.

(2.1) [Wang Jin]$_{(1)}$ says that [he]$_{(1)}$ has decided not to continue serving as KMT vice chairman.

In English, three different types of nouns phrases can be chosen for referring to an entity:

1. **Proper names** are names of specific entities. For instance, "Zhuanbi Village" and "Eight Route Army" in Example 2.2 are proper names.

   (2.2) This is **Zhuanbi Village**, where the **Eight Route Army** was headquartered back then.

[1] The indices in parentheses denote to which key entity the mentions belong.

2. **Nominals** are noun phrases that have a noun as their head. For instance, "a wall" and "the headquarters" in Example 2.3 are nominals. Nominals are also called common nouns.

   (2.3) We found a map on **a wall** outside **the headquarters**.

3. **Pronouns** can in turn be from one of the following categories:

   (a) Reflexive: A lone protester parked **herself** outside the UN.

   (b) Definite: My mother was Thelma Wahl. **She** was ninety years old.

   (c) Indefinite: As **one** can see from the picture, the results were not satisfying.

   (d) Demonstrative: Nobody mentioned **that** as a possibility for me.

It is worth noting that none of the above forms, which are listed for referring noun phrases, are always referring. Based on the context, each noun phrase can take one of the following semantic functions (Poesio, 2016):

Referring: A referring noun phrase either introduces a new entity in a discourse, or it refers to a previously introduced entity. The noun phrases specified in Example 2.1 are referring noun phrases. Referring noun phrases, or in general, referring expressions, are called **mentions** in the coreference literature. A mention that introduces a new entity into the discourse is called a **discourse-new** mention. A discourse-new mention may be a singleton or it may be the first mention of a coreference chain. Mentions which refer to previously introduced entities are called **discourse-old** mentions. Discourse-old mentions are commonly referred to as **anaphoric** mentions in the coreference resolution literature, e.g. Zhou & Kong (2009), Ng (2009), Wiseman et al. (2015), Lassalle & Denis (2015), inter alia.

Predicative: A predicative noun phrase expresses a property of an object. For instance, the noun phrase "a weekly series that premiered three weeks ago" in Example 2.4 expresses a property of "Capital City" and does not refer to an entity.

(2.4) Capital City is a weekly series that premiered three weeks ago

Expletive: Expletive noun phrases only fill a syntactic position. For instance, "it" in Example 2.5 is an expletive pronoun.

(2.5) **It** seems that we have a scheme for the future.

Quantificational: A quantificational noun phrase is a noun phrase that quantifies. For instance, "some of the attendees" in Example 2.6 is such a noun phrase.

(2.6) **Some of the attendees** did not find that discussion interesting. They were bored during the discussion.

In this example, the pronoun "they" refers to "some of the attendees". However, they do not corefer because "some of the attendees" is not a referring expression and therefore does not have a referent.

A *substitution test* can be used in such cases to examine coreference relations (Mitkov, 2002). For instance, "they" in Example 2.7 can be substituted by "John and Mary" and the sentence would have the same meaning. Example 2.9 shows the sentence of Example 2.6 in which "they" is substituted with "some of the attendees". As we can see, the substitution changes the statement of the sentence. Therefore, "some of the attendees" and "they" in Example 2.6 do not corefer.

(2.7) John and Mary did not find that discussion interesting. They were bored during the discussion.

(2.8) John and Mary did not find that discussion interesting. John and Mary were bored during the discussion.

(2.9) Some of the attendees did not find that discussion interesting. Some of the attendees were bored during the discussion.

It is also worth mentioning that while the majority of coreference resolvers only resolve noun phrases, coreference relations are not limited to noun phrases. For instance, the verb "talk" in Example 2.10 is an example of coreferring expressions that are not noun phrases.

(2.10) The vicar refuses to $[\text{talk}]_{(1)}$ about it, saying $[\text{it}]_{(1)}$ would reopen the wound.

Confusion in Coreference Definition. It is not always easy, even for humans, to correctly recognize coreference relations. There are numerous coreference annotated corpora with different annotation schemes. The existence of many distinct annotation schemes for coreference resolution is indeed an indicator that there is a disagreement in defining coreference relations.

The following cases are examples of disagreements in defining coreference relations:

1. referring vs. predicative:

(2.11) Mr. Lieber, the actor who plays Mr. Hoffman, says he was concerned that the script would "misrepresent an astute political mind" but that his concerns were allayed. The producers, he says, did a good job of depicting **someone "who had done so much, but who was also a manic-depressive"**.

The boldfaced noun phrase in Example 2.11[2] can be either interpreted as a mention referring to *Mr. Hoffman* or a predicative noun phrase. Predicative noun phrases, even when they are clearly predicative, are annotated in coreference relations of the MUC and ACE datasets. For instance, in Example 2.12 from the ACE dataset, "the Hong Kong club" and "a charitable entity" are annotated as coreferent.

(2.12) [The Hong Kong club]$_{(1)}$ is [a charitable entity]$_{(1)}$.

2. referring vs. expletive:

(2.13) I guess we might as well go through Dansville. So does **it** seem like a reasonable alternative to dealing with the engine that's hanging out in Elmira.

it in Example 2.13[3] can be either interpreted as a noun phrase referring to *go through Dansville* or an expletive.

3. referring vs. quantificational:

(2.14) [Some groups]$_{(1)}$, [they]$_{(1)}$ are rehearsing [[their]$_{(1)}$ service]$_{(2)}$, hoping that [they]$_{(1)}$ will become used to [the service]$_{(2)}$.

Many quantificational noun phrases are annotated in coreference relations of the OntoNotes dataset. Example 2.14 is an annotated sentence from the CoNLL-2012 development set.

2.2 Corpora

There are numerous corpora with coreference annotations including MUC, ACE, CoNLL, etc. The CoNLL-2012 shared task dataset (Pradhan et al., 2012) is the largest available corpus that is annotated with coreference information. After the introduction of this dataset, it became the most prominent corpus in the coreference literature. Henceforth, we refer to this dataset as CoNLL or CoNLL-2012.

As we will discuss in Chapter 5, it is not enough to only evaluate coreference resolvers on a single corpus, i.e. CoNLL-2012. Therefore, we choose another corpus, i.e. WikiCoref (Ghaddar & Langlais, 2016b), for out-of-domain evaluations. The reason for choosing WikiCoref is that the coreference annotation scheme of WikiCoref is almost the same as that of CoNLL-2012, which makes it a great candidate for out-of-domain evaluations when a system is trained on the CoNLL data.

[2]From Poesio (2016)
[3]From Poesio (2016)

2.2.1 CoNLL-2012

CoNLL-2012 is a subpart of the OntoNotes 5.0 corpus (Weischedel et al., 2013). It is a large, cross-domain and cross-lingual corpus of text that includes several layers of syntactic and shallow semantic annotations. Figure 2.1 shows an example sentence from the CoNLL corpora. CoNLL contains annotated documents from various domains including broadcast conversations (bc), broadcast news (bn), magazine articles (mz), newswire (nw), Bible text (pt), telephone conversations (tc), and weblog texts (wb). Annotated texts include three different languages including English, Chinese, and Arabic. In this work, we only use the English portion of this corpus.

It is worth noting that singletons, i.e. mentions that do not belong to a coreference chain, are not annotated in CoNLL.

Standard splits for training, development and test sets are established for coreference evaluations based on the ones used in the CoNLL-2012 shared task. We use these standard splits for in-domain evaluations of this work. There are 2374, 303 and 322 documents in the training, development and test sets, respectively. Long documents in CoNLL are split into two or more short documents. However, there is no relation between the coreference annotations of the split documents. Therefore, they are considered independent documents.

Documents from different domains are distributed uniformly in the training, development and test data. For every 10 numbered documents, 8 are in the training, 1 is in the development and 1 is in the test set. Therefore, all splits contain all included domains.

bc/cctv/00/cctv_0000	0	0	The	DT	(TOP(S(NP(NP*	-	-	-	Speaker#1	*	(ARG1*	-
bc/cctv/00/cctv_0000	0	1	construction	NN	*)	construction	-	1	Speaker#1	*	*	-
bc/cctv/00/cctv_0000	0	2	of	IN	(PP*	-	-	-	Speaker#1	*	*	-
bc/cctv/00/cctv_0000	0	3	Hong	NNP	(NP(NML*	-	-	-	Speaker#1	(FAC*	*	(12|(23
bc/cctv/00/cctv_0000	0	4	Kong	NNP	*)	-	-	-	Speaker#1	*	*	23)
bc/cctv/00/cctv_0000	0	5	Disneyland	NNP	*)))	-	-	-	Speaker#1	*)	*)	12)
bc/cctv/00/cctv_0000	0	6	began	VBD	(VP*	begin	01	1	Speaker#1	*	(V*)	-
bc/cctv/00/cctv_0000	0	7	two	CD	(ADVP(NP*	-	-	-	Speaker#1	(DATE*	(ARGM-TMP*	-
bc/cctv/00/cctv_0000	0	8	years	NNS	*)	year	-	1	Speaker#1	*	*	-
bc/cctv/00/cctv_0000	0	9	ago	RB	*)	-	-	-	Speaker#1	*)	*)	-
bc/cctv/00/cctv_0000	0	10	,	,	*	-	-	-	Speaker#1	*	*	-
bc/cctv/00/cctv_0000	0	11	in	IN	(PP*	-	-	-	Speaker#1	*	(ARGM-TMP*	-
bc/cctv/00/cctv_0000	0	12	2003	CD	(NP*)))	-	-	-	Speaker#1	(DATE)	*)	(13)
bc/cctv/00/cctv_0000	0	13	.	.	*))	-	-	-	Speaker#1	*	*	-

Figure 2.1: An example sentence from the CoNLL-2012 corpus in the CoNLL format (Pradhan et al., 2011). Annotations include part-of-speech tags (5th column), constituency parses (6th column), predicate lemma for the rows with semantic role information (7th column), speaker or author name where available (10th column), named entities (11th column), predicate argument structure information (12th to the one before the last column), coreference chain information (last column).

2.2.2 WikiCoref

We use the WikiCoref (Ghaddar & Langlais, 2016b) dataset for out-of-domain evaluations. WikiCoref is a small English dataset with coreference annotations. It contains 30 difference documents from the English version of Wikipedia. The coreference annotation scheme in WikiCoref is almost the same as that of the CoNLL dataset. The only difference is that nested mentions and verbs are not annotated in WikiCoref annotations.

Each coreferential mention in the WikiCoref dataset is tagged with the corresponding Freebase topic when available. We have not used this information in our experiments.

The size of documents in WikiCoref is relatively larger than that of the CoNLL-2012 documents. Figure 2.2 shows an example of coreferring mention annotations in the WikiCoref dataset.

```
<markable id="markable-444" span="word-2519..word-2519" coref-class="set-520"
topic="http://rdf.freebase.com/ns/m.02mjmr" coreftype="ident"
mentiontype="ne" mmax-level="coref" />
<markable id="markable-2436" span="word-2570..word-2575" coref-class="set-539"
topic="nan" coreftype="ident" mentiontype="np" mmax-level="coref" />
<markable id="markable-1526" span="word-2768..word-2768" coref-class="set-540"
topic="nan" coreftype="ident" mentiontype="pro" mmax-level="coref" />
```

Figure 2.2: An example of WikiCoref annotations. Each “markable” element represent a mention. The “id”, “span”, and “coref-class” attributes determine the mention id, the mention span and the corresponding coreference chain id, respectively. “topic” specifies the Freebase topic if available. “ident” value of “coreftype” specifies coreferring mentions. “mentiontype” determines the type of mentions, i.e. proper name (ne), nominal (np), and pronominal (pro).

2.2.3 Other Corpora

While the main focus of the recent literature is on the CoNLL dataset, there are also numerous other coreference annotated corpora. The main reason that these corpora are not used together, e.g. one or more for training and many more for testing, is that they have different coding schemes resulting from different definitions of coreference relations. As shown in the literature (Stoyanov et al., 2009; Recasens & Vila, 2010), corpus parameters should be in agreement for the fair comparison of coreference approaches.

MUC (Hirschman & Chinchor, 1997), ACE (Mitchell et al., 2002), ARRAU (Uryupina et al., 2016) are examples of corpora with various annotation schemes compared to the CoNLL dataset. Before the introduction of the CoNLL corpus, MUC and ACE were the standard

corpora for evaluating coreference systems. MUC-6 and MUC-7 are the first two corpora for evaluating coreference approaches, which were created as part of the Message Understanding Conference (MUC). The ACE corpus is created as part of the Automatic Content Extraction (ACE) initiative. A critical discussion regarding the MUC and ACE coding scheme is that nominal predicates and appositive phrases are treated as coreferential (van Deemter & Kibble, 2000).

Unlike MUC and CoNLL, ACE includes singletons and is limited to seven semantic types, i.e. person, organization, geo-political entity, location, facility, vehicle, and weapon.

In comparison to CoNLL, ARRAU allows ambiguity, and it also includes the annotation of discourse deixis.

There are also various coreference annotated corpora for scientific domains. The corpora annotated by Cohen et al. (2010), Schäfer et al. (2012), and Chaimongkol et al. (2014) are examples of such datasets.

The main focus of the current literature is to improve the coreference resolution performance in English, as it is the case in this thesis. However, there are also coreference annotated corpora in other languages. The Potsdam Commentary Corpus and Tüba-D/Z corpora for German (Stede, 2004; Hinrichs et al., 2005), COREA for Dutch (Hendrickx et al., 2008), AnCora for Catalan and Spanish (Recasens & Martí, 2009), and the Live Memories corpus for Italian (Rodrıguez et al., 2010) are examples of coreference annotated corpora for other languages. The SEMEVAL-2010 Corpus (Recasens et al., 2010) includes subsets of Tüba-D/Z, COREA, AnCora, Live Memories, and CoNLL corpora. All the included datasets are converted to a common format and annotated in the most consistent manner. Singletons are automatically detected and annotated in the SEMEVAL-2010 Corpus.

2.3 Coreference Resolution Models

Current models for coreference resolution can be classified into three main categories: (1) mention-pair models, (2) mention-ranking models, and (3) entity-based models. We briefly overview each of the above models in the following sections.

2.3.1 Mention-Pair Models

Before the popularity of ranking models (Durrett & Klein, 2013; Martschat & Strube, 2015; Wiseman et al., 2015), mention-pair models were the most common coreference model since Soon et al. (2001). The feature set of mention-pair models includes individual properties of antecedent and anaphor and also a set of features that describe the pairwise relation of these two mentions.

Mention-pair models classify given pairs of mentions as either coreferent or non-coreferent. Mention-pairs are usually constructed by considering each mention as an anaphor and all of its previous mentions as candidate antecedents.

Coreference chains are then constructed based on pairwise decisions and by using a clustering algorithm. The clustering algorithm can vary from naive methods like:

- closest-first: connecting each mention to its closest antecedent that is labeled as coreferent, e.g. Soon et al. (2001)
- best-first: connecting each mention to its highest scoring antecedent with a coreferent label, e.g. Ng & Cardie (2002)
- merge-all: connecting all mention-pairs that are labeled as coreferent, e.g. Denis & Baldridge (2009a)

Such naive clustering methods only consider compatibility of individual mention-pairs. Therefore, dependencies beyond mention-pairs will be ignored by these methods. For instance, a pairwise model may detect both (Mr. Kostunica, Kostunica) and (Kostunica, she) pairs as coreferent. However, the model could have inferred from the first pair that "Kostunica" is a male name and therefore it cannot be coreferent with "she".

In order to address this problem in pairwise approaches, several more informed clustering methods have been proposed including:

- Bell-tree clustering: Luo et al. (2004) model the search space of creating entities from individual mentions as a tree. The root of the tree is a partial entity that contains the first mention of the document. Each level of the tree is created by processing one mention at a time. Each mention can be either linked to one of the previous partial entities or starts a new entity. For instance, the second mention of the text can be either linked to the first partial entity at the root node, or starts a new entity. Leaves of the tree represent all possible coreference outcomes. The probability of linking a mention m to a partial entity e is estimated by either of the following equations:

$$Pr(\text{link}|e, m), \tag{2.15}$$

$$\max_{m_j \in e} Pr(\text{link}|m_k, m) \tag{2.16}$$

  If the Bell-tree clustering is applied on the output of a mention-pair model, Equation 2.16 will be used for scoring each decision. However, it is also possible to apply this clustering algorithm on the output of a mention-entity model, i.e. using Equation 2.15 for scoring each decision.

  Luo et al. (2004) present an algorithm in order to find optimal clusterings in the tree. They made their algorithm computationally feasible by pruning the search space.

- graph partitioning: one can build a graph in which each node is a mention and edges are scored based on the coreference probability of the corresponding nodes, i.e. mention-pairs. Various graph clustering algorithms like MinCut (Stoer & Wagner, 1997), e.g. Nicolae & Nicolae (2006), or relaxation labeling (Hummel & Zucker, 1983), e.g. Sapena et al. (2010), can be used in order to find partitions of mentions, i.e. estimated coreference chains, based on the graph structure and not only individual mention-pairs.

  It is worth noting that this approach can also be used based on the output of an unsupervised coreference resolution system. For instance, Moosavi & GhassemSani (2014) build a graph incrementally based on the output of Stanford rule-based system (Raghunathan et al., 2010). They then apply a relaxation labeling algorithm on each partially constructed graph in order to determine partial entities of the document.

- integer linear programming: in order to obtain a clustering with regard to the transitivity of coreference relations, one can apply integer linear programming on the output of a pairwise coreference resolver. Klenner (2007) and Finkel & Manning (2008) are examples of such approaches.

- joint clustering: above clustering methods are performed independently from and after the classification of mention-pairs. However, the classification and clustering steps can also be performed jointly. Methods proposed by McCallum & Wellner (2003), Finley & Joachims (2005), Song et al. (2012) are examples of joint approaches in which the mention-pair classifier and the clustering method are learned jointly.

2.3.2 Mention-Ranking Models

Mention-ranking models are the most successful and popular coreference models right now. The general idea is to enhance the mention-pair modeling of coreference relations by considering all antecedents of a single mention together. This way, the model can capture the competition among various candidate antecedents of a single mention.

For instance, consider the following two examples. Gold mentions are enclosed in square brackets. Mentions with the same text are marked with different indices. The indices in parentheses denote to which key entity the mentions belong.

(2.17) These items, which were the pride of [the Ocean Park$_1$]$_{(1)}$, have made [this place]$_{(1)}$ the most popular tourist attraction in [Hong Kong$_1$]$_{(2)}$ for some time. However, since [Disney]$_{(3)}$ entered [Hong Kong$_2$]$_{(2)}$, [the Ocean Park$_2$]$_{(1)}$, sharing the same city as [Disney$_2$]$_{(3)}$, has felt the pressure of competition.

(2.18) After six decades [the airman]$_{(1)}$ is exhumed from [his$_1$]$_{(1)}$ icy tomb and thawed

out. But [he]$_1$ is wearing no military ID. Did [this World War Two pilot]$_{(1)}$ perish when [his$_2$]$_{(1)}$ training flight crashed in the mountains?

A mention-pair model tries to learn all pairwise relations including ("this place", "the Ocean Park$_2$") and ("his$_1$", "this World War Two pilot"). However, these pairs are not informative pairs to learn from. On the other hand, a ranking model does not enforce the classifier to learn from all coreferring pairs. Instead, the model can only concentrate on learning from more informative pairs, e.g. ("the Ocean Park$_1$", "the Ocean Park$_2$") and ("the airman", "this World War Two pilot"). The advantage of ranking models is that we do not need to manually determine informative mention-pairs. The informative pairs could be automatically determined during the learning process.

Yang et al. (2003) propose a simplified ranking model, i.e. the twin-candidate model. Instead of considering all candidate antecedents together, they compare pairs of candidate antecedents at a time to determine which of them is a better antecedent. The candidate antecedent that wins most of the pairwise comparisons, will be selected as the antecedent of the examined anaphor.

The twin-candidate model is extended in later works to consider all antecedents together. Assume we want to determine the antecedent of a mention m_i. Let $A(m_i)$ and $T(m_i)$ be the set of m_i's all candidate antecedents, and true antecedents, respectively. In the ranking model, the best candidate antecedent is selected using the following equation:

$$a^* = \underset{a_k \in A(m_i)}{\arg\max}\, score(a_k|m_i) \tag{2.19}$$

As we can see from Equation 2.19, the burden of the ranking model is put on the scoring function, i.e. $score(a_k|m_i)$. The inference method of the ranking model in Equation 2.19 is similar to the best-first clustering method of Section 2.3.1. However, the difference is in the learning of the scoring function. Various methods have been proposed for training a ranking model including:

- **learning from best antecedents during training, selecting best antecedents heuristically**: This approach is first used by Denis & Baldridge (2008). They learn the model parameters in a way that the closest true antecedent of a mention gets a higher score compared to other antecedents.

- **learning from best antecedents during training, selecting best antecedents automatically**: The ranking models used by Chang et al. (2012), Wiseman et al. (2015), and Clark & Manning (2016a) are examples of such methods. In such approaches, the model selects the highest-scoring true antecedent under the current model, i.e. $\hat{t} = \arg\max_{t \in T(m_i)} score(t|m_i)$, as the best antecedent during learning.

If we connect each mention to its highest scoring antecedent and consider the antecedent selection of all mentions of the document together, the resulting structure would be a tree. In this tree, the parent of each node is its selected antecedent. Therefore, coreference models that are known as antecedent-trees, e.g. Yu & Joachims (2009), Fernandes et al. (2012), Fernandes et al. (2014), Chang et al. (2013), Lassalle & Denis (2015), can also be put in this category of coreference resolution models.

- **summing over scores of all true antecedents**: For training the ranking model, instead of selecting one best antecedent, one can sum over all true antecedents of a mention.

  In this approach, a classifier is not forced to learn from every individual pair. Besides, it is not focused only on a single antecedent for each given mention. Instead, it can learn from all true antecedents that it finds informative.

  The ranking approaches used by Durrett & Klein (2013) and Lee et al. (2017) are examples of such approaches.

Denis & Baldridge (2008) first use an anaphoricity determination module to determine whether m_i is anaphoric. If m_i is classified as non-anaphoric, it wouldn't be processed by the ranking model.

For tackling non-anaphoric and anaphoric mentions in a more unified way, the later ranking approaches, e.g. Chang et al. (2012), Durrett & Klein (2013), use a dummy mention, i.e. $\emptyset$, among the list of candidate antecedents. Selecting a dummy mention as an antecedent of m_i indicates that m_i is a non-anaphoric mention. Therefore, for a non-anaphoric mention m_i the set of true antecedents is $\{\emptyset\}$.

2.3.3 Entity-Based Models

Mention-entity and entity-centric models are two variations of entity-based models. Mention-entity models, e.g. Luo et al. (2004), Daumé III & Marcu (2005), Yang et al. (2008), Rahman & Ng (2011), Klenner & Tuggener (2011), Ma et al. (2014), Björkelund & Kuhn (2014), Wiseman et al. (2016), process mentions of the text in a left-to-right fashion and decide about merging each of the processed mentions to a partially constructed entity.

Entity-centric models, e.g. Culotta et al. (2007), Stoyanov & Eisner (2012), Lee et al. (2013), Clark & Manning (2015), Clark & Manning (2016a), decide about merging two partially constructed entities.

In theory, the advantage of entity-based models in comparison to mention-pair and mention-ranking models is that they can incorporate more expressive features that are defined over entities instead of mentions or mention-pairs.

The majority of existing entity-based models define entity-based features manually. For instance, entity-based features can be constructed by applying *all*, *most*, and *none* coarse quantifier predicates to mention-based features. As an example, from "mention type=pronoun", one can build "all-pronouns=true" that indicates all mentions of the examined cluster are pronouns. Entity-based features can also be constructed by concatenating properties of individual mentions, e.g. "proper name-pronoun-pronoun" is an entity-based feature that is constructed based on the mention type property and a cluster that includes one proper name and two pronouns.

Wiseman et al. (2016) on the other hand, propose an approach in which entity-based features are learned automatically and implicitly by applying an LSTM network (Hochreiter & Schmidhuber, 1997) on partially constructed entities.

Overall, entity-based approaches are the most complex, potentially the most representative, and practically the least successful approaches in coreference resolution.

As an example, in Moosavi & Strube (2014), we convert all entity-based features of Lee et al. (2013) to their corresponding pairwise features and yet we obtain on-par performance with that of Lee et al. (2013). The entity-based model of Clark & Manning (2016a) compared to their mention-ranking model is another counterexample for the superiority of entity-based models or features. Clark & Manning (2016a) show that the results of the entity-based model are slightly better than those of their mention-ranking model. However, their mention-ranking model outperforms the entity-based one when they use better preprocessing and more training epochs (Clark & Manning, 2016b).

2.4 Examined Coreference Resolvers

In this section, we briefly review the coreference resolvers that we use as baselines throughout this work. The examined systems include: (1) the Stanford rule-based system[4] (Raghunathan et al., 2010; Lee et al., 2011), (2) the Berkeley coreference resolver[5] (Durrett & Klein, 2013), (3) cort[6] (Martschat & Strube, 2015), (4) deep-coref[7] (Clark & Manning, 2016a), and (5) e2e-coref[8] (Lee et al., 2017).

2.4.1 Stanford Rule-based System

The Stanford rule-based system uses a small set of simple heuristics, mainly string match features, to resolve coreference relations. This simple system is the winner of the CoNLL

[4]Available at `https://stanfordnlp.github.io/CoreNLP/coref.html`
[5]Available at `http://nlp.cs.berkeley.edu/projects/coref.shtml`
[6]Available at `https://github.com/smartschat/cort`
[7]Available at `https://github.com/clarkkev/deep-coref`
[8]Available at `https://github.com/kentonl/e2e-coref`

2011 shared task. The Stanford rule-based system does not require any training and it also does not incorporate any lexical features.

This system has a sieve architecture. Each sieve processes the text based on a different set of features. Sieves are ordered based on the precision of their included features. The text is first processed by the most precise sieve. Partially constructed entities of the first sieve will be extended by later sieves. Coreference decisions that are made by earlier sieves will not be disturbed by the following less-precise sieves.

The set of features that is used in the rule-based system is as follows. Features are listed in order of decreasing precision.

Speaker identification.

- "I" pronouns that have the same speakers are coreferent.

  (2.20) [*I*] mean [*I*] went to bed with nothing in my stomach either.

- "you" pronouns that have the same speaker are coreferent. This feature links the specified mentions Example 2.21 and Example 2.22. However, only the specified mentions of Example 2.22 are annotated as coreferent in the CoNLL development set and none of the "you" or "your" pronouns [9] in Example 2.21 are tagged as coreferent mentions.

  (2.21) here, [*you*] can come up close with the stars in [*your*] mind.

  (2.22) and that's all [*you*] had to sustain [*you*].

- a mention that has the same string as the speaker of an "I" pronoun is coreferent with the "I" pronoun. For instance, in the following example, the speaker of the second "I" is specified as "Paula", and therefore the system makes a link between the second "I" and "Paula" based on this feature.

  (2.23) I can assure you that [*Paula*]. Yeah right. [*I*] made a free house call for your doctor.

Exact match. If two mentions have the same span they are coreferent. For instance, this feature holds for three mention pairs in the following example.

(2.24) With this new [*Hong Kong*]$_1$ - [Zhuhai]$_2$ - [Macao]$_3$ bridge that basically leads to all three places, [Hong Kong]$_1$, [Macao]$_3$, and [Zhuhai]$_2$ ...

Relaxed string match. If two mentions have the same string after dropping the words after the head words, they are coreferent. This feature links the two identified mentions in the following example.

[9] The speaker of both mentions is identified as "SPEAKER #1"

(2.25) we got an interview with [the witness who led investigators to a landfill in the search for Natalee Holloway] ... [the witness] told us directly that in the beginning he felt like nobody was believing his story.

Precise features.

- acronym: two mentions that have an NNP tag and one of them is the acronym of the other are coreferent.

  (2.26) Back in [LA] the image hits too close to home ... Savela Vargas CNN [Los Angeles].

- Demonym: If one mention is the demonym of the other, they are coreferent. For instance, the rule-based system links "America" and "the Americans" in Example 2.27 and similarly "Aruban" and "the Aruban" in Example 2.28 based on the demonym feature. However, "the Aruban" in Example 2.28 is not annotated as a coreferent mention in the development set.

  (2.27) He wasn't killed because of his positive traits, which were his belief in Arab unity and challenging [America] ... The curse of Saddam will continue to chase them, chase [the Americans], and chase his executioners.

  (2.28) The last time that I was here in [Aruba] ... [the Aruban] authorities and we've had conversations about this.

It is worth noting that other features like appositive, role appositive, relative pronoun, and demonym are also included among the precise features. However, these relations are not annotated as coreferent relations in the CoNLL dataset and are only useful for other datasets like ACE [10].

Strict head match. Two mentions are coreferent based on this feature if all of the following conditions hold:

- cluster head match: the anaphor head should match the head word of at least one of the mentions that are in the partial cluster of the antecedent.
- word inclusion: the set of non-stop words in the partial cluster of the anaphor should be included in the set of non-stop words of the partial cluster of the antecedent.
- compatible modifiers: all the modifiers of the anaphor should be included in the modifiers of the candidate antecedent.

[10] Refer to Raghunathan et al. (2010) for detailed definitions of these features

– not i-within-i: anaphor and the candidate antecedent should not be in an i-within-i construct, i.e. one mention is a child noun phrase in the other mention's noun phrase constituent.

The marked mentions in Example 2.29 can be linked based on the strict head match feature.

(2.29) But Ye Daying in fact failed when he took [the reinstated exam]. Although it was the first time he had taken [the exam], it left him even more convinced that movies were something he could do.

Less strict head match. Two variants of the strict head match feature can be created by

– removing the compatible modifiers condition from the set of conditions in order to create a link between two mentions:

(2.30) So, Ye Daying went to take the exams for [the film institute]. His mother had never thought that he would sign up to take the exam for her work unit, [the Beijing Film Institute].

– removing the word inclusion condition. For instance, based on this feature, the system links "people" and "people like you" in Example 2.31. However, these two mentions are not coreferent.

(2.31) If [people] and their governors were enlightened enough to follow them we would all be a lot better off ... I am sworn to oppose you and [people like you].

Proper head match. Two mentions that are headed by proper nouns are coreferent if their heads match and the following constraints hold:

– Not i-within-i: the anaphor and the candidate antecedent should not be in an i-within-i construct.

– No location mismatches: the modifier of examined mentions should not contain different location named entities, spatial modifiers, or other proper nouns.

– No numeric mismatches: if the anaphor contains a number, it should also appear in the antecedent.

For instance, consider the "the China Traditional Martial Arts Imperial College" and "the China Traditional Literature Imperial College" mentions in Example 2.32. The system detects "College" as the head of both mentions and they do not violate location or numeric mismatch constraints. Therefore, based on the proper head match feature,

the rule-based system links these two mentions. However, they are not coreferent in the text.

(2.32) With the aim of promoting Chinese culture internationally, [the China Traditional Martial Arts Imperial College] and [the China Traditional Literature Imperial College] were established today in Beijing.

Relaxed head match. If the head of the anaphor matches any word in the antecedent's partial cluster, they are coreferent. Based on this feature, "Olson" can be linked to "Ted Olson lawyer for George W. Bush".

(2.33) [Ted Olson lawyer for George W. Bush], laying out the campaign's main claim. [Olson] made it through only 56 seconds of his arguments before a justice broke in.

Pronoun resolution. Except for the speaker identification features, all other features are focused on the resolution of nominal mentions. The Stanford rule-based system considers the following features in order to resolve a pronoun to a candidate antecedent:

- Number, gender, person, animacy and named entity labels: a pronominal anaphor should agree with its antecedent based on the number, gender, person and animacy attributes. They should also have compatible named entity labels. The named entity labels are acquired using the Stanford named entity recognition tool. If the value of an attribute is not available, the value is set to "unknown". The "unknown" value matches any other value.
- Distance: the number of sentences between the candidate antecedent and the pronoun should not be larger than three.

for instance, based on this feature the "he" pronoun in Example 2.34 is resolved to "Wang Jin - pyng".

(2.34) [Wang Jin - pyng] says that [he] has decided not to continue serving as KMT vice chairman.

2.4.2 Berkeley Coreference Resolver

The Berkeley coreference resolver that is introduced by Durrett & Klein (2013) is a learning-based coreference resolver that uses a mention-ranking resolution method.

The learning loss function of Durrett & Klein (2013) is then as follows:

$$l(w) = \sum_{i=1}^{n} \log(\sum_{a_k \in A(m_i)} \exp(\triangle(a_k, m_i)) score(a_k|m_i)) + \lambda ||w||_1 \tag{2.35}$$

where $\triangle$ is a mistake-specific cost function. $\triangle$ assigns different costs to different types of errors, i.e. selecting an antecedent for a non-anaphoric mention, not selecting an antecedent for an anaphoric mention, and selecting a wrong antecedent for an anaphoric mention.

$score(a_k|m_i)$ is computed as follows:

$$score(a_k|m_i) = \exp(\sum_{f_j} w_j f_j(a_k, m_i)) \tag{2.36}$$

where f_j is a feature function that could describe the properties of anaphor, antecedent or their pairwise relation. If $a_k = \emptyset$, only features that examine the properties of m_i are used.

Unlike the Stanford rule-based system that only includes heuristic linguistic features, Durrett & Klein (2013) use a simple feature set that mainly includes lexical features.

The Berkeley coreference resolver has two different feature sets, namely *surface* and *final*. The *surface* feature set includes the following set of lexical features for each mention, i.e. either anaphor or antecedent:

- Mention type: proper name, nominal, or pronominal
- Mention string: complete string of a mention
- Head, first and last words: head, first and last word of a mention
- Preceding and following words: two immediately preceding and following words of a mention

The set of non-lexical features that is included in the *surface* feature set is as follows:

- Mention length: number of included words in the mention
- Distance: distance between two mentions based on the number of sentences or mentions
- Exact string match: strings of anaphor and antecedent match
- Head match: heads of anaphor and antecedent match

Apart from the above set of lexical and non-lexical features, two conjunction forms of each feature are also included: (1) the conjunction of the mention type of anaphor and each feature, and (2) the conjunction of the type of anaphor and the type of antecedent and each feature.

Durrett & Klein (2013) extend the *surface* feature set to *final* by adding the following features:

- Speaker information: the speaker of each mention
- Gender and number: the gender and number information of each mention

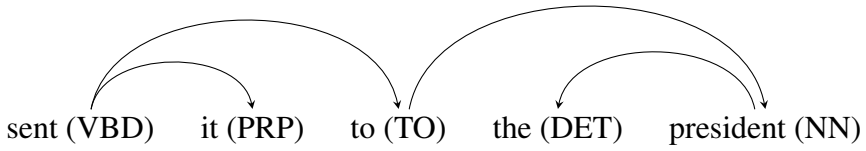


Figure 2.3: The dependency parse tree of "sent it to the president". The POS tag of each word is specified in the parentheses. The example is taken from Durrett & Klein (2013).

- Nested mentions: whether two mention are nested, e.g. "Assurances Generales de France" is nested in "The state-controlled insurer Assurances Generales de France".
- Mention head ancestry: the dependency path from the mention head to its grandparent including the POS tags of intermediate nodes and arc directions, e.g. the mention head ancestry of the mention "the president" from the example of Figure 2.3 is "president $\xleftarrow{Right}$TO$\xleftarrow{Right}$VBD".

2.4.3 cort

cort is a neural coreference resolver that is introduced by Martschat & Strube (2015). cort includes various resolution models including mention-pair, mention-ranking and antecedent tree[11]. The mention-ranking model is the best performing model of cort.

During learning and for each anaphor, Martschat & Strube (2015) choose the best scored antecedent under the current model. They use a variation of Durrett & Klein (2013)'s mistake-specific cost function. Martschat & Strube (2015) use a structured latent perceptron (Sun et al., 2009) to learn the parameters of the model.

cort includes lexical features and also a considerable number of non-lexical features. The lexical feature set of cort, i.e. *lexical*, includes:

- Head, first and last words
- Preceding and following words of a mention
- Governor: the word of the syntactic parent of a mention

The non-lexical features that are used to describe each mention, i.e. *mention-based-non-lexical*, includes:

[11]The antecedent tree model is a natural extension of the ranking model in which all antecedent decisions for all anaphors are made together.

- Mention type
- Gender and number information
- Semantic class: the semantic class of a mention could be one of the following classes: person, object, numeric and unknown. cort uses WordNet in order to compute the semantic class information
- Dependency relation: the dependency relation of the mention head to its parent
- Named entity tag: named entity tag (NER) of the mention head. The value is "none" for mentions that are not named entities
- Mention length: the number of words in a mention
- Mention ancestry: similar to the ancestry feature in the Berkeley coreference resolver without including the head word itself, e.g. "president $\xleftarrow{Right}$TO$\xleftarrow{Right}$VBD" for "the president" in Example 2.3.

The set of non-lexical pairwise features in cort, i.e. *pairwise-non-lexical*, includes (1) exact match, (2) head match, (3) same speaker, (4) acronym, (5) nested mentions, (6) string of one mention is contained in the other, (7) the head of one mention is contained in the other, (8) distance between two mentions based on the number of sentences and words.

cort uses a single layer neural network with no hidden layers to combine various input features. Therefore, the feature combination is performed manually and in a heuristic way in cort. cort creates additional features by combining basic features as follows: (1) Combining corresponding *lexical* and *mention-based-non-lexical* features of anaphor and antecedent, e.g. combining the first word of the anaphor with the first word of the antecedent as a new feature. Lets call these combinatorial features *anaphor-antecedent-combinatorial*. (2) Combining the mention type of anaphor and all *lexical*, *mention-based-non-lexical*, *pairwise-non-lexical* and *anaphor-antecedent-combinatorial* features. (3) Combining the type of anaphor and the type of antecedent with the *lexical*, *mention-based-non-lexical*, *pairwise-non-lexical* and *anaphor-antecedent-combinatorial* features. The last two combinatorial features are inspired by Durrett & Klein (2013).

For instance, the set of non-combinatorial features for describing the pair (Disney, it) in Example 2.37 is as follows:

(2.37) The most important thing about [*Disney*] is that [*it*] is a global brand.

- *lexical* features for antecedent: head=Disney, first=Disney, last=Disney, preceding=about, following=is, governor=thing

- *lexical* features for anaphor: head=it, first=it, last=it, preceding=that, following=is, governor=brand
- *mention-based-non-lexical* features for antecedent: type=proper, gender=neutral, number=single, semantic-class=object, dependency-relation=nmod, NER=organization, length=1, ancestry=VBZ $\xrightarrow{R}$ NN $\xrightarrow{R}$
- *mention-based-non-lexical* features for anaphor: type=it[12], gender=neutral, number=single, semantic-class=object, dependency-relation=nsubj, NER=none, length=1, ancestry= VBZ $\xrightarrow{R}$ NN $\xrightarrow{L}$
- *pairwise-non-lexical* features: exact-match=false, head-match=false, same-speaker=true[13], acronym=false, nested=false, string-contained=false, head-contained=false, sentence-distance=0, word-distance=2

2.4.4 deep-coref

deep-coref is a deep neural model that is introduced by Clark & Manning (2016a). deep-coref includes various resolution models including mention-pair, top-pairs, mention-ranking, and entity-based models.

The mention-ranking model has the best reported results among various deep-coref models.

For the mention-ranking model, Clark & Manning (2016a) use the training objective proposed by Wiseman et al. (2015) that encourages separating the highest scoring true antecedent and incorrect antecedents.

The loss function of mention-ranking introduced by Wiseman et al. (2015) is as follows:

$$L(\theta) = \sum_{i=1}^{n} \max_{a \in A(m_i)} \triangle(a, m_i)(1 + score(a, m_i) - score(\hat{t}_i, m_i)) + \lambda||\theta||_1 \tag{2.38}$$

where $\hat{t}_i = \arg\max_{t \in T(m_i)} score(t, m_i)$. As before, $A(m_i)$ is the set of all candidate antecedents of m_i, i.e. all mentions preceding m_i and $\emptyset$ for non-anaphoric mentions, $T(m_i)$ is the set of true candidate antecedents of m_i, and $\triangle(a, m_i)$ is a mistake-specific cost function.

The mention-ranking model of Clark & Manning (2016a) uses a simpler ranking model, which is called top-pairs model, and also a mention-pair model for pretraining. The *top-pairs* model is first introduced by Clark & Manning (2015). It only processes the highest and lowest scoring antecedents for each mention by a probabilistic loss function:

$$-\sum_{i=1}^{n}[\max_{t \in T(m_i)} \log score(t, m_i) + \min_{f \in A(m_i) - T(m_i)} \log(1 - score(f, m_i))] \tag{2.39}$$

[12]For pronouns, the mention type has one of the following values: 'i', 'you', 'he', 'she', 'it', 'we', 'they'

[13]In the CoNLL development set all the words in this sentence have the same speaker.

deep-coref also has a variation of the mention-ranking model in which the best hyper-parameter settings for the loss function is set in a reinforcement learning framework (Sutton & Barto, 1998). Error penalties for various kinds of errors in the mistake-specific cost function are examples of hyper-parameters that need to be set manually in the original mention-ranking model.

In order to pose the model in the reinforcement learning framework, Clark & Manning (2016a) consider the mention-ranking model as an agent that takes a series of actions for resolving all mentions of the document. Each of the actions links a mention to a candidate antecedent, which can also be the dummy antecedent.

After the agent performs a sequence of actions, it receives a reward. Clark & Manning (2016a) define the reward function based on a standard coreference evaluation metric, i.e. B^3. This way, the model parameters can be directly optimized based on coreference evaluation metrics.

deep-coref mainly uses lexical features and it incorporates word embeddings instead of words. deep-coref incorporates the word embeddings of the head, first and last words, two preceding and two followings words and the governor of each mention. It also uses a set of averaged word embeddings including the average of the embeddings of:

- all mention words
- five preceding words
- five following words
- all words in the mention sentence
- all words in the mention document

Apart from the above set of lexical features, it also includes the following set of non-lexical features for mentions or mention-pairs: (1) mention type, (2) mention length, (3) mention position: the relative position of the mention in the document, i.e. index of the mention divided by the number of all mentions in the document, (4) nested mentions, (5) speaker match, (6) the string of one mention is the speaker of the other mention, (7) exact match, (8) refined head match: two mentions have the same entity type and the head of one mention is included in the tokens of the other one, (9) relaxed string match: if there is a word with one of the "NN", "NNS", "NNP", "NNPS" POS tags in one mention, it should also exist in the other mention, (10) distance of two mentions based on the number of sentences or mentions, and (11) genre: the genre of the document in which mentions are positioned. The value is 'none' if the document is not from the CoNLL dataset.

deep-coref uses three hidden layers on top of input features. As a result, there is no need for manual feature combination in deep-coref as it was the case for the Berkeley coreference resolver and cort.

2.4.5 e2e-coref

e2e-coref is yet another coreference resolver that is based on the mention-ranking model. e2e-coref is developed by Lee et al. (2017), and it has the best reported performance on the CoNLL 2012 test set.

Lee et al. (2017) use a cost-insensitive variation of Durrett & Klein (2013)'s mention-ranking objective function. They mention that they also experiment with more complex variations of the ranking model, i.e. the cost-sensitive and margin-based variations used by Wiseman et al. (2015) and Clark & Manning (2016a). However, the cost-insensitive maximum-likelihood objective performs better in their experiments. The marginal log-likelihood objective function of Lee et al. (2017) is as follows:

$$\log \prod_{i=1}^{n} \sum_{a_k \in \hat{T}(m_i)} Pr(a_k|m_i) \tag{2.40}$$

where $\hat{T}(m_i)$ are the set of true antecedents of m_i that appear before m_i in the text. $Pr(a_k|m_i)$ is defined as follows:

$$Pr(a_k|m_i) = \frac{exp(score(a_k, m_i))}{\sum_{a \in A(m_i)} exp(score(a, m_i))} \tag{2.41}$$

e2e-coref is an end-to-end coreference resolver. It does not use any mention detection and it performs mention detection and coreference resolution jointly. Therefore, e2e-coref considers all possible spans of the text as candidate mentions. As a result, in order to compute a score for two candidate spans to be coreferent, it considers the scores of the given spans to be mentions as well as the score of the two spans to be in a coreference relation:

$$score(a_k, m_i) = \begin{cases} score_m(a_k) + score_m(m_i) + score_p(a_k, m_i) & \text{if } a_k \neq \emptyset \\ 0 & \text{if } a_k = \emptyset \end{cases} \tag{2.42}$$

where $score_m$ is a mention scoring function and $score_p(a_k, m_i)$ is a pairwise scoring function for determining the likelihood of a_k to be the antecedent of m_i. By learning both $score_m$ and $score_p$ functions, e2e-coref learns mention detection and coreference resolution jointly.

$score_m$ does not learn to detect any mention, it learns to detect coreferring mentions. As a result, mention detection, coreferring mention detection, and coreference resolution are all learned jointly, while each of these three steps is learned separately in previous systems. The detection of coreferring mentions is an optional preprocessing step in previous coreference

resolvers and it has a big impact on the overall performance in in-domain evaluations (Moosavi & Strube, 2016a).

In order to maintain computation efficiency, Lee et al. (2017) prune candidate spans both during training and testing. They prune candidate spans that are longer than a predefined threshold. Besides, from each document of length $|D|$, they only keep up to $\alpha|D|$ highest scoring candidate spans and for each mention, they only consider up to K candidate antecedents.

Lee et al. (2017) mainly use lexical features for learning coreference relations. They use a bidirectional LSTM (Hochreiter & Schmidhuber, 1997) for learning the mention representations and the representation of their corresponding context from word embeddings. An independent bidirectional LSTM is used for each sentence, i.e. they did not find cross-sentence context useful in their experiments. Each word of the sentence is then encoded using the LSTM.

Unlike previous coreference resolvers, in which mention heads are determined using the syntactic parses and heuristic rules, Lee et al. (2017) determine mention heads using an attention mechanism (Bahdanau et al., 2014) over including words in each mention. By using an attention mechanism, they compute each mention head as a weighted function of mention word representations. The weights are learned automatically during training. Each mention is then presented by concatenating the representations of its start and end words and its estimated head. Apart from above features that are based on word embeddings, they also incorporate the length of each mention.

The dot product of the mention representations of the candidate antecedent and anaphor is then computed as a similarity vector of the mention pair. Lee et al. (2017) also incorporate the following features for enriching the description of pairwise relations: speaker match, the distance of two mentions, and the genre of the document. A pair of mentions is then represented by concatenating the representations of antecedent and anaphor, the similarity vector, and the embedding learned from the above non-lexical features.

Apart from the above model that uses a *single* classifier, Lee et al. (2017) use an *ensemble* of five single models with various random initializations. At test time, they prune spans based on the averaged value of the mention scores of all models. This pruning is used for all models. The estimated antecedent scores of each model are then averaged to compute the final scores.

Part II

A Robust Framework for Coreference Evaluation

Chapter 3

Robust Evaluation Metric

The first step for having a robust evaluation framework is to have a reliable evaluation metric. The disagreement in coreference resolution is not limited to the definition of coreference relations. Several evaluation metrics have been introduced for coreference resolution (Vilain et al., 1995; Bagga & Baldwin, 1998; Luo, 2005; Recasens & Hovy, 2011; Tuggener, 2014). The experimental results in the coreference resolution literature are reported using three or more different evaluation metrics. Metrics that are commonly reported are *MUC* (Vilain et al., 1995), B^3 (Bagga & Baldwin, 1998), *CEAF* (Luo, 2005), and *BLANC* (Recasens & Hovy, 2011). The reasons for having and reporting multiple metrics include: (1) There are known flaws for each of the existing metrics, (2) the agreement between all these metrics is relatively low (Holen, 2013), and (3) it is not clear which metric is the most reliable.

In order to have a single point of comparison, the CoNLL-2011/2012 shared tasks (Pradhan et al., 2011; 2012) start using an *average* of three metrics, i.e. *MUC*, B^3, and *CEAF*, following a proposal by Denis & Baldridge (2009b), for comparing and ranking participating systems. However, averaging individual metrics is nothing but a compromise. One should not expect to get a reliable score by averaging three unreliable ones. Furthermore, when an average score is used for comparisons, it is not possible to analyze recall and precision to determine which output is more precise and which one covers more coreference information. This is a requirement for coreference resolvers to be used in end-tasks.

In this chapter, we first review the common evaluation metrics that are used in the coreference literature. Then we explain why we cannot trust the existing evaluation metrics and need a new one. At the end, we propose *LEA*, a Link-based Entity Aware evaluation metric, that overcomes the known problems of the existing metrics.

3.1 Current Evaluation Metrics

Current systematic evaluation metrics represent entities either as a set of links or as a set of mentions. According to the selected entity representation, current evaluation metrics essentially boil down to two different categories: (1) link-based metrics and (2) mention-based metrics. *MUC* and *BLANC* are link-based, and B^3, $CEAF_m$ and $CEAF_e$ are mention-based.

As mentioned by Luo (2005), interpretability and discriminative power are two basic requirements for a reasonable evaluation metric. In regard to the *interpretability* requirement a high score should indicate that the vast majority of coreference relations and entities are detected correctly. Similarly, a system that resolves none of the coreference relations or entities should get a zero score. An evaluation metric should also be *discriminative*. It should be able to distinguish between outputs that are obviously different.

3.1.1 Notation

In coreference evaluation, we have a set of gold entities and a set of response entities. Gold entities are commonly referred to as key entities in the coreference literature. We use these two terms interchangeably. Response entities are those entities that are generated by a coreference resolver. In what follows, $K = \{k_1, \ldots, k_i\}$ is the set of key entities and $R = \{r_i, \ldots, r_j\}$ is the set of response entities.

3.1.2 Link-Based Metrics

Link-based metrics represent entities by the links between mentions. The *MUC* and *BLANC* metrics lie in this category. The difference between *MUC* and *BLANC* is that *MUC* only uses coreference links to represent an entity while *BLANC* uses both coreferent and non-coreferent links.

3.1.2.1 MUC

MUC is the first systematic coreference evaluation metric. It was proposed by Vilain et al. (1995) as part of the MUC-6 and MUC-7 coreference resolution evaluation tasks. *MUC* evaluates coreference output based on the number of missing or extra links in the output in comparison to the gold entities. *MUC* only considers coreferent links to represent entities. Besides, it only considers the minimum number of coreferent links that are required to connect the mentions of each entity. Figure 3.1 depicts the entity representation of the *MUC* metric.

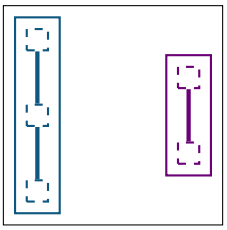


Figure 3.1: *MUC* entity representation.

The *MUC* metric computes recall based on the minimum number of missing links in the response entities in comparison to the gold entities:

$$\text{Recall} = \frac{\sum_{k_i \in K}(|k_i| - |p(k_i)|)}{\sum_{k_i \in K}(|k_i| - 1)}$$

where $p(k_i)$ is a partition of k_i relative to R. $p(k_i)$ is created by intersecting k_i with the response entities that overlap k_i. For example, if $k_1 = \{m_1, m_2, m_3, m_4\}$ and $R = \{\{m_1, m_2\}\}$, then $p(k_1)$ is equal to $\{\{m_1, m_2\}, \{m_3\}, \{m_4\}\}$. $|k_i| - 1$ is the minimum number of links that are required to connect k_i's mentions. On the other hand, $|p(k_i)| - 1$ is the minimum number of missing links in the response entities that are required to connect k_i's mentions. *MUC* uses the subtraction of these two numbers as the number of correctly resolved links.

MUC precision is computed by switching the role of the key and response entities:

$$\text{Precision} = \frac{\sum_{r_i \in R}(|r_i| - |p(r_i)|)}{\sum_{r_i \in R}(|r_i| - 1)}$$

3.1.2.2 BLANC

Recasens & Hovy (2011) propose another link-based metric for coreference resolution named *BLANC*, BiLateral Assessment of Noun-phrase Coreference. *BLANC* was originally proposed for evaluating coreference on gold mentions. Luo et al. (2014) extend *BLANC* to be used on system mentions as well.

Unlike *MUC*, instead of only considering the minimum number of coreference links, *BLANC* represents each entity by all of its coreferent and non-coreferent links. Figure 3.2 depicts the entity representation by the *BLANC* metric.

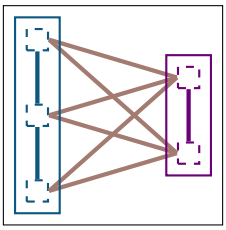


Figure 3.2: *BLANC* entity representation.

BLANC adapts the Rand index (Rand, 1971) to coreference resolution evaluation. The Rand index is a general evaluation metric for clustering. It measures the similarity of two clusters by examining how each pair of data points is assigned in the given clustering. Recasens & Hovy (2011) rephrase the motivations behind the Rand index in coreference terms as: (1) every mention is unambiguously assigned to a specific entity; (2) entities are defined as much by those mentions that do not belong to them as by those mentions which they include; and (3) all mentions are of equal importance in determining the set of entities.

Assume C_k and C_r are the sets of coreference links in the key and response entities, respectively. Let N_k and N_r be the sets of non-coreference links in the key and response entities, respectively. Recall for coreferent and non-coreferent links is computed as:

$$R_c = \frac{|C_k \cap C_r|}{|C_k|}, \quad R_n = \frac{|N_k \cap N_r|}{|N_k|}$$

Similarly, precision of coreferent and non-coreferent links is computed as:

$$P_c = \frac{|C_k \cap C_r|}{|C_r|}, \quad P_n = \frac{|N_k \cap N_r|}{|N_r|}$$

BLANC recall and precision are then computed by averaging the recall and precision of coreference and non-coreference links:

$$\text{Recall} = \frac{R_c + R_n}{2}$$

$$\text{Precision} = \frac{P_c + P_n}{2}$$

Boundary cases: when the denominator of either of R_c, R_n, P_c or P_n is zero, special care needs to be taken for the *BLANC* computations:

1. If $C_k = C_r = \emptyset$ and $N_k = N_r = \emptyset$ then *BLANC*=$I(\{m_k : \exists k_i \subset K | m_k \in k_i\} = \{m_r : \exists r_j \subset R | m_r \in r_j\})$. $I(.)$ is an indicator function that returns one if its argument is true, and returns zero otherwise. If a document has only one mention without any links, this boundary case can happen.

2. If $C_k = C_r = \emptyset$ and $|N_k| + |N_r| > 0$ then *BLANC*=$F_n = \frac{2R_nP_n}{R_n+P_n}$. This case happens when the key and response entities only contain singletons, and therefore there are no coreferent links.

3. If $N_k = N_r = \emptyset$ and $|C_k| + |C_r| > 0$ then *BLANC*=$F_c = \frac{2R_cP_c}{R_c+P_c}$. This case happens when the key and response entities contain only one entity, and therefore there are no non-coreferent links.

3.1.3 Mention-based Metrics

There is another category of evaluation metrics that represent entities by their mentions instead of their links. B^3 is a mention-based metric. Another metric that is discussed in this section is the *CEAF* metric. The entity representation of the *CEAF* metric is flexible and depends on its similarity metric. All the similarity metrics proposed by Luo (2005) are mention-based. Therefore, we classify *CEAF* as a mention-based metric. Nevertheless, it is important to keep in mind that *CEAF* can also be a link-based metric if it uses a link-based similarity metric.

Besides the above evaluation metrics, Tuggener (2014) also introduces a mention-based metric. Tuggener (2014) defines recall as $\frac{tp}{tp+wl+fn}$ and precision as $\frac{tp}{tp+wl+fp}$. tp is the number of mentions that are correctly resolved, fp is the number of mentions that are non-coreferent but recognized as coreferent in the system output, fn is the number of coreferent mentions that are detected as non-coreferent by the coreference resolver, and finally wl is the number of coreferent mentions that are linked to an incorrect antecedent.

Tuggener (2014)'s metric considers coreference resolution as classifying individual mentions instead of a clustering task, i.e. entity structures are not considered in this metric. Besides, Tuggener (2014)'s metric is designed to evaluate coreference outputs for higher-level applications. We do not include this metric in further analysis.

3.1.3.1 B^3

The B^3 metric is proposed by Bagga & Baldwin (1998). Figure 3.3 depicts the entity representation of B^3.

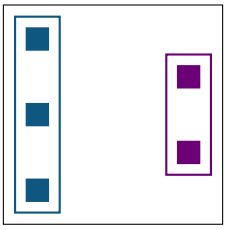


Figure 3.3: B^3 entity representation.

B^3 computes the overall recall and precision based on the recall and precision of the individual mentions. For each mention m in the key entities, B^3 recall considers the fraction of the correct mentions that are included in the response entity that contains m. Assume mention m is in entity k_i. The recall of m is computed as:

$$\text{Recall}_m = \sum_{r_j \in R} \frac{|k_i \cap r_j|}{|k_i|} \tag{3.1}$$

The overall B^3 recall is then computed by averaging the recall of all mentions in the key set:

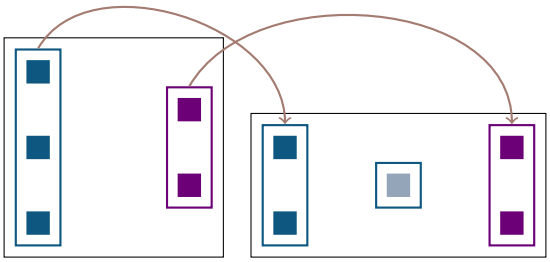
$$\text{Recall} = \frac{\sum_{k_i \in K} \sum_{r_j \in R} \frac{|k_i \cap r_j|^2}{|k_i|}}{\sum_{k_i \in K} |k_i|}$$

Similar to MUC, B^3 precision is computed by switching the role of the key and response entities:

$$\text{Precision} = \frac{\sum_{r_i \in R} \sum_{k_j \in K} \frac{|r_i \cap k_j|^2}{|r_i|}}{\sum_{r_i \in R} |r_i|}$$

3.1.3.2 CEAF

The *CEAF* metric is proposed by Luo (2005). *CEAF*'s main assumption is that each key entity should be mapped to only one response entity, and vice versa. *CEAF* then computes recall and precision by measuring the similarity of the mapped entities. *CEAF* finds the best one-to-one mapping of the key to the response entities (g^*) using the given similarity measure. Figure 3.4 depicts the *CEAF* one-to-one mapping.



(a) (b)

Figure 3.4: *CEAF* one-to-one mapping.

Let $\phi(i, j)$ be the similarity between the ith key entity and the jth system entity, and x_{ij} be the binary variable that represents that the ith key entity is mapped to the jth system entity. The goal is to find the optimal one-to-one mapping that maximizes $\sum_{i,j} \phi(i, j)x_{ij}$. Luo (2005) uses the Kuhn-Munkres algorithm (Kuhn, 1955; Munkres, 1957) to find this optimal mapping.

Assuming K^* is the set of key entities that is included in the optimal mapping, i.e. g^*, recall is computed as:

$$\text{Recall} = \frac{\sum_{k_i \in K^*} \phi(k_i, g^*(k_i))}{\sum_{k_i \in K} \phi(k_i, k_i)} \tag{3.2}$$

For computing *CEAF* precision, the nominator of Equation 3.3, i.e. the similarity value of the mapped key-response entities in the optimal mapping, remains the same and the denominator, which is the similarity of the key entities to themselves, changes according to response entities and therefore becomes the similarity of the response entities to themselves:

$$\text{Precision} = \frac{\sum_{k_i \in K^*} \phi(k_i, g^*(k_i))}{\sum_{R_i \in R} \phi(r_i, r_i)} \tag{3.3}$$

Luo (2005) discusses four different variants for the similarity measure:

$$\phi_1(K, R) = \begin{cases} 1 & \text{if } K = R \\ 0 & \text{otherwise} \end{cases}, \quad \phi_2(K, R) = \begin{cases} 1 & \text{if } K \cap R \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_3(K, R) = |K \cap R|, \quad \phi_4(K, R) = \frac{2|K \cap R|}{|K| + |R|}$$

ϕ_1 and ϕ_2 are two extreme similarity measures. ϕ_1 considers two entities similar if all mentions in those two entities are identical. On the other hand, ϕ_2 considers two entities similar if they have at least one mention in common.

ϕ_3 and ϕ_4 are more practical similarity measures. Instead of considering the similarity of two entities as a binary variable, ϕ_3 and ϕ_4 compute the similarity as a continues value between zero and one. The zero value indicates that entities are not similar at all and the one value indicates that the entities are the same.

Since ϕ_3 computes the similarity as the number of common mentions between two entities, the variant of *CEAF* that uses ϕ_3 is called mention-based *CEAF*, i.e. $\textit{CEAF}_m$. When ϕ_4 is used, the denominators of *CEAF*'s recall and precision equal to the number of entities in the key and system output, respectively. Therefore, the variant of *CEAF* that uses ϕ_4 is called entity-based *CEAF*, i.e. $\textit{CEAF}_e$.

3.2 Why Do We Need a New Evaluation Metric?

There are known drawbacks for each of the existing evaluation metrics. Therefore instead of relying on a single metric for evaluating coreference improvements, following a proposal by Denis & Baldridge (2009b) and since the CoNLL-2011/2012 shared tasks (Pradhan et al., 2011; 2012), an *average* of three metrics, i.e. *MUC*, B^3, and $\textit{CEAF}_e$, is used. The average of the F_1 values of the *MUC*, B^3, and $\textit{CEAF}_e$ metrics is called the *CoNLL* score. Currently,

coreference resolution improvements are mainly assessed by considering the improvements in the *CoNLL* score. However, averaging three unreliable scores is not a solution for creating a reliable one, especially when the agreement between the averaged metrics is relatively low (Holen, 2013).

Besides, when an average score is used for assessments, it is not possible to analyze recall and precision to determine which output is more precise and which one covers more coreference information. It is also important for the corresponding recall and precision values to be interpretable and discriminative. Depending on the end-task that uses coreference resolvers, recall or precision may be of various importance. For example, as Stuckhardt (2003) argues, a coreference resolver needs high precision to meet the specific requirements of text summarization and question answering.

In the following sections, we discuss the drawbacks of the current evaluation metrics.

3.2.1 Bias Towards Including More Gold Mentions, Even in a Wrong Entity: B^3, CEAF, BLANC

Mention detection is an important preprocessing step for coreference resolution. However, we show that besides the implicit effect that mention identification has on coreference resolution, it could also explicitly influence B^3, *CEAF* and *BLANC* scores. We show that these metrics reward the inclusion of gold mentions in the output entities, even if they are in completely wrong entities. The mention identification bias affects the recall and precision of the B^3, *CEAF* and *BLANC* metrics so that they cannot be interpretable or reliable.

	MUC			B^3			CEAF_m			CEAF_e			BLANC		
	R	P	F_1	R	P	F_1	R	P	F_1	R	P	F_1	R	P	F_1
deep-coref															
Base	69.36	80.75	74.62	57.15	71.62	63.57	62.18	72.94	67.14	53.87	64.82	58.84	58.56	72.49	64.64
More precise	69.36	86.34	76.92	55.57	75.10	63.88	60.34	74.73	66.77	50.84	61.87	55.82	55.40	73.49	62.92
Less precise	69.36	73.28	71.26	61.15	68.67	64.69	66.99	67.84	67.41	63.81	56.72	60.06	69.57	73.14	71.26
nn-coref															
Base	69.75	77.49	73.42	56.95	66.83	61.50	61.89	69.35	65.40	53.85	62.14	57.70	58.94	67.63	62.92
More precise	69.75	82.95	75.78	55.24	69.89	61.71	59.80	70.83	64.85	50.32	58.79	54.23	55.50	68.37	61.12
Less precise	69.75	71.04	70.39	60.67	64.75	62.64	66.29	65.10	65.69	63.02	55.23	58.87	69.05	68.42	68.66

Table 3.1: Mention identification effect on the CoNLL-2012 development set.

In order to show how the mention identification bias could affect the mentioned metrics, we choose the output of two state-of-the-art coreference resolvers, i.e. the ranking models of deep-coref (Clark & Manning, 2016a; 2016b) and nn-coref (Wiseman et al., 2016). The *Base* rows in Table 3.1 represent the performance of these two systems on the CoNLL-2012 test set. For both systems, we perform two different experiments; (1) making the outputs more precise, and (2) making the outputs less precise.

Let $M_{k,r}$ be the set of common mentions between the key and response entities. Assume $C_k(m)$ and $C_r(m)$ are the sets of coreference links of mention m in the key and response entities, respectively. Mention m is incorrectly resolved if $m \in M_{k,r}$ and $L_k(m) \cap L_r(m) = \emptyset$, i.e. m is a coreferent mention that exists in both key and response entities, however, none of it's coreferent links in the response entities are correct.

If we remove the incorrectly resolved mentions, the response entities will become more precise. The precision improves because the response entities contain less incorrect coreference relations. Besides, the recall does not change because no correct coreference relations or entities are added to or removed from the response entities.

We make more precise outputs from those of *deep-coref* and *nn-coref* by removing 889 and 951 incorrectly resolved mentions, respectively. *More precise* rows in Table 3.1 show the scores for these more precise outputs. As we can see, (1) recall changes for all the metrics except for *MUC* while we do not add or remove any correct coreference relation or entity from the output; (2) both *CEAF*$_e$ recall and precision significantly decrease; and (3) the F_1 values of *CEAF*$_m$, *CEAF*$_e$, and *BLANC* drop considerably in comparison to the *Base* results.

On the second experiment, we add completely wrong entities to the base outputs. The addition of new completely incorrect entities to the response entities should not affect the recall and it should decrease the precision.

Let $M_{d,k,\bar{r}}$ be the set of mentions of document d that exists in the key entities but is missing from the response entities. We create completely incorrect entities by linking $m_1 \in M_{d,k,\bar{r}}$ to mention $m_2 \in M_{d,k,\bar{r}}$ that is non-coreferent with m_1. This way, we add 1333 and 1244 new wrong entities of size two to the *deep-coref* and *nn-coref* outputs. These results are shown in *Less precise* rows. In this experiment, we do not add or remove any correct coreference relation or entity while we add completely wrong entities to the output. Therefore, recall should not change while precision should decrease.

As we can see, (1) except for *MUC*, recall changes for all metrics; and (2) the B^3, *CEAF*$_m$, *CEAF*$_e$ and *BLANC* scores improve in comparison to those of *Base* while the output is doubtlessly worse.

We refer to the problem that is causing these contradictory results as the **mention identification effect**.

The **mention identification effect** arises in B^3 because it uses mentions instead of coreference relations to evaluate the response entities. Therefore, if a gold mention exists in a response entity, it is considered as a resolved mention regardless of whether it has a correct coreference relation in the response entity.

Similar to B^3, the **mention identification effect** of *CEAF* is caused by using the number of common mentions between two entities, i.e. $|k_i \cap r_j|$, in ϕ_3 and ϕ_4 similarity measures. In this way, even if two mapped entities (k_i and r_j) only have one mention in common, *CEAF*$_m$

rewards recall and precision by $\frac{1}{\sum_{k_i}|k_i|}$ and $\frac{1}{\sum_{r_j}|r_j|}$, respectively. Similarly, $CEAF_e$ rewards recall and precision by $\frac{2}{(|k_i|+|r_j|)\times|K|}$ and $\frac{2}{(|k_i|+|r_j|)\times|R|}$, respectively.

Because of considering non-coreferent relations, the **mention identification effect** affects *BLANC* most strongly. When the number of gold mentions that exist in the response entities is larger, the number of detected non-coreference links will also get larger. Therefore, it results in higher values for *BLANC* recall and precision ignoring whether those gold mentions are resolved.

3.2.2 No Representation for Singletons: MUC

Since *MUC* presents each entity by the minimum number of links that are required to connect it's including mentions, it is not capable of evaluating singletons. Singletons are not annotated in several corpora including OntoNotes and MUC. However, the detection and evaluation of singletons are important in some other corpora including ACE, ARRAU, and SemEval.

3.2.3 Undiscriminating: MUC

It is not trivial to determine which evaluation metric is the most discriminative metric. However, it is easier to determine which of the current evaluation metrics are not generally capable of distinguishing between various coreference outputs. In comparison to other metrics, *MUC* is the least discriminative metric.

The *MUC* evaluation is only based on the minimum number of missing/extra links in the response compared to the key entities. For instance, *MUC* does not differentiate whether an extra link merges two singletons or the two most prominent entities of the text while the latter error does more damage than the first one. For example, consider a document that is mainly about two important entities that are each mentioned more than 100 times in the text. Assume the document also contains two insignificant entities that are each mentioned only twice in the text. According to *MUC*, there is no difference between a system that merges the two most prominent entities and a system that only merges the two insignificant entities.

3.2.4 Bias Towards Larger Entities: MUC

MUC has an incorrect preference in evaluating coreference outputs. *MUC* favors the outputs in which entities are over-merged (Luo, 2005). For instance, if we link all the key mentions of the CoNLL-2012 test set into a single response entity, the corresponding *MUC* scores will be all surprisingly very high, i.e. Recall=100, Precision=78.44 and F_1=87.91.

3.2.5 Repeated Mentions Problem: B^3

As discussed by Luo & Pradhan (2016), B^3 cannot properly handle repeated mentions in the response entities. Repeated mentions can exist in a system output. For example, if system mentions are read from a parse tree where an NP node has a single child, a pronoun, and where both nodes are considered as candidate mentions. If a gold mention is repeated in several response entities, B^3 receives credit for all the repetitions. However, it is worth noting that none of the state-of-the-art coreference resolvers contain repeated mentions in their outputs.

3.2.6 Ignoring Unmapped Correct Coreference Relations: CEAF

As mentioned by Denis & Baldridge (2009a), due to one-to-one mapping, *CEAF* ignores all correct decisions of unmapped response entities. This property could indeed lead to counterintuitive results. In order to illustrate this problem, we use a sample text from the CoNLL-2012 development set as an example (Figure 3.5). Gold mentions are enclosed in square brackets. Mentions with the same text are marked with different indices. The indices in parentheses denote to which key entity the mentions belong.

> [The American administration]$_{(1)}$ committed a fatal mistake when [it$_1$]$_{(1)}$ [executed]$_{(2)}$ [this man]$_{(3)}$, in a way for which [it$_2$]$_{(1)}$ will pay a hefty price in the near future. [[His$_1$]$_{(3)}$ survival]$_{(4)}$ would have benefited [it$_3$]$_{(1)}$ much more than [[his$_2$]$_{(3)}$ execution]$_{(2)}$ if [they$_1$]$_{(1)}$ understood politics as [they$_2$]$_{(1)}$ should, because [[his$_3$]$_{(3)}$ survival]$_{(4)}$ could have been a card to threaten [the sectarians]$_{(5)}$ and keep [them$_1$]$_{(5)}$ as servants to [them$_1$]$_{(1)}$ and [their]$_{(1)}$ schemes. , just as [it$_4$]$_{(4)}$ could have been a bargaining chip for calming down [the resistance]$_{(6)}$ and the Iraqi sector supporting [it$_5$]$_{(6)}$. But [they]$_2$ do not know anything except grudges, and [they]$_2$ do not want anything other than inflicting more degradation on the Arabs and Muslims by his execution.

Figure 3.5: Sample text from the CoNLL-2012 development set.

Consider cr_1 and cr_2 in Table 3.2, which are different responses for entity (1) of Figure 3.5. cr_1 resolves many coreference relations of entity (1). However, it misses that *they*$_1$ refers to an entity which is already referred to by 'it'. Therefore cr_1 produces two entities instead of one because of this missing relation. On the other hand, cr_2 only recognizes half of the correct coreference relations of entity (1).

The results of Table 3.3 show that both *CEAF*$_m$ and *CEAF*$_e$ prefer cr_2 over cr_1 even though cr_1 makes more correct decisions. *CEAF* only selects one of the output entities of cr_1 for giving credit to the correct decisions. The other response entity is only used for penalizing the precision of cr_1. This counterintuitive result is only because of the stringent constraint of *CEAF* that the mapping of key to response entities should be one-to-one.

	Response entities
cr_1	r_1={the American administration, it$_1$, it$_2$, it$_3$} , r_2={they$_1$, they$_2$, them, their}
cr_2	r_1={the American administration, it$_1$, it$_2$, it$_3$}

Table 3.2: Different system outputs for Figure 3.5.

	MUC	B^3	$CEAF_m$	$CEAF_e$	BLANC
cr_1	92.30	66.66	50.00	44.44	60.00
cr_2	60.00	40.00	66.66	66.66	32.29

Table 3.3: F_1 scores for Table 3.2's response entities.

3.2.7 Assigning Equal Importance to All Entities: $CEAF_e$

Another problem with *CEAF*$_e$, mentioned by Stoyanov et al. (2009), is that *CEAF*$_e$ weights entities equally regardless of their sizes (or any other properties). For example, in Figure 3.5, a system that does not detect entity (1), the most prominent entity of the text, gets the same *CEAF*$_e$ score as a system which does not detect entity (4) of size 2.

3.3 LEA: Our New Evaluation Metric

In this section, we present a new evaluation metric, i.e. the Link-Based Entity-Aware metric (*LEA*). *LEA* overcomes the known shortcomings of the previous evaluation metrics.

LEA considers two factors for evaluating each entity: (1) how important the entity is in the given text and (2) how well it is resolved. *LEA* evaluates the given set of entities as follows:

$$\frac{\sum_{e_i \in E}(importance(e_i) \times resolution\text{-}score(e_i))}{\sum_{e_k \in E} importance(e_k)}$$

E is the set of key entities (K) for computing recall and it is the set of response entities (R) for computing precision.

3.3.1 Resolution Score

Similar to *MUC* and *BLANC*, *LEA* is also a link-based metric. However, instead of representing each entity by the minimum number of coreferent links or by all coreferent and non-coreferent links, *LEA* represents each entity by all of its coreferent links. Figure 3.6 depicts the *LEA* representation for each entity.

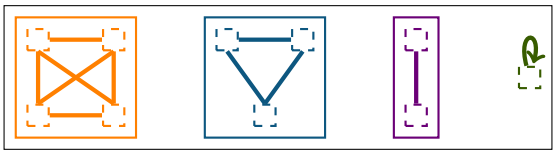


Figure 3.6: LEA entity representation.

As mentioned in Section 3.2.2, *MUC* is not capable of representing or evaluating singletons because they do not have a coreferent link. On the other hand, *BLANC* evaluates singletons by considering their non-coreferent links. *LEA* does not use coreferent or non-coreferent links for evaluating singletons. Instead, *LEA* considers a special link for singletons, namely the *self-link*. A self-link means that the corresponding mention cannot have any coreferent links. A singleton is considered as correctly resolved if and only if it is detected as a singleton, i.e. has a self-link, in the system output.

LEA computes the resolution score based on the number of correctly resolved links, i.e. coreferent links or self-links. Entity e with n mentions has $link(e) = \frac{n \times (n-1)}{2}$ unique coreferent links if $n > 1$. If e is a singleton, then e only has one link, i.e. self-link. The resolution score of key entity k_i is computed as the fraction of correctly resolved coreferent links (or self-links) of k_i:

$$resolution\text{-}score(k_i) = \sum_{r_j \in R} \frac{link(k_i \cap r_j)}{link(k_i)}$$

For each k_i, *LEA* checks all the response entities to see whether they are partial matches for k_i. r_j is a partial match for k_i, if it contains at least one of the coreferent links of k_i, i.e. at least two mentions of k_i. Thus, if a response entity only contains one mention of k_i, it is not a partial mapping of k_i. If entity k_i is a singleton, $link(k_i \cap r_j)$ is one only if r_j is a singleton and contains the same mention as k_i.

3.3.2 Importance Measure

According to the end-task or domain used, one can choose various importance measures based on different factors like the size or type of e_i, or types of mentions that are included in e_i.

Chen & Ng (2013) argue that a coreference evaluation metric should not be linguistically agnostic and mentions should be treated as linguistic objects instead of generic ones. Based on their argument, if a link contributes more to the text understanding, it should be rewarded more. They incorporate linguistic knowledge in coreference evaluation metrics so that informativeness of mentions is used for assigning different weights to various mention types. Chen & Ng (2013) propose a weighting scheme in which the highest weight is assigned to a link that contains a proper name. Similarly, a link that contains a nominal has a higher weight

than a link that only contains pronouns. Similar to Chen & Ng (2013), Holen (2013) suggests that each mention carries different information values, and considering this information could benefit the qualitative evaluation of coreference resolution.

We believe that discriminating which link or mention is more important for inferring the underlying entity or understanding the text is very subjective and highly depends on the end-task in which the coreference resolver is being used. Therefore, as the default importance measure, we consider a more general attribute, i.e. the size of an entity ($importance(e) = |e|$). In this way, the more prominent entities of the text get higher importance values. However, if one wants to incorporate linguistic knowledge into coreference evaluation metric, the $importance$ measure of *LEA* is the appropriate place.

3.3.3 Recall and Precision Definitions

Having the definitions of $resolution\text{-}score$ and the default $importance$ measure, *LEA* recall is computed as:

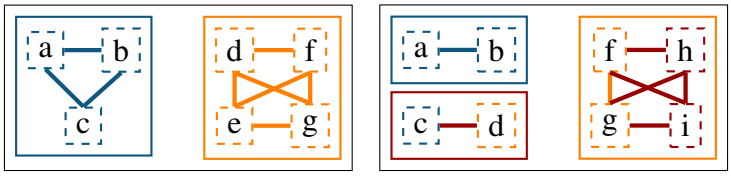
$$\text{Recall} = \frac{\sum_{k_i \in K}(|k_i| \times \sum_{r_j \in R} \frac{link(k_i \cap r_j)}{link(k_i)})}{\sum_{k_z \in K} |k_z|}$$

LEA precision is computed by switching the role of the key and response entities:

$$\text{Precision} = \frac{\sum_{r_i \in R}(|r_i| \times \sum_{k_j \in K} \frac{link(r_i \cap k_j)}{link(r_i)})}{\sum_{r_z \in R} |r_z|}$$

3.3.4 An Illustrative Example

In order to show the process of computing the *LEA* scores, we use the example from Pradhan et al. (2014) that is shown in Figure 3.7. In this example, $K = \{k_1 = \{a, b, c\}, k_2 = \{d, e, f, g\}\}$ is the set of key entities and $R = \{r_1 = \{a, b\}, r_2 = \{c, d\}, r_3 = \{f, g, h, i\}\}$ is the set of response entities.



(a) (b)

Figure 3.7: An example for LEA scores: (a) key entities and (b) response entities.

In this example, we use the default $importance$ measure that corresponds to entity size, i.e. $importance(k_1) = 3$ and $importance(k_2) = 4$. $\{ab, ac, bc\}$ and $\{de, df, dg, ef, eg, fg\}$ are

the sets of all coreferent links in k_1 and k_2, respectively. Hence, $link(k_1) = 3$ and $link(k_2) = 6$.

The only common link between k_1 and r_1 is ab. There are no common links between k_1 and the two other response entities. Similarly, k_2 has one common link with r_3, i.e. fg, and it has no common links with r_1 or r_2. As a result, $resolution\text{-}score(k_1) = \frac{1+0+0}{3}$ and $resolution\text{-}score(k_2) = \frac{0+0+1}{6}$. *LEA* recall is then computed as:

$$\frac{\sum importance(k_i) \times resolution\text{-}score(k_i)}{\sum importance(k_j)}$$
$$= \frac{3 \times \frac{1}{3} + 4 \times \frac{1}{6}}{3+4} \approx 0.24$$

For computing precision we need to change the role of key and response entities. The importance values of the response entities are $importance(r_1) = 2$, $importance(r_2) = 2$, and $importance(r_3) = 4$. r_1 has one common link with k_1 and no common link with k_2. Therefore, $resolution\text{-}score(r_1) = \frac{1}{1}$. r_2 has no common link with any of the key entities, therefore, $resolution\text{-}score(r_2) = 0$. r_3 has no common link with k_1 and one common link with k_2. Therefore, $resolution\text{-}score(r_3) = \frac{0+1}{6}$. As a result, precision is computed as:

$$\frac{2 \times \frac{1+0}{1} + 2 \times \frac{0+0}{1} + 4 \times \frac{0+1}{6}}{2+2+4} \approx 0.33$$

3.3.5 LEA in a Nutshell

In summary, *LEA* is a link-based metric with the following properties:

1. *LEA* takes into account all coreference links instead of only extra/missing links. Therefore, it has more discriminative power than MUC.

2. *LEA* evaluates resolved coreference relations instead of resolved mentions. *LEA* also does not rely on non-coreferent links in order to detect entity structures or singletons. Therefore, the **mention identification effect** does not apply to *LEA*. As a result, *LEA* recall and precision values are more reliable than those of B^3, *CEAF* and *BLANC*.

3. *LEA* allows one-to-many mappings of entities. Unlike *CEAF*, all correct coreference relations are rewarded by *LEA*. More splits (or similarly merges) of entity k_i result in a smaller $\sum_{r_j \in R} link(k_i \cap r_j)$. Therefore, splitting/merging of an entity into/with several entities will be penalized implicitly in $resolution\text{-}score$.

4. *LEA* has an adjustable importance measure that differentiates between different entities with varying importance. Therefore, unlike $CEAF_e$, it differentiates between the outputs missing the most important and the most insignificant entities.

5. *LEA* considers resolved coreference relations instead of resolved mentions. Therefore, the existence of repeated mentions in different response entities is not troublesome in *LEA* evaluations.

3.4 Analysis

In this section, we analyze *LEA* and previous evaluation metrics with regard to various coreference resolution errors. In order to perform this analysis, we consider two different settings: (1) when singletons are not included and (2) when singletons are also included in coreference evaluations.

The set of key entities in the first set of experiments contains one entity of size 20, two entities of size ten, three entities of size five, one entity of size four, and ten entities of size two. For the second set of experiments, apart from the above set of entities, we also add 30 singletons to the key entities.

3.4.1 Correct Links

We analyze different metrics based on the ratio of correctly resolved coreferent links in two different conditions: (1) There is no wrong coreferent link in the output (Figures 3.8 and 3.9), i.e. except for 100%, the F_1 measure of mention identification is lower than 100%, and (2) there are wrong coreferent links, and therefore wrong entities, in the output. In this case, gold mentions that are not connected to a correct coreferring mention are linked to one or more non-coreferent mentions (Figures 3.10, 3.11 and 3.12).

In the experiments of Figures 3.8 and 3.9, only mentions that are correctly resolved exist in the response. Therefore, F_1 of mention detection is less than 100% in all cases except for the case that the correct link ratio is 100%. In Figures 3.10, 3.11 and 3.12, apart from the mentions that are correctly resolved, other gold mentions are linked to at least one non-coreferent mention. Therefore, the F_1 value of mention detection is always 100%.

When the ratio of correctly resolved coreferent links is low, there are more ways to create incorrect entities in the experiments of Figures 3.11 and 3.12 in comparison to the experiments of Figure 3.10 . In Figure 3.11, we create small incorrect entities, i.e. including between two to six mentions. On the other hand, the size of incorrect entities in Figure 3.12 is relatively larger, i.e. between two to seventeen.

It is worth noting that for the experiments in which singletons are included, we follow *LEA*'s scheme for considering a single link for singletons for computing the number of links.

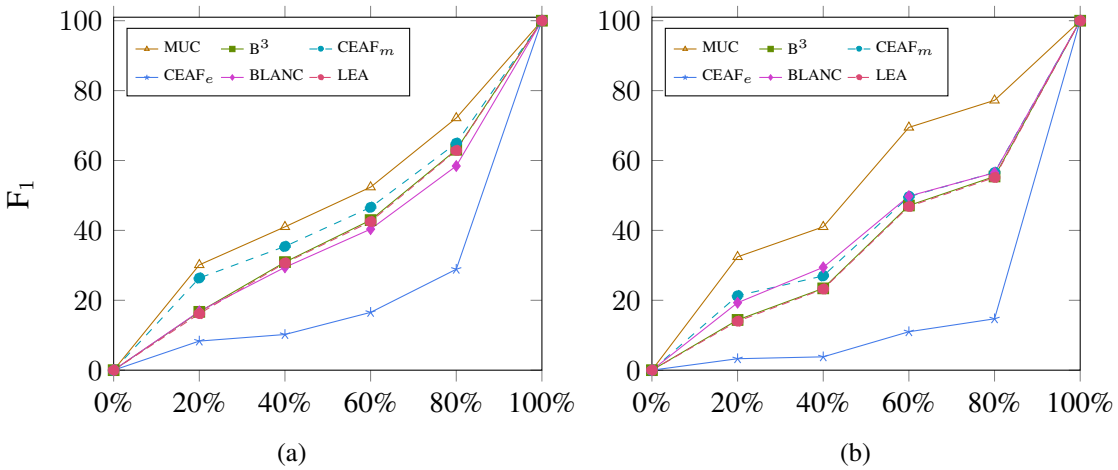


Figure 3.9: Resolved coreference links ratio without incorrect links. In these experiments, we resolve the links of larger entities first: (a) without singletons, and (b) with singletons.

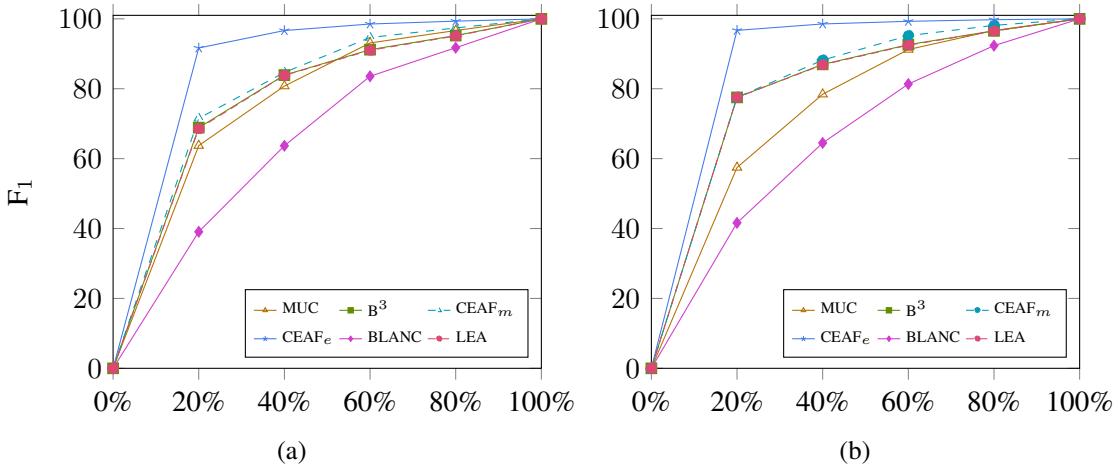


Figure 3.8: Resolved coreference links ratio without incorrect links. In these experiments, we resolve the links of smaller entities first: (a) without singletons, and (b) with singletons.

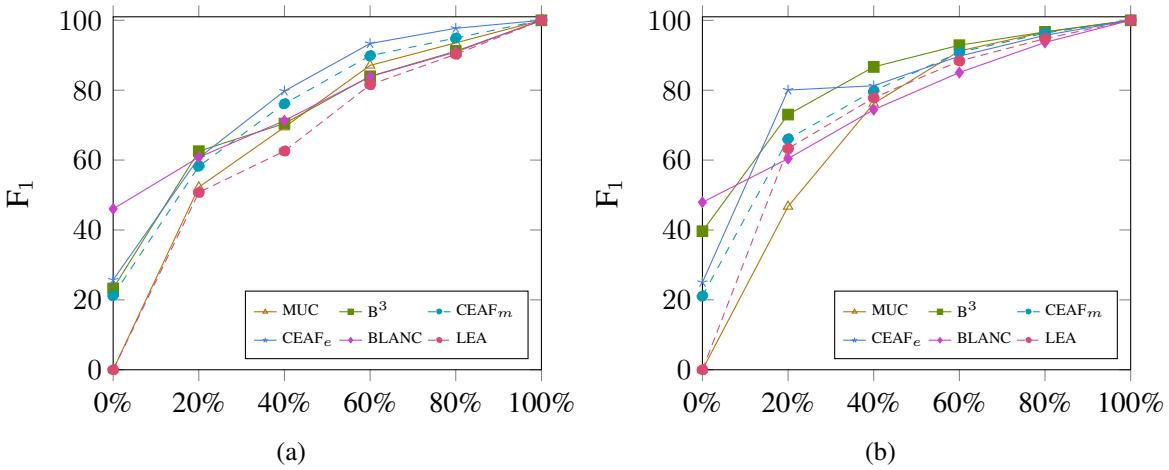


Figure 3.10: Resolved coreference links ratio in the presence of incorrect links. Links of smaller entities are resolved first: (a) without singletons, and (b) with singletons.

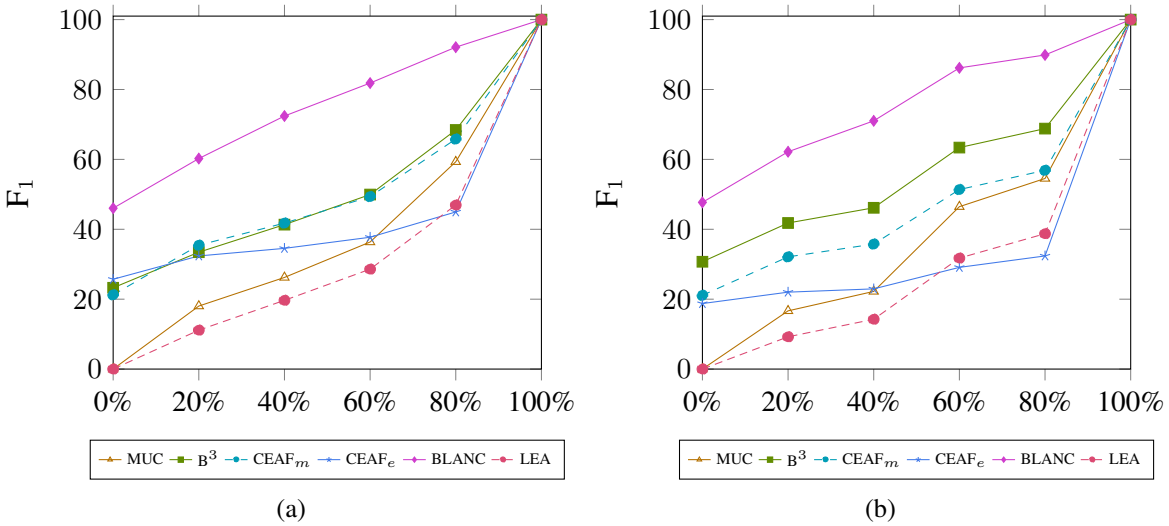


Figure 3.11: Resolved coreference links ratio in the presence of incorrect links. Links of larger entities are resolved first: (a) without singletons, and (b) with singletons. The size of incorrect entities is between two to six.

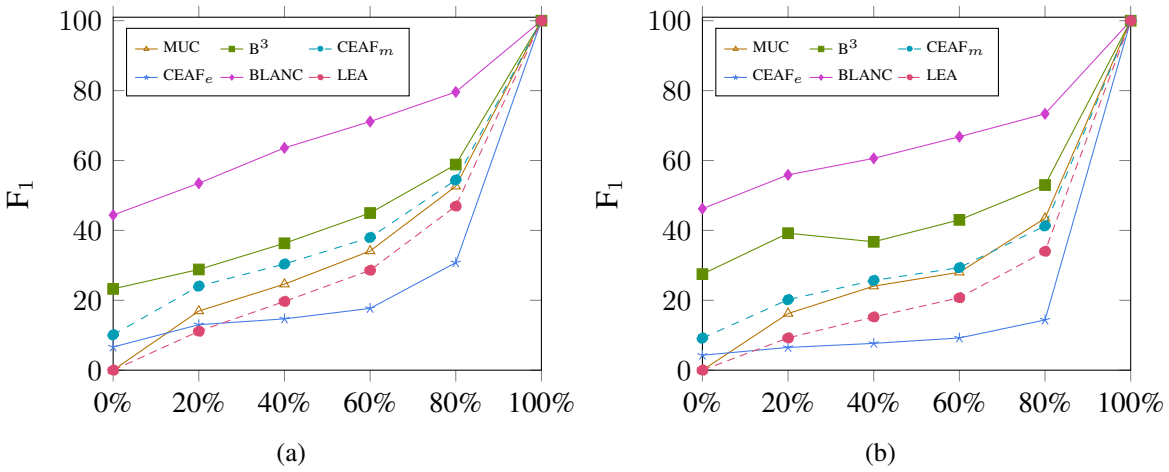


Figure 3.12: Resolved coreference links ratio in the presence of incorrect links. Links of larger entities are resolved first: (a) without singletons, and (b) with singletons. The size of incorrect entities is between two to seventeen.

The following observations can be drawn from above experiments:

1. The *CEAF*$_e$ values in Figures 3.8 and 3.9 show that *CEAF*$_e$ does not discriminate the resolution of smaller or larger entities. It mainly focuses on the number of correctly resolved entities regardless of how important or prominent they are. As a result, when the links of smaller entities are resolved first, *CEAF*$_e$ has the highest scores. On the other hand, *CEAF*$_e$ has the lowest scores when the links of larger entities are resolved first and there are no incorrect links in the output. The *CEAF*$_e$ scores for this setting are even lower than the *CEAF*$_e$ scores when the links of larger entities are resolved first and the output also includes wrong links. For example, when 20% of the links from larger entities are resolved and the output does not contain any incorrect links, the F_1 score is 8.33%, while F_1 is 32.44% when the same correct links are resolved and the output also includes many wrong links. This difference is due to the mention identification effect, which is discussed in Section 3.2.1.

2. *MUC* and *LEA* are the only measures which give a zero score to the response that contains no correct coreference relations. Non-zero values of the B^3, *CEAF* and *BLANC* metrics for this case are results of the mention identification effect.

3. If there are no incorrect links in the output, B^3 and *LEA* produce the same values.

4. If output entities contain incorrect links, the F_1 score of *BLANC* for the case in which no correct link or entity is resolved (0%) is higher than that of all the other metrics.

Similarly, if the links of larger entities are resolved first and the output contains incorrect links, the *BLANC* value is considerably higher than those of other metrics.

5. The absence or presence of singletons does not notably change the overall behavior of the examined metrics in the above experiments.

6. The rankings of the examined metrics agree on all cases except for B^3 in the experiment of Figure 3.12 with singletons (b). B^3 F_1 for the case in which 40% of coreferent links are resolved is lower than the case in which there are only 20% resolved coreference links.

3.4.2 Correct Entities

Apart from correctly resolved links, a coreference metric should also take into account the ratio of correctly resolved entities. In this section, we analyze the coreference resolution metrics based on the number and the size of correctly resolved entities. In these experiments, each entity is either resolved completely, or all of its mentions are absent from the response. In Figure 3.13, the key entities are added to the response in decreasing order of their size. Figure 3.14 shows the experiments in which the entities are resolved in increasing order.

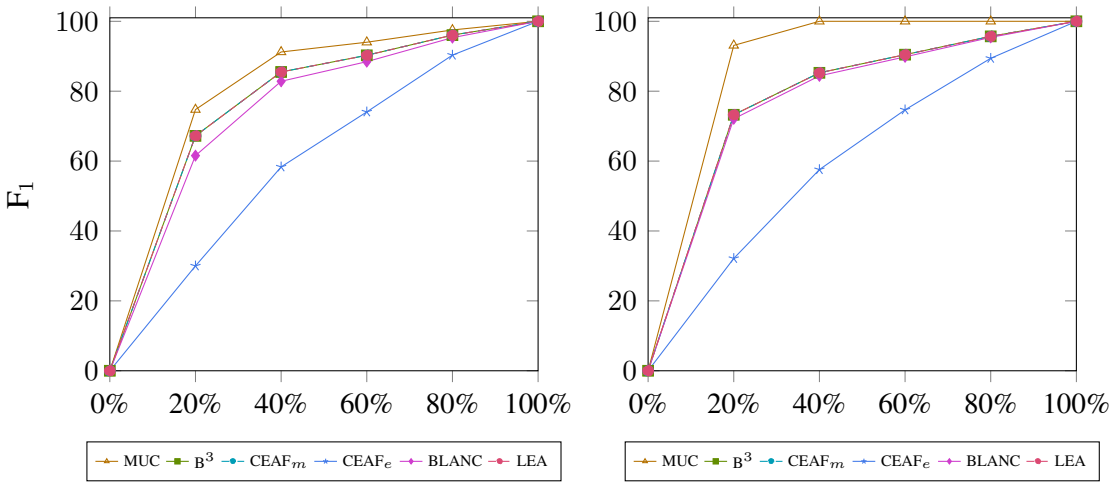


Figure 3.13: Resolving entities in decreasing order: (a) without singletons, and (b) with singletons. F_1 scores of the B^3, $CEAF_m$, and LEA metrics are the same in both (a) and (b).

We can observe the following points from Figure 3.13 and Figure 3.14:

1. *CEAF*$_e$ results in the same F_1 values regardless of the size of entities that are resolved or are missing. If larger entities are resolved first, *CEAF*$_e$ produces the lowest scores

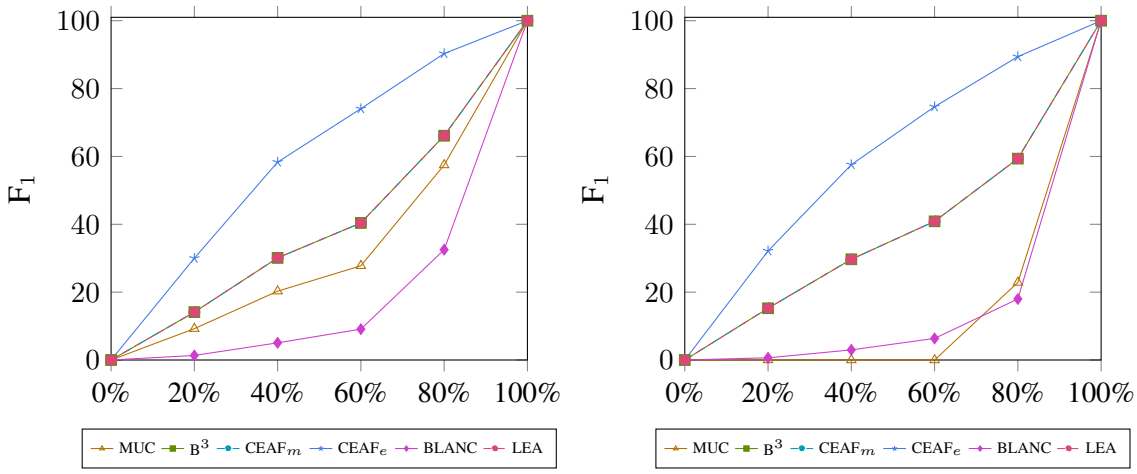


Figure 3.14: Resolving entities in increasing order. F_1 scores of B^3, CEAF, and LEA are the same.

and if smaller entities are resolved first, *CEAF*$_e$ results in the highest scores among all the examined evaluation metrics.

2. *B*^{3}, *CEAF*$_m$ and *LEA* result in the same F_1 values for all experiments, whether entities are resolved in decreasing/increasing order or whether singletons are included/excluded.

3. *MUC* and *BLANC* are the only metrics whose overall behavior changes with regard to the inclusion or exclusion of singletons.

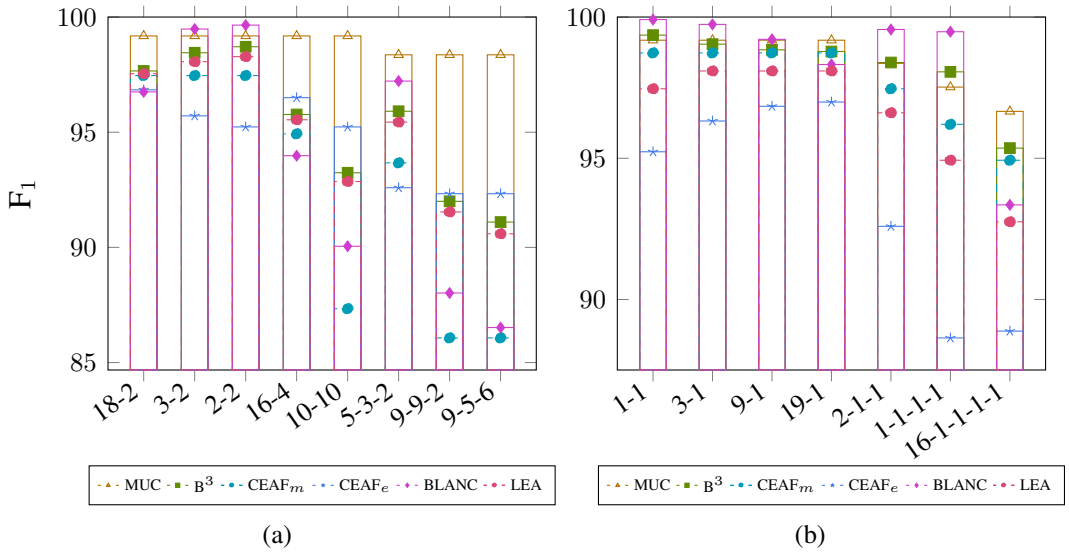
4. *BLANC* is very sensitive to the total number of links. *BLANC* F1 in Figure 3.14 for the response that correctly resolved 80% of the entities (but mainly the small ones) is smaller than *BLANC* F1 in Figures 3.10, 3.11, or 3.12 when the response contains *no* correct coreference link (0%).

5. The rankings of all metrics agree for all experiments of Figure 3.13 and Figure 3.14 with no singletons. When singletons are included, *MUC* is the only metric with different rankings because *MUC* cannot evaluate singletons.

3.4.3 Splitting/Merging Entities

The effect of splitting a single entity into two or more entities is studied in Figure 3.15. The overall effect of merging entities would be similar to that of splitting if the roles of the key and

response entities change. At each experiment, only one key entity is split. For the experiments in which singletons are not included, entities are split in a way that no singletons are created. For the experiments with singletons, each entity is split in such a way that at least one singleton is created. For example, 18-2 in the horizontal axis indicates that an entity of size 20 is split into two entities of size 18 and two. Similarly, 16-1-1-1-1 indicates that an entity of size 20 is split into five entities: one entity of size 16 and four singletons. For all experiments of Figure 3.15, the key set is the one without any singletons.



(a) (b)

Figure 3.15: Effect of splitting entities: (a) without singletons, and (b) with singletons.

The following observations can be drawn from Figure 3.15:

1. If the splits are in the same entity, all metrics prefer a smaller number of splits.

2. If no singleton is created, *MUC* only recognizes the number of splits regardless of the size of the split entity.

3. $CEAF_e$ does not differentiate 2-2 from 10-10, and 9-9-2 from 9-5-6. The reason is that the optimal mapping, as well as the number of response entities and mentions, are the same in both cases. Therefore, $CEAF_e$ does not discriminate these cases.

4. The highest disagreement occurs if the number of splits and the size of split entity are different, i.e., B^3: 18-2>5-3-2>16-4, *BLANC*: 5-3-2>18-2>16-4, *CEAF*: 18-2>16-4>5-3-2, and *LEA*: 18-2>16-4>5-3-2. These are the cases that are even hard for humans to rank. It is important to keep in mind that such indistinct differences can also lead to different rankings.

5. Among the experiments that we have performed so far, the F_1 values of the examined evaluation metrics have the highest disagreement for the experiments of Figure 3.15, especially when singletons are created after splitting.

3.4.4 Extra/Missing Mentions

Figure 3.16 shows the effect of adding extra mentions, i.e. mentions that are not included in any key entity, to the response entities. If we change the roles of the key and response entities, the overall effect of missing mentions would be similar to that of extra mentions. In the horizontal axis, the first number shows the number of extra mentions. The second number shows the size of the entity to which extra mentions are added. A zero entity size indicates that extra mentions are linked together.

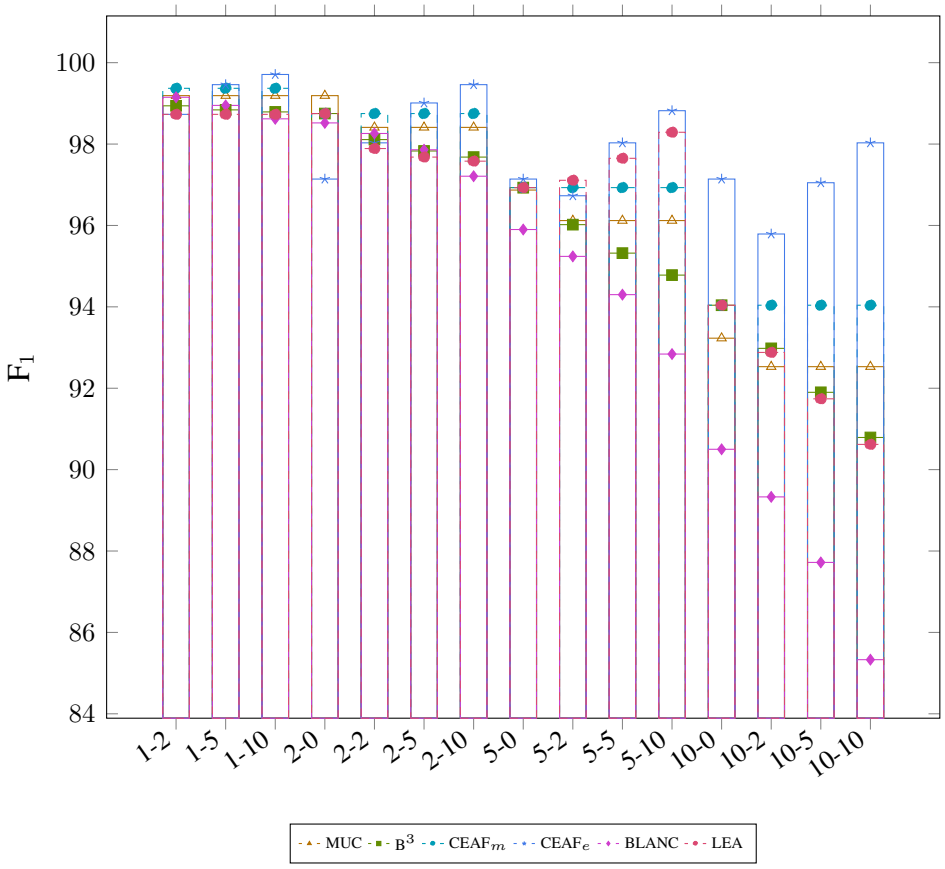


Figure 3.16: Effect of adding extra mentions to existing entities or as new entities. Singletons are not included in evaluations.

The following points are worth noting from the results of Figure 3.16:

1. *MUC* and *CEAF*$_m$ are the least discriminative metrics when the system output includes

extra mentions; i.e. *MUC* ranking: {1-2, 1-5, 1-10, 2-0}>{2-2, 2-5, 2-10, 3-0}>{5-0}>{5-2, 5-5, 5-10}>{10-0}>{10-2, 10-5, 10-10}, and *CEAF*$_m$ ranking:{1-2, 1-5, 1-10}>{2-0,2-2, 2-5, 2-10}>{5-0, 5-2, 5-5, 5-10}>{10-0, 10-2, 10-5, 10-10}.

2. When it comes to evaluating the effect of extra mentions, the examined evaluation metrics do not seem to agree much on the ranking of different outputs. According to *MUC*, B^3, *CEAF*$_m$, *CEAF*$_e$ and *BLANC*, the worst F_1 scores are assigned to {10-2, 10-5, 10-10}, {10-10}, {10-0, 10-2, 10-5, 10-10, 10-10}, {10-2}, {10-10} and {10-10}, respectively. Therefore, except for *CEAF*$_e$, all metrics rank 10-10 among the worst outputs in which the largest number of extra mentions are added to the largest entity. Similarly, based on the F_1 scores of *MUC*, B^3, *CEAF*$_m$, *CEAF*$_e$ and *BLANC*, {1-2, 1-5, 1-10, 2-0}, {1-2}, {1-2, 1-5, 1-10}, {1-10}, {1-2} and {2-0} are the best ranked outputs.

3. In the output 2-0, the extra mentions are linked together and therefore no incorrect information is added to the correctly resolved entities. *LEA* is the only metric that recognizes 2-0 as a less harmful error than 1-2 or 1-10. However, *LEA* does not discriminate the different outputs in which only one extra mention is added to an entity. If k extra mentions are added to an entity of size n, i.e. $n > 1$, the ratio of correctly resolved coreferent links is $\frac{n\times(n-1)}{(n+k)\times(n+k-1)}$. The corresponding resolution error multiplied by the importance of the response entity is $(n+k) \times (1 - \frac{n\times(n-1)}{(n+k)\times(n+k-1)})$. If $k = 1$, this equation is 2 regardless of n's value.

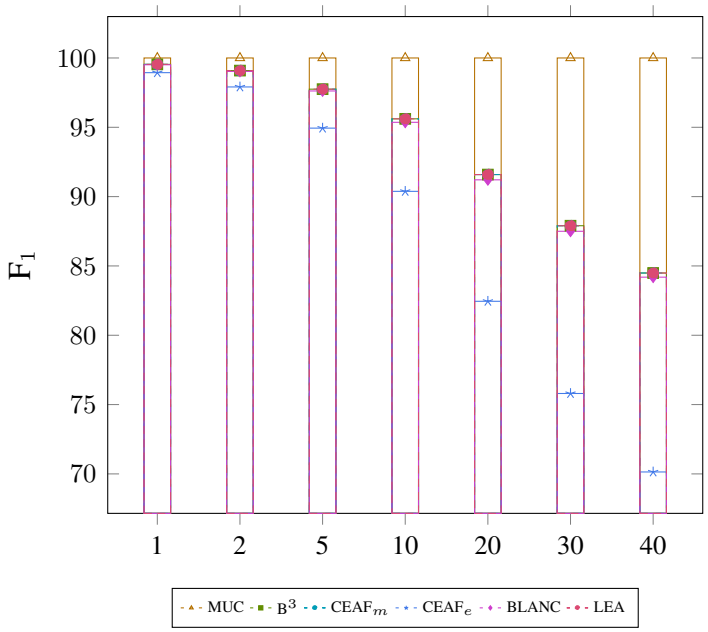


Figure 3.17: Effect of adding extra mentions to output entities as singletons. F_1 of B^3, *CEAF*$_m$, and *LEA* are the same.

The four main points that can be observed from the results of Figure 3.17 are:

1. *MUC* does not see singletons and therefore it has the highest results for all the experiments, i.e. 100%.

2. *CEAF*$_e$ results into the lowest scores for all the experiments in which extra mentions are added as singletons.

3. B^3, *CEAF*$_m$, and *LEA* produce the same F_1 scores for all experiments of Figure 3.17 and the *BLANC* values are also very close to those of these three metrics.

4. If we consider the results of Figure 3.16 and Figure 3.17 together, B^3, *CEAF*$_m$, *BLANC* and *LEA* rank the output in which only one singleton is added to the set of output entities as the best one. However, *CEAF*$_e$ rank 1-10 as the best output. Overall, these experiments show that *CEAF*$_e$ is very sensitive to singletons, or in other words, to the number of recognized entities.

3.4.5 Mention Identification

The mention identification effect that is explained in Section 3.2.1 is demonstrated in Figure 3.18. In all the experiments of Figure 3.18, the number of correct coreferent links is *zero*. The horizontal axis shows the mention identification accuracy in the system output. For instance, in 20%, only 20% of gold mentions are identified in the system output.

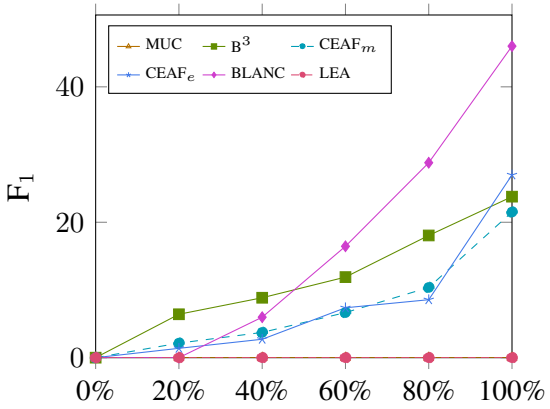


Figure 3.18: Effect of mention identification. The F_1 values of *MUC* are the same as *LEA*.

As we can see, B^3, *CEAF* and *BLANC* scores improve as the mention identification accuracy increases, even though no correct coreferent link is resolved. When all of the gold mentions are included in the system output, but all in a wrong entity, *BLANC* F_1 is higher than the case in which 80% of the key entities (in increasing order) are resolved correctly, i.e.

Figure 3.14(a). The F_1 of B^3, *CEAF* and *BLANC* in these experiments clearly violates the interpretability requirement. A coreference resolver with a non-zero score should have resolved some of the coreference relations.

3.5 LEA in Practice

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1	Avg. F_1	R	P	F_1
Peng15	69.54	75.80	72.53	56.91	65.40	60.86	55.49	55.98	55.73	63.04	51.91	58.97	55.21
Martschat15	68.55	77.22	72.63	54.64	66.78	60.11	52.85	60.30	56.33	63.02	50.64	62.87	56.10
Wiseman15	69.31	76.23	72.60	55.83	66.07	60.52	54.88	59.41	57.05	63.39	51.78	62.12	56.48
Wiseman16	69.75	77.49	73.42	56.95	66.83	61.50	53.85	62.14	57.70	64.21	53.15	63.13	57.71
Clark16a	70.38	79.47	74.65	58.03	69.46	63.23	54.57	63.58	58.73	65.54	54.37	65.79	59.54
Clark16b	70.98	78.81	74.69	58.97	69.05	63.61	55.66	63.28	59.23	65.84	55.31	65.32	59.90

Table 3.4: Results of state-of-the-art coreference resolvers on the CoNLL-2012 test set.

In this section, we show how coreference scores and system rankings change if we use *LEA* instead of previous evaluation metrics. For this purpose, we evaluate several state-of-the-art coreference resolvers in addition to the systems participating in the CoNLL-2012 shared task with the *LEA* metric. Among all the examined evaluation metrics in this chapter, *MUC*, B^3, $CEAF_e$ and the average of the F_1 scores of these three metrics, i.e. the *CoNLL* score, are the most popular ones. Therefore, we report the results of this section using *LEA* and these four metrics.

Table 3.4 shows the scores the mention-pair model of Peng et al. (2015a), the mention-ranking models of Wiseman et al. (2015), Martschat & Strube (2015), Wiseman et al. (2016) and Clark & Manning (2016a), and the mention-ranking model of Clark & Manning (2016a) that is directly optimized with reinforcement learning, i.e. Clark & Manning (2016b).

Except for *Peng15* and *Martschat15*, the overall rankings of the averaged CoNLL score and the *LEA* score are the same. However, based on the *CoNLL*, *Peng15* is ranked higher than *Martschat15* while according to *LEA*, *Martschat15* performs significantly better based on an approximate randomization test (Noreen, 1989). $CEAF_e$ also agrees with *LEA* for this ranking. However, $CEAF_e$ recall and precision are similar for *Peng* while based on *LEA*, *Peng*'s precision is marginally better than recall.

In addition to the state-of-the-art systems, we report the scores of boundary cases on the CoNLL-2012 test set in Table 3.5. The boundary cases include (1) *gold-sing*: all gold mention are detected correctly but all of them are left as singletons; (2) *gold-1ent*: all gold mentions are detected correctly but they are all linked together in a single entity; (3) *sys-sing*: all system

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1	Avg. F_1	R	P	F_1
gold-sing	0	0	0	22.93	100	37.3	51.76	11.87	19.31	18.87	0	0	0
gold-1ent	100	78.44	87.91	100	13.66	24.03	3.55	46.49	6.59	39.51	100	12.24	21.82
sys-sing	0.00	0.00	0.00	19.72	39.05	26.20	50.32	4.99	9.08	11.76	0.00	0.00	0.00
sys-1ent	88.01	29.58	44.28	84.87	2.53	4.91	1.50	19.63	2.80	17.33	82.31	2.27	4.43

Table 3.5: Results of boundary cases on the CoNLL-2012 test set.

mentions are detected as singletons in the system output; and (4) *sys-1ent*: all system mentions are linked together in a single entity.

According to the *MUC* and *LEA* scores, the ranking of the boundary cases are *gold-1ent* $>$ *sys-1ent* $>$ {*gold-sing*, *sys-sing*}. B^3 and $CEAF_e$ rank these cases as *gold-sing* $>$ *sys-sing* $>$ *gold-1ent* $>$ *sys-1ent*. Finally, based on *CoNLL*, we have a new ranking that is *gold-1ent* $>$ *gold-sing* $>$ *sys-1ent* $>$ *sys-sing*.

It is worth noting that *gold-sing* and *gold-1ent* are not realistic boundary cases. Unless all coreference relations are correctly resolved, it is impossible to identify all gold mentions correctly.

Table 3.6 presents the evaluations of the participating systems in the CoNLL-2012 shared task (closed task with predicted mentions). The rankings are specified in parentheses. We perform a significance test (Noreen, 1989) for assigning the same ranking to systems without significant differences for the *MUC*, B^3, $CEAF_e$ and *LEA* metrics. We consider an improvement as statistically significant if $p < 0.05$.

The main difference between the rankings of *CoNLL* and *LEA* is the rank of *xu*. Based on *LEA*, *xu* is significantly better than *chen* and *chunyuang*, while *CoNLL* ranks these two above *xu*. The recall values of *chen* and *chunyuang* for mention identification are 75.08 and 75.23, which are higher than those of the best performing systems in this shared task, i.e 72.75 for *fernandes*, and 74.23 for *martschat*. *chen* and *chunyuang* include 1850 and 1735 gold mentions in their outputs that do not have a single correct coreferent link. On the other hand, the number of these gold mentions in *xu* is 757. Therefore, these different rankings could be a direct result of the mention identification effect.

Overall, using a reliable evaluation metric instead of an average score benefits us in various ways: (1) the overall system rankings are more reliable, (2) we can perform a significance test to check whether there is a meaningful difference, and (3) the recall and precision values are meaningful.

Besides the above advantages of using *LEA* in evaluations, the use of *LEA* has an added benefit in optimizations.

	MUC	B^3	$CEAF_m$	$CEAF_e$	BLANC	CoNLL	LEA
fernandes	70.51 (1)	57.58 (1)	61.42	53.86 (1)	58.75	60.65	53.28 (1)
martschat	66.97 (3)	54.62 (2)	58.77	51.46 (2)	55.04	57.68	49.99 (2)
bjorkelund	67.58 (2)	54.47 (2)	58.19	50.21(3)	55.42	57.42	49.98 (2)
chang	66.38 (3)	52.99 (4)	57.10	48.94 (4)	53.86	56.10	48.50 (4)
chen	63.71 (6)	51.76 (5)	55.77	48.10 (4)	52.87	54.52	46.24 (6)
chunyuang	63.82 (6)	51.21 (5)	55.10	47.58 (6)	52.65	54.20	45.84 (6)
shou	62.91 (8)	49.44 (8)	53.16	46.66 (7)	50.44	53.00	43.97 (8)
yuan	62.55 (8)	50.11 (8)	54.53	45.99 (8)	52.10	52.88	44.76 (8)
xu	66.18 (5)	50.30 (5)	51.31	41.25 (11)	46.47	52.58	46.83 (5)
uryupina	60.89 (10)	46.24 (10)	49.31	42.93 (9)	46.04	50.02	41.15 (11)
songyang	59.83 (11)	45.90 (11)	49.58	42.36 (9)	45.10	49.36	41.25 (10)
zhekova	53.52 (12)	35.66 (12)	39.66	32.16 (12)	34.80	40.45	29.98 (12)
xinxin	48.27 (14)	35.73 (12)	37.99	31.90 (12)	36.54	38.63	29.22 (12)
li	50.84 (13)	32.29 (14)	36.28	25.21 (14)	31.85	36.11	27.32 (14)

Table 3.6: The results of the CoNLL-2012 shared task.

	MUC			B^3			CEAF$_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1	Avg. F_1	R	P	F_1
	CoNLL test set												
deep-coref [conll]	70.55	79.13	74.59	58.17	69.01	63.13	54.20	63.44	58.45	65.39	54.55	65.35	59.46
deep-coref [lea]	70.43	79.57	74.72	58.08	69.26	63.18	54.43	64.17	58.90	65.60	54.55	65.68	59.60
	WikiCoref												
deep-coref [conll]	58.59	66.63	62.35	44.40	54.87	49.08	42.47	51.47	46.54	52.65	40.36	50.73	44.95
deep-coref [lea]	57.48	70.55	63.35	42.12	60.13	49.54	41.40	53.08	46.52	53.14	38.22	55.98	45.43

Table 3.7: Comparison of the results on the CoNLL test set and WikiCoref.

During training, deep-coref (Clark & Manning, 2016a) chooses the best model based on the averaged CoNLL score on the development set. *deep-coref [conll]* in Table 3.7 shows the performance of deep-coref's ranking model in which the best trained model is selected based on the averaged CoNLL score on the development set. *deep-coref [lea]*, on the other hand, shows the performance of the same model in which the best set of weights is selected based on the *LEA* metric. As we can see, *deep-coref [lea]* performs slightly better than *deep-coref [conll]* based on all metrics.

In the experiments of Table 3.7, we use *LEA* instead of *CoNLL* only for choosing the best trained model. The model is not directly optimized based on either *CoNLL* or *LEA*. Among the common evaluation metrics, *CEAF* and *BLANC* are very time-consuming in their computations and *MUC* is also known to be the least discriminative one. Therefore, this leaves B^3 to be the popular metric in direct optimizations in coreference resolution (Clark &

Manning, 2016b; Le & Titov, 2017). Since *LEA* computations are as fast as those of B^3, we can easily use *LEA* for direct optimizations during training.

Le & Titov (2017) propose a modification in the coreference training objective that allows a direct optimization based on evaluation metrics. Le & Titov (2017) try both B^3 and *LEA* for their optimizations. They observe that the use of *LEA* instead of B^3 for optimization results in a slightly better performance based on *MUC*, B^3, *CEAF* and *LEA*.

3.6 Summary

In this chapter, we discuss the first step in establishing a reliable framework for coreference resolution evaluations, i.e. a reliable evaluation metric.

We overview the current evaluation metrics. Each of the current evaluation metrics has known drawbacks that make them individually unreliable for coreference evaluations. Besides the known drawbacks of the current evaluation metrics, we report a new drawback, namely the mention identification effect, which is common between B^3, $CEAF_m$, $CEAF_e$ and *BLANC*.

We then show how current evaluation metrics lead to counterintuitive results and therefore result in incorrect rankings or invalid recall/precision analyses.

Because of the lack of a reliable evaluation metric, the community comes up with the solution of averaging three of the existing metrics, namely *MUC*, B^3 and $CEAF_e$. However, as obvious as it is, averaging three unreliable metrics cannot result in a reliable one.

As a solution, we introduce the Link-Based Entity-Aware metric (*LEA*). *LEA* does not have the known drawbacks of the existing evaluation metrics, which makes it a reliable option for coreference evaluations. We perform thorough analyses on the behavior of *LEA* and the existing evaluation metrics in case of different types of coreference errors.

The *LEA* implementation is available at `https://github.com/ns-moosavi/coval`.

Chapter 4

Minimum Span Coreference Evaluation

The common practice in scoring coreference outputs is to evaluate detected coreferring mentions in an "all or nothing" manner. For instance, consider Example 4.1 from the CoNLL-2012 development set in which indices in parentheses denote to which key entity the coreferring mentions belong.

(4.1) This promoted a group of them to establish [a private bilingual school called Little Oxford]$_{(1)}$ in February 1998. [The school]$_{(1)}$ admits children from kindergarten through grade five of elementary school.

Consider a coreference resolver that correctly recognizes that "the school" refers to a private bilingual school. However, due to parsing errors, it detects "a private bilingual school called Little Oxford in February 1998" as the maximum boundary of the first mention and therefore links "the school" to this detected mention.

This system gets penalized based on both recall and precision and for all evaluation metrics. Recall drops because the system does not recognize the mention "a private bilingual school called Little Oxford". Precision drops because the system detects a spurious mention, i.e. "a private bilingual school called Little Oxford in February 1998".

The scores of this system are the same as those of a system that links "the school" to the mention "this" in the first sentence.

As we will further discuss in Section 4.1, detecting incorrect mention boundaries is a direct result of syntactic parsing errors or ambiguities. Therefore, with the current practice, we explicitly penalize coreference resolvers because of syntactic errors with respect to both recall and precision. Furthermore, such variations in detected mention boundaries do not even make a difference in understanding the coreference relations of the text.

This problem is not a new one but a well-known issue. The proposed solution to this problem is to annotate the minimum span as well as the maximum logical span of each mention during coreference annotations. This solution is followed by various coreference corpora

including MUC (Hirschman & Chinchor, 1997), ACE (Mitchell et al., 2002), and ARRAU (Uryupina et al., 2016). However, because of the resulting extra cost, minimum span annotations are not included in all available corpora. The CoNLL-2012 corpus, which is at the moment the main corpus for coreference evaluations, is an example of such corpora in which the annotation of minimum spans has been discarded so that the annotation size decreases and thus it would be possible to annotate a bigger corpus.

In this chapter, we introduce MINA, a MINimum span extraction Algorithm. MINA automatically extracts minimum spans at the evaluation level. Based on our analysis of the MUC and ARRAU corpora, automatically extracted minimum spans are compatible with those that are manually annotated, i.e. in more than 90% of examined mentions, automatically extracted minimum spans include the annotated ones. Therefore, by using MINA we can benefit from minimum span evaluations for all corpora including those without any minimum span annotations without adding any additional annotation cost.

Beside minimum span evaluations, MINA also provides means to analyze the effect of mention detection in coreference performance, which is discussed in Section 4.5.

4.1 Why Use Minimum Spans?

The common way of determining mention boundaries is to specify the largest logical span of each mention. If the same parse tree is used for detecting mention boundaries during both annotation and coreference resolution, retrieving the largest logical span of each candidate mention would be straightforward.

However, different parse trees are used during these two steps, e.g. gold parse trees for detecting gold mention boundaries and various system detected parse trees by each coreference resolver. Therefore, due to parsing variations, detecting the same largest logical spans as those that are extracted based on gold annotations is not a trivial task.

The CoNLL data contains both gold parse tree annotations, in `_gold_conll` files, as well as system detected parse trees, in `_auto_conll` files. If we compare mention boundaries that are detected based on the given system parse trees with the annotated boundaries, i.e. boundaries detected based on gold parse trees, we find numerous mismatch cases that are due to variations in the Prepositional Phrase (PP) attachment, which is a known challenge in parsing.

Examples 4.1, 4.2 and 4.3 are instances of such mismatches due to different PP-attachments in gold vs. system parse trees.

According to the gold parse tree of Figure 4.1, “a private bilingual school called Little Oxford” is the maximum span while the corresponding mention is “a private bilingual school called Little Oxford in February 1998” based on the system parse tree of Figure 4.2.

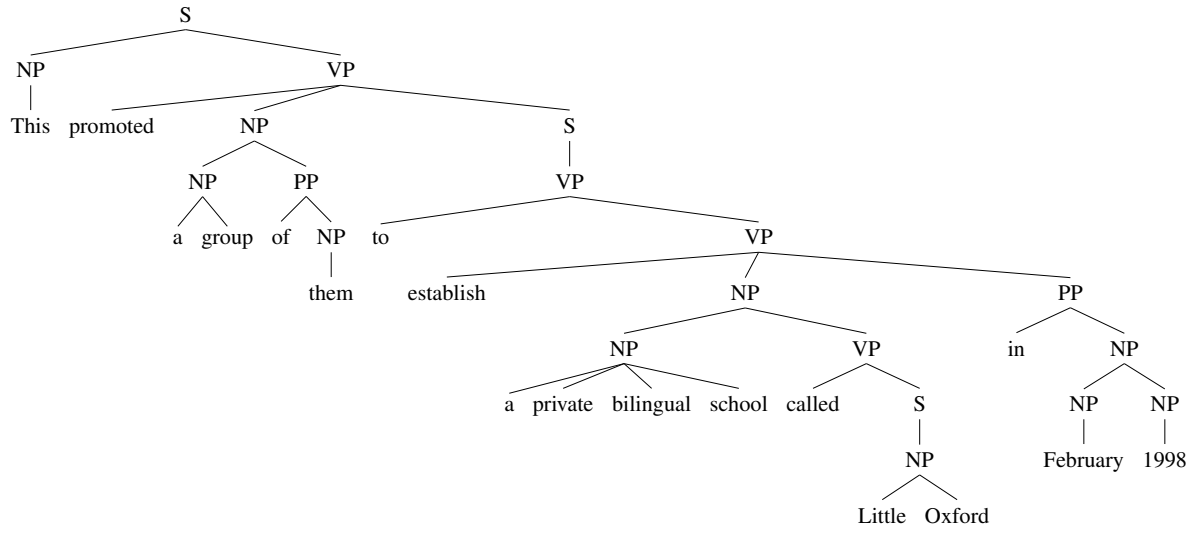


Figure 4.1: Gold parse tree of Example 4.1.

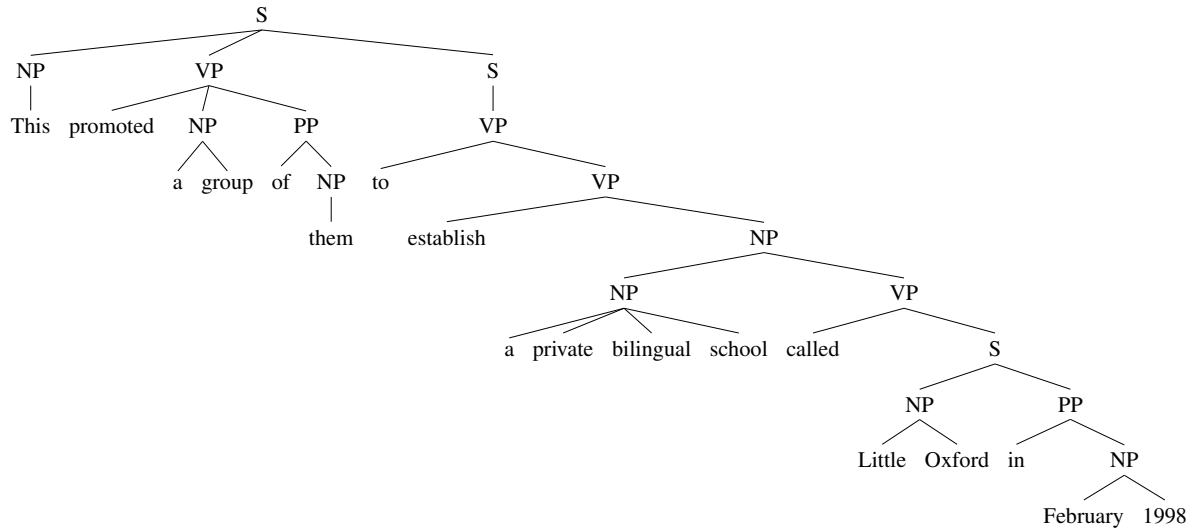


Figure 4.2: System parse tree of Example 4.1.

(4.2) These things that happened are examples for us.

Based on the gold parse tree of Figures 4.3, “examples” is annotated as the maximum span while according to the parse tree of Figures 4.4, the detected maximum span is “examples for us”.

(4.3) Rupert Murdoch’s News Corp. has [an extensive presence]$_{(1)}$, of course in this country.

Similarly, based on the system parse tree of Example 4.3, “an extensive presence, of course

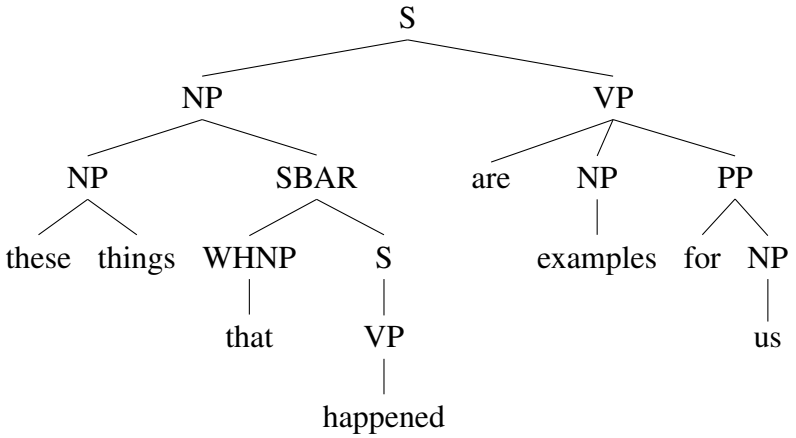


Figure 4.3: Gold parse tree of Example 4.2.

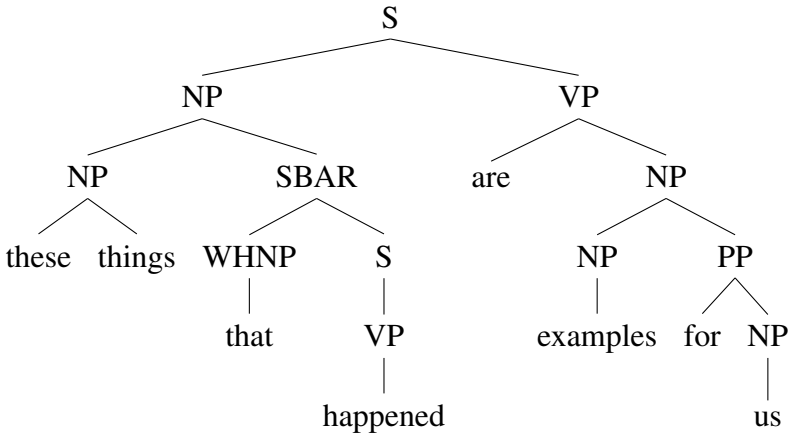


Figure 4.4: System parse tree of Example 4.2.

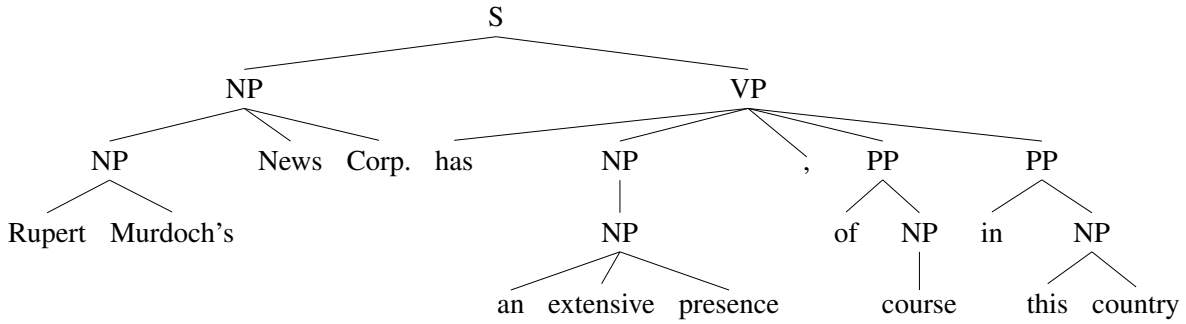


Figure 4.5: Gold parse tree of Example 4.3. "an extensive presence" is the annotated maximum span.

in this country" is the detected maximum boundary while based on the gold parse tree of Figure 4.5, "an extensive presence" is the corresponding annotated maximum boundary.

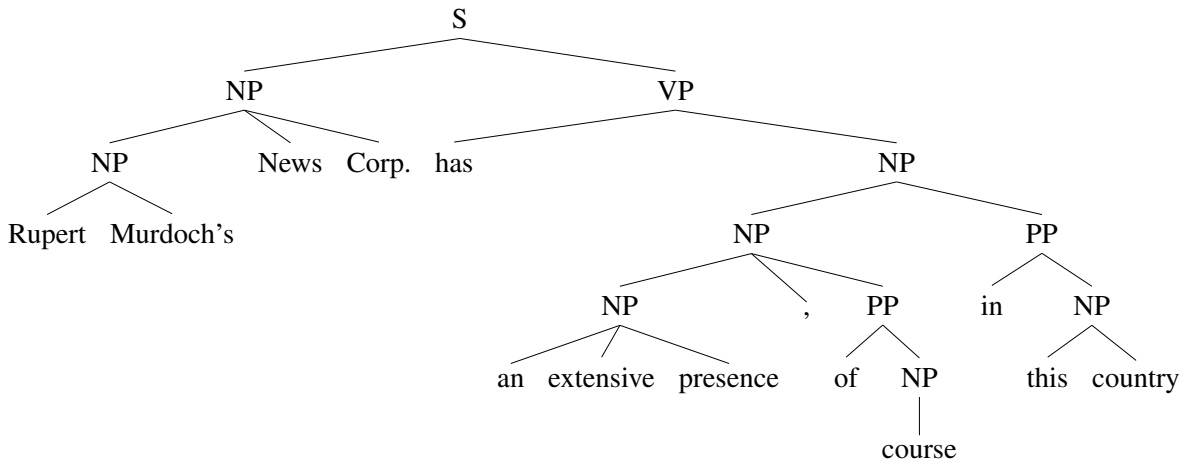


Figure 4.6: System parse tree of Example 4.3. "an extensive presence, of course in this country" is the detected maximum span for its corresponding gold mention "an extensive presence".

Differences between gold annotated compared to their corresponding system detected mentions are not limited to variations of PP-attachments. We can see examples of such variations in system vs. gold parse trees of Example 4.4 and 4.5 in Figure 4.7-Figure 4.10.

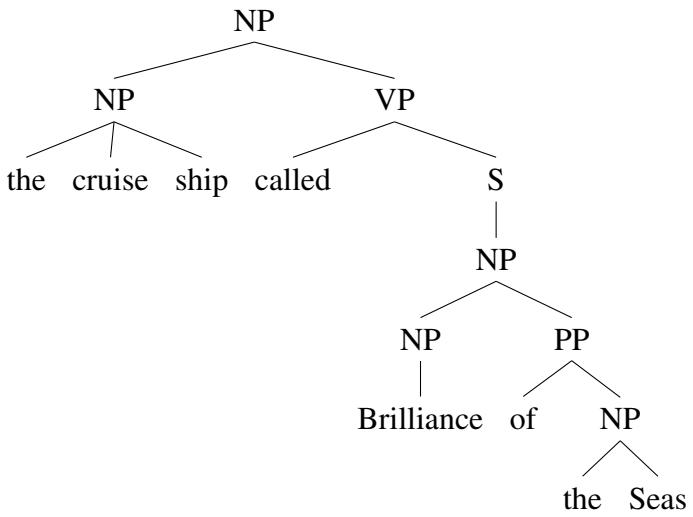


Figure 4.7: Gold parse tree of the annotated mention in Example 4.4.

(4.4) Federal investigators are now focusing on three men who were traveling on [the cruise ship called Brilliance of the Seas]$_{(1)}$ two Russian brothers who live in Brooklyn and one man who lives in California.

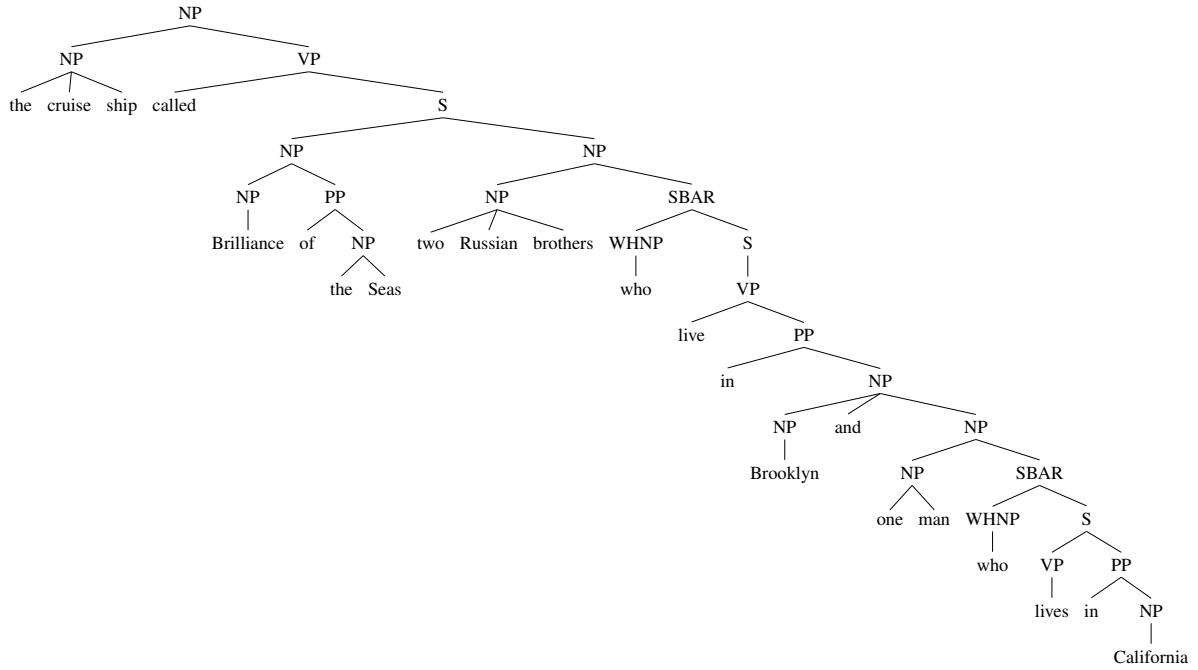


Figure 4.8: System parse tree of the detected mention in Example 4.4.

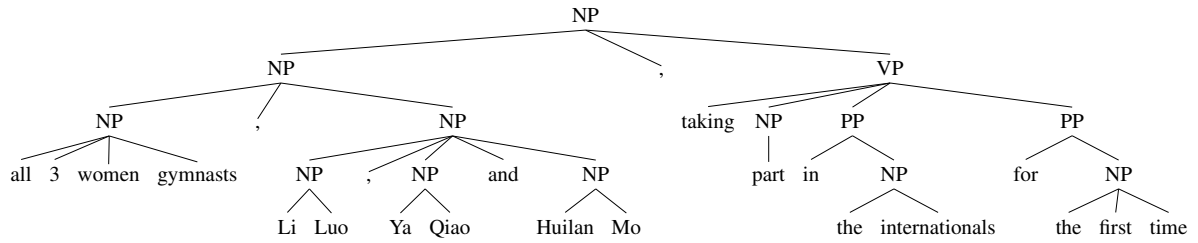


Figure 4.9: Gold parse tree of the gold mention in Example 4.5.

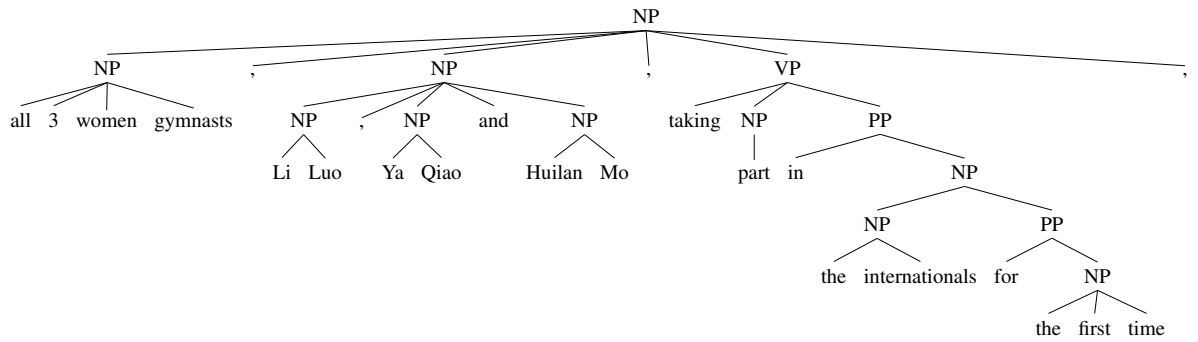


Figure 4.10: System parse tree of the detected maximum span corresponding to the gold mention of Figure 4.9.

(4.5) However, [all 3 women gymnasts, Li Luo, Ya Qiao and Huilan Mo, taking part in

the internationals for the first time]$_{(1)}$, performed well.

In each of the above examples, any system that uses the provided system parse trees for detecting mention boundaries, misses the resolution of the corresponding gold mentions, and potentially their coreferring mentions, only because of parsing errors.

4.2 How to Determine Minimum Spans?

In order to automatically determine minimum spans, we process the constituency-based parse trees of given mentions in a breadth-first manner. We then collect all the terminal nodes of the mention parse tree that (1) have the shortest distance to the root and (2) are children of a node with a valid phrase tag. Valid phrases include NP (noun phrase), NML (nominal modifier), QP (quantifier phrase used within NP) and NX (used within certain complex NPs for marking the head of the NP) if the given mention is a noun phrase. If a mention is a verb, VP is the only valid phrase tag. We only consider terminal nodes whose corresponding words contain characters and/or digits, i.e. we exclude terminal nodes that only include commas, dots or parentheses.

```
Algorithm MINA (root, tags, min-spans)
    if tags=∅ then
        if root is an NP then
            tags= {NP acceptable tags}
        end

        else if root is a VP then
            tags={VP acceptable tags}
        end
    end
    if root has acceptable terminal children then
        min-spans.add(all acceptable terminal nodes)
    end
    else
        for child ∈ root.children
        do
            if tags ! = ∅ and child.syntactic-tag ∈ tags then
                MINA (child, tags, min-spans)
            end
        end
    end
```

Algorithm 1: The minimum span extraction algorithm.

The minimum span extraction process is given in Algorithm 1. The input of the algorithm is the root node of the mention tree and the output is stored in "min-spans".

For instance, consider the mention parse tree of Figure 4.7. The terminal nodes that have the shortest distance to the root are "the cruise ship", which are children of an NP node, and "called", which is the child of a VP node. The root of the parse tree is an NP, therefore, VP is not a valid tag for minimum spans of this mention. As a result, "the cruise ship" will be selected as the minimum span of the given mention.

Based on this algorithm, the extracted minimum spans of all examples of Section 4.1 would be the same based on both gold and system parse trees, i.e. "a private bilingual school" based on Figure 4.1 and Figure 4.2, "examples" based on Figures 4.3 and 4.4, "News Corp." based on Figures 4.5 and 4.6, "the cruise ship" based on Figures 4.7 and 4.8, and "all 3 women gymnasts" based on Figures 4.9 and 4.10.

If the algorithm does not detect any minimum span for the given parse tree of a mention, it returns the whole span of the mention as the minimum span. Based on our analysis of the CoNLL development set, if we use gold parse trees for extracting minimum spans, this condition occurs for 14 mentions of the CoNLL development set of which ten are one-word mentions, e.g. "ours" is detected as an "ADJP" or "who" as "WHNP". Similarly, if we use system detected parse trees of the development set, this occurs for 60 mentions of which 51 mentions are one-word mentions.

In order to compute standard evaluation scores based on minimum spans, we extract minimum spans of all mentions in the key and response outputs based on the constituency-based parse trees of the key documents, i.e. the same parse information is used for extracting minimum spans of both key and response mentions. Parse trees do not necessarily need to be gold parse trees. If the parse information does not exist in the given key file, MINA automatically parses the key file using the Stanford PCFG parser (Klein & Manning, 2003) and uses the resulting parse information for extracting minimum spans.

In order to show that the extracted minimum spans are compatible with those that are manually determined by human experts, we compare the extracted minimum spans with the annotated ones in the MUC and ARRAU corpora. We parse above corpora with the Stanford English PCFG parser and then extract minimum spans according to the resulting parse trees.

The ratio of the extracted minimum spans that contain the corresponding manually annotated ones are 96% and 93% for the MUC-6 and MUC-7 corpora. For analyses on the ARRAU corpus, we randomly picked one thousand mentions that include more than one word from the RST Discourse Treebank subpart of the corpus. 98% of the extracted minimum spans include the corresponding annotated ones.

It is worth noting that our extracted minimum spans are generally larger than the annotated ones. However, in the majority of the examined mentions, the extracted minimum spans do include and therefore are compatible with the manually annotated ones. The mismatches are

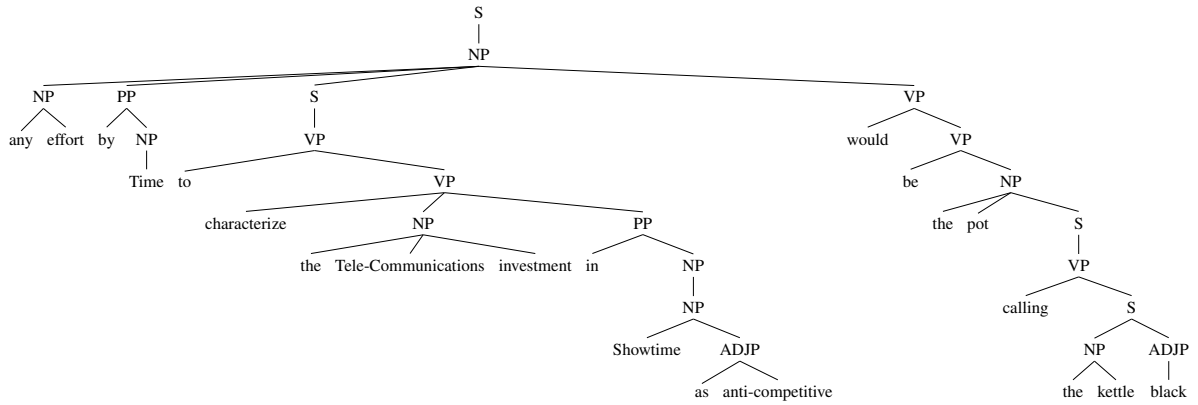


Figure 4.11: Generated parse tree for Example 4.6.

mainly due to parsing errors, i.e. the annotated mention boundary is not detected as a phrase in the parse tree.

For instance, in the ARRAU corpus, "the Tele-Communications investment in Showtime" in the following example is annotated as a mention. However, based on the automatic parse tree that is shown in Figure 4.11, this mention is not detected as a phrase in the parse tree.

(4.6) any effort by Time to characterize the Tele-Communications investment in Showtime as anti-competitive would be the pot calling the kettle black.

4.3 Related Work

As mentioned before, MUC, ACE, and ARRAU are among coreference corpora in which minimum spans as well as maximum spans are explicitly annotated. The reasoning behind annotating minimum spans is to not penalize coreference resolvers because of parsing errors in detecting mention boundaries. Simply put, it is an attempt to not make coreference resolution harder than it already is.

For MUC annotations, annotators were asked to specify the maximum span of a noun phrase as well as its minimum span. The specified minimum spans indicate the minimum strings that a coreference resolver must identify for the corresponding mentions. Therefore, if a coreference resolver identifies a minimum span instead of the corresponding maximum boundary, it will receive full credit. According to the MUC task definition[1], any response mention that includes the specified minimum string and does not include any token beyond those specified in the maximum span is valid. The minimum span is most often the head of

[1] http://www-nlpir.nist.gov/related_projects/muc/proceedings/co_task.html

the noun phrase. However, it can also be any part of the noun phrase the annotator decides can represent the identity of the noun phrase better. For instance, "Haden MacLellan PLC", "friends and colleagues" and "plane" are specified as the minimum strings of "Haden MacLellan PLC of Surrey, England", "our friends and colleagues who perished on a mission of peace" and "The Air Force T-43 plane" mentions, respectively.

The ACE annotation guidelines also include specifying minimum spans, which are referred to as heads. Similar to the MUC annotations, the specified heads/minimum spans are not necessarily a single word. For instance, in the ACE annotations "Public School of Health" and "Troy Brenna" are specified as heads of the "Harvard Public School of Health" and "Troy Brenna" mentions, respectively. The ARRAU annotation guideline regarding minimum spans is in line with that of the ACE dataset.

The coreference resolver of Peng et al. (2015b) is developed around the idea that working with mention heads is more robust than working with maximum mention boundaries. In this regard, they develop a system that resolves coreference relations based on mention heads. The resolved mention heads are then expanded to full mention boundaries using a separate classifier that is trained to extend heads to complete mention boundaries. Peng et al. (2015b) report the evaluation scores using maximum mention boundaries and also using mention heads. According to their results, the use of mention heads instead of maximum spans results in about two percent difference in evaluation scores of each of the examined systems. The exception is the Stanford rule-based system (Lee et al., 2011) for which the difference is up to four percent.

Peng et al. (2015b) extract mention heads using Collins' head finder rules (Collins, 1999). As an example, Collins' rules for finding the head of an NP-tagged phrase are:

- Return (last-word) if the POS tag of the last word is *POS*, i.e. possessive ending
- Else search from right to left for the first child that is either *NN* (singular or mass noun) *NNP* (singular proper name), *NNPS* (plural proper noun), *NNS* (plural noun), NX[2], *POS*, or *JJR* (comparative adjective)
- Else search for the first *NP* child from left to right
- Else search for the first *$*, *ADJP* (adjective phrase), or *PRN* (parenthetical) child from right to left
- Else search for the first *CD* (cardinal number) child from right to left
- Else search for the first *JJ* (adjective), *JJS* (superlative adjective), *RB* (adverb), or *QP* (quantifier phrase) child from right to left
- Else return the last word

[2]Used within certain complex NPs to mark the head of the NP

Peng et al. (2015b) use gold constituency-based parse trees and gold named entity information. The parse information is only used during training to train their mention head detection classifier. The gold named entity information is used to specify the whole span of named entities as their heads. The reason is that the head finding rules only specify one word as a head, and one-word heads can be troublesome for named entities, e.g. "York" will be detected as the head of "New York".

Unlike Peng et al. (2015b), MINA can use system-generated parse trees and does not require named entity information. Besides, except for one-word mentions, extracted minimum spans by MINA are mainly longer than one word, e.g. all terminal nodes of all valid phrases that have the shortest distance to the root, including determiners and adjectives, are returned as the minimum span.

4.4 Analysis

	maximum span		minimum span	
	CoNLL	LEA	CoNLL	LEA
fernandes	60.65	53.28 (1)	62.23	55.15 (1)
martschat	57.68	49.99 (2)	59.60	52.29 (2)
bjorkelund	57.42	49.98 (2)	58.88	51.61 (2)
chang	56.10	48.50 (4)	57.96	50.69 (4)
chen	54.52	46.24 (6)	56.57	48.64 (5)
chunyuang	54.20	45.84 (6)	56.11	48.11 (7)
shou	53.00	43.97 (8)	54.83	46.13 (8)
yuan	52.88	44.76 (8)	54.86	47.02 (8)
xu	52.58	46.83 (5)	53.86	48.39 (5)
uryupina	50.02	41.15 (11)	51.00	42.28 (10)
songyang	49.36	41.25 (10)	51.32	43.52 (10)
zhekova	40.45	29.98 (12)	42.20	31.86 (13)
xinxin	38.63	29.22 (12)	41.25	31.91 (12)
li	36.11	27.32 (14)	36.38	27.59 (14)

Table 4.1: Evaluation scores for the CoNLL-2012 shared task systems based on both maximum and minimum spans.

Table 4.1 shows the evaluations of the participating systems in the CoNLL-2012 shared task (closed task with predicted syntax and mentions) based on both maximum and minimum spans. Minimum spans are extracted using the gold parse trees of the CoNLL data. The rankings based on the *LEA* scores are specified in parentheses. We perform a significance test (Noreen, 1989) for assigning the same ranking to systems without significant differences. We consider an improvement as statistically significant if $p < 0.05$.

	Maximum span				Minimum span-gold parse				Minimum span-sys parse			
	CoNLL	LEA			CoNLL	LEA			CoNLL	LEA		
		R	P	F_1		R	P	F_1		R	P	F_1
gold	100	100	100	100	99.98	99.95	99.95	99.95	99.93	99.85	99.85	99.85
rule-based	55.60	43.73	51.53	47.31	57.62	45.89	54.24	49.72	57.51	45.77	54.11	49.59
Durrett13	61.24	49.66	59.17	54.00	63.09	51.48	61.94	56.23	62.94	51.33	61.74	56.06
Peng15	63.04	51.91	58.97	55.21	63.40	52.50	59.60	55.82	63.25	52.34	59.36	55.63
Martschat15	63.02	50.64	62.87	56.10	64.60	52.34	65.19	58.06	64.59	52.33	65.19	58.06
Wiseman15	63.39	51.78	62.12	56.48	65.18	53.62	64.78	58.67	65.09	53.50	64.67	58.56
Wiseman16	64.21	53.15	63.13	57.71	66.02	55.09	65.79	59.97	65.98	54.99	65.72	59.88
Clark16-ranking	65.59	54.01	66.44	59.58	67.27	55.84	68.83	61.66	67.19	55.72	68.73	61.55
Clark16-reinforce	65.84	55.31	65.32	59.90	67.50	55.62	69.60	61.83	67.43	55.52	69.54	61.75
Lee17-single	67.23	57.99	64.89	61.24	68.62	59.43	66.92	62.95	68.56	59.38	66.79	62.87
Lee17-ensemble	68.87	58.07	69.30	63.19	70.20	59.47	71.33	64.86	70.11	59.37	71.13	64.72

Table 4.2: Maximum and minimum span evaluations on the CoNLL-2012 test set. Minimum spans are extracted using gold parse trees.

Similarly, Table 4.2 presents the *LEA* and *CoNLL* scores of several recent coreference resolvers on the CoNLL-2012 test set based on both maximum and minimum span evaluations. We examined two different settings for minimum span evaluations: (1) extracting minimum spans using gold parse trees, i.e. "Minimum span-gold parse" columns, and (2) extracting minimum spans using system parse trees of the CoNLL data, i.e. "Minimum span-sys parse" columns.

The examined coreference resolvers are (1) "rule-based" representing the Stanford rule-based system (Lee et al., 2011), (2) "Durrett13" representing the results of Berkeley coreference resolver (Durrett & Klein, 2013) with the final feature set, (3) "Peng15" representing the scores of Peng et al. (2015b) coreference resolver, (4) "Martschat15" representing the ranking model of Martschat & Strube (2015), (5-6) "Wiseman15" and "Wiseman16" representing the ranking models of Wiseman et al. (2015) and Wiseman et al. (2016), respectively, (7-8), "Clark16-ranking" and "Clark16-reinforce" representing the ranking and reward rescaling models of deep-coref introduced by Clark & Manning (2016a) and Clark & Manning (2016b), respectively, and finally (9-10) "Lee17-single" and "Lee17-ensemble", which represent the single and ensemble models of e2e-coref introduced by Lee et al. (2017). Apart from above coreference resolvers we also report the evaluation of the key data compared to itself that is represented as "gold" in Table 4.2.

When a system is compared to itself, i.e. "gold", all evaluation scores should be 100%. However, it isn't the case for minimum span evaluations. We examined the cases that cause this performance drop. We found that it occurs because of a very few coreference annotation and parse tree errors.

For instance, one of the annotation problems occurs in the following sentence in which "Attorney Richard Jackson" and "Attorney Richard Jackson of Dallas" are annotated as two separate mentions referring to two different entities while the minimum span of both mentions is the same, i.e. "Attorney Richard Jackson".

(4.7) [[Attorney Richard Jackson]$_1$ of Dallas]$_2$ says a judgment for Triland could be satisfied in ways other than a monetary award

Example 4.8 and its corresponding gold parse tree in Figure 4.12 show an example of errors in the parse tree that confuses minimum span evaluations. According to the given parse tree, "President Bush's" is detected as the minimum span of "President Bush's uh meeting uh uh photo op some called it with US troops in Iraq". Therefore, coreference clusters of this mention and "President Bush's" get mixed up in minimum span evaluations.

(4.8) [[President Bush's]$_{(1)}$ uh meeting uh uh photo op some called [it]$_{(2)}$ with US troops in Iraq]$_{(2)}$

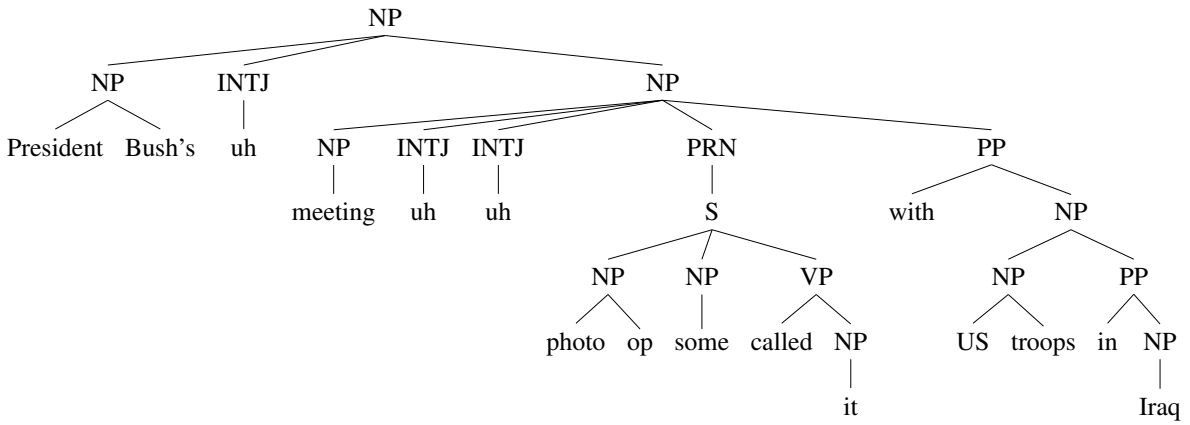


Figure 4.12: Gold parse tree of a gold mention in Example 4.8.

As we can see from the results of Table 4.1 and Table 4.2:

1. Performance based on various evaluation scores improves about two percent if we evaluate coreference relations based on minimum spans instead of maximum spans. In other words, parsing errors/variations result in two percent difference in overall scores in maximum span evaluations.

2. Based on "minimum span-gold parse" vs. "minimum span-system parse" scores, parsing errors/variations only result in about 0.2 percent difference in minimum span evaluations.

3. The ranking of coreference resolvers could be different if we use minimum spans instead of maximum spans.

Another interesting observation from Table 4.2 is that while *Peng15* (Peng et al., 2015b) is developed based on a similar idea of using minimum spans, i.e. using mention heads instead of maximum spans, it has the smallest difference between its maximum and minimum span evaluation scores, i.e. around 0.3 percent while it is one to two percent for other systems. This small difference is despite the fact that based on Peng et al. (2015b) results, there is about two percent difference between the performance of their system if it is evaluated on mention heads instead of maximum spans.

Only around 18% of Peng et al. (2015b)'s detected mentions in the final output are longer than two words. However, for gold mentions, this ratio is around 40%. Therefore, it seems that Peng et al. (2015b)'s mention boundary detection module does not go far beyond the boundaries of detected head words. On the other hand, as mentioned in Section 4.2, for mentions that include more than one word, the detected minimum spans by our algorithm are generally longer than one word.

Another difference between Peng et al. (2015b) head vs. maximum span compared to our minimum vs. maximum span evaluations is that based on their results, there is about four percent difference between the performance of the Stanford rule-based system in head vs. maximum span evaluations. However, this difference is about two percent in minimum vs. maximum span evaluations.

We also perform minimum span evaluation for the WikiCoref dataset. There is no gold parse information available for this dataset. We parse this dataset with the Stanford parser and report the minimum span evaluations based on the resulting parse trees. The results are presented in Table 4.3. There are 92 annotated singletons in the WikiCoref dataset. These singletons are removed in the evaluations of Table 4.3. Similar to the evaluations on the CoNLL data, there is about two percent difference between maximum vs. minimum span evaluations on the Wikicoref dataset. As we can see from the results of Table 4.3, "Clark16-reinforce" perform worse than "rule-based" and "Lee17-single" in the standard evaluation setting. However, "Clark16-reinforce" outperforms both of these systems if we use minimum spans instead of maximum spans.

4.5 Putting the Effect of Mention Detection into Relief

Recent state-of-the-art coreference resolvers vary in many different aspects including preprocessing modules, incorporated features, resolution models, and learning algorithms. Therefore, it is not clear where the improvements come from, and whether recent improvements are complementary and can be combined effectively.

	Maximum span				Minimum span-sys parse			
	CoNLL	LEA			CoNLL	LEA		
		R	P	F_1		R	P	F_1
gold	100	100	100	100	99.82	99.59	99.59	99.59
rule-based	52.23	39.34	49.01	43.64	54.28	41.15	51.97	45.93
Clark16-ranking	53.38	39.37	54.30	45.64	55.73	41.61	57.69	48.35
Clark16-reinforce	50.99	42.84	41.70	42.26	54.62	46.93	45.95	46.44
Lee17-single	50.75	41.24	47.56	44.17	52.46	42.68	49.90	46.01
Lee17-ensemble	54.15	41.27	56.97	47.87	55.60	42.54	59.11	49.47

Table 4.3: Maximum vs. minimum span evaluations on the WikiCoref dataset.

Mention detection is one of the most influential but the least challenged modules of coreference resolvers. For instance, as we saw in Section 4.4, the performance could vary up to two percent because of varying mention boundaries. Apart from minimum span evaluations, the MINA package provides means to analyze the effect of mention detection in coreference evaluations.

The majority of existing coreference resolvers are pipelined systems. They use a separate mention detection module in a preprocessing step. Existing mention detection modules are mainly rule-based approaches that extract candidate noun phrases from syntactic parse trees. Pipelined coreference resolvers can only resolve mentions that are presented to them, i.e. detected by their corresponding mention detection.

In MINA evaluations, apart from key and system outputs, one can also provide the output of the corresponding mention detection in which all detected mentions are annotated[3]. By having the set of all detected mentions, MINA can then remove all mentions that are not recognized by the mention detection from the key data. This way, one can evaluate coreference resolvers solely based on their performance for mentions that were available to them. For instance, the CoNLL data includes gold mentions that are verb phrases. However, the majority of existing coreference resolvers only extract noun phrases as their candidate mentions. Therefore, they get penalized for not resolving verb phrases that were not even presented to them. Missing candidate mentions are not limited to verb phrases. As an example, the Stanford CoreNLP mention detection does not extract some named entities like time, date and percent.

If this evaluation, i.e. normalizing the effect of mention detection, is performed on a dataset

[3]All detected mentions need to have a coreference label in this file. For instance, they can be all assigned the same coreference chain or each mention can have its own coreference label.

like CoNLL, which does not include singletons, MINA also removes the singletons that are created by removing undetected mentions. For instance, if the verb phrase "remained" is not detected by the mention detection and the mention "this" is the only coreferring mention with "remained", MINA also removes "this" from the key data.

Table 4.4 shows the evaluation scores based on maximum spans if the system is evaluated (1) on all key mentions, i.e. "maximum span" columns, and (2) on all key mentions that are detected by the system's mention detection, i.e. "maximum span-normalized" columns.

	Maximum span				Maximum span-Normalized			
	CoNLL	LEA			CoNLL	LEA		
		R	P	F_1		R	P	F_1
rule-based	55.60	43.73	51.53	47.31	59.87	54.13	51.59	52.83
Durrett13	61.24	49.66	59.17	54.00	65.83	59.22	59.17	59.19
Martschat15	63.02	50.64	62.87	56.10	68.26	61.15	62.78	61.95
Wiseman15	63.39	51.78	62.12	56.48	68.20	61.79	62.12	61.96
Wiseman16	64.21	53.15	63.13	57.71	69.24	63.41	63.13	63.27
Clark16-ranking	65.59	54.01	66.44	59.58	71.56	65.94	66.45	66.19
Clark16-reinforce	65.84	55.31	65.32	59.90	71.75	65.67	67.28	66.46

Table 4.4: Normalizing the maximum span evaluation scores with regard to the recall of the mention identification module. Results are reported on the CoNLL-2012 test set.

Similarly, Table 4.5 presents the evaluation scores for minimum span evaluations if the system is evaluated (1) on all key mentions, i.e. "minimum span" columns, and (2) on key mentions that their corresponding minimum spans are detected by the mention detection, i.e. "minimum span-normalized" columns. Since end-to-end systems do not include a mention detection step, and therefore do not have a predefined set of candidate mentions, the coreference models of Lee et al. (2017) are not included in the experiments of Table 4.4 and Table 4.5.

It is worth noting that the resulting scores that are normalized with regard to the recall of employed mention detections are not intended for comparing the overall performance of various coreference resolvers. Instead, they are only a means to analyze the extent to which the recall of a mention detection can affect the performance of its corresponding coreference resolver.

As can be seen from the results of Table 4.4 and Table 4.5:

1. If we normalize the effect of mention detection, there is no considerable difference between the performance of examined coreference resolvers in minimum vs. maximum span evaluations, i.e. "maximum span-normalized" vs. "minimum span-normalized".

	Minimum span				Minimum span-normalized			
	CoNLL	LEA			CoNLL	LEA		
		R	P	F_1		R	P	F_1
rule-based	57.62	45.89	54.24	49.72	60.74	52.99	54.27	53.62
Durrett13	63.09	51.48	61.94	56.23	66.17	57.37	61.94	59.57
Martschat15	64.60	52.34	65.19	58.06	68.20	58.98	65.07	61.87
Wiseman15	65.18	53.62	64.78	58.67	68.41	59.78	64.78	62.18
Wiseman16	66.02	55.09	65.79	59.97	69.40	61.40	65.79	63.52
Clark16-ranking	67.27	55.84	68.83	61.66	71.63	63.88	68.82	66.26
Clark16-reinforce	67.50	55.62	69.60	61.83	71.82	63.67	69.61	66.51

Table 4.5: Normalizing the minimum span evaluation scores with regard to the recall of the mention identification module. Results are reported on the CoNLL-2012 test set. Minimum spans are extracted using gold parse trees.

2. The limited recall of a mention detection has a considerable effect on the overall performance, i.e. five to six percent difference between “maximum span” and “maximum span-normalized”. The use of minimum spans instead of maximum spans in coreference evaluations is a step to reduce this gap to some extent, i.e. three to four percent difference between “Minimum span” and “Minimum span-normalized”.

	Maximum span				Maximum span-NP only			
	CoNLL	LEA			CoNLL	LEA		
		R	P	F_1		R	P	F_1
rule-based	55.60	43.73	51.53	47.31	56.30	45.24	51.62	48.22
Durrett13	61.24	49.66	59.17	54.00	62.35	51.36	59.21	55.01
Martschat15	63.02	50.64	62.87	56.10	64.26	52.35	62.91	57.15
Wiseman15	63.39	51.78	62.12	56.48	64.56	53.57	62.14	57.54
Wiseman16	64.21	53.15	63.13	57.71	65.43	54.99	63.17	58.80
Clark16-ranking	65.59	54.01	66.44	59.58	66.92	55.85	66.48	60.70
Clark16-reinforce	65.84	55.31	65.32	59.90	67.12	55.63	67.28	60.90
Lee17-single	67.23	57.99	64.89	61.24	68.35	59.49	65.54	62.37
Lee17-ensemble	68.87	58.07	69.30	63.19	69.93	59.53	69.75	64.24

Table 4.6: Maximum span evaluations on (1) all mentions and (2) only on noun phrases. Results are reported on the CoNLL-2012 test set.

As mentioned above, verb phrases are among undetected mentions in the examined coreference resolvers. By using MINA one can evaluate coreference outputs only on noun phrases. In this evaluation, MINA removes all verb phrases from the key data. Similar to the normalized evaluations, if the evaluation is performed on data like the CoNLL dataset, MINA also removes the singletons that are created after removing verb phrases.

	Minimum span				Minimum span-NP only			
	CoNLL	LEA			CoNLL	LEA		
		R	P	F_1		R	P	F_1
rule-based	57.62	45.89	54.24	49.72	58.36	47.49	54.34	50.68
Durrett13	63.09	51.48	61.94	56.23	64.23	53.25	61.98	57.28
Martschat15	64.60	52.34	65.19	58.06	65.87	54.11	65.22	59.15
Wiseman15	65.18	53.62	64.78	58.67	66.38	55.47	64.80	59.77
Wiseman16	66.02	55.09	65.79	59.97	67.29	56.99	65.82	61.09
Clark16-ranking	67.27	55.84	68.83	61.66	68.64	57.75	68.87	62.82
Clark16-reinforce	67.50	55.62	69.60	61.83	68.85	57.53	69.64	63.01
Lee17-single	68.62	59.43	66.92	62.95	69.78	60.99	67.60	64.12
Lee17-ensemble	70.20	59.47	71.33	64.86	71.30	60.97	71.81	65.95

Table 4.7: Minimum span evaluations on (1) all mentions and (2) only on noun phrases. Results are reported on the CoNLL-2012 test set. Minimum spans are extracted using gold parse trees.

Table 4.6 and Table 4.7 present the evaluation scores of the examined coreference resolvers if they are only evaluated on noun phrase coreference relations of the key data.

Among the examined coreference resolvers, "Lee17-single" and "Lee17-ensemble" models of Lee et al. (2017) are the only systems that also include verb phrases in their output. For NP-only evaluations, MINA also performs the process of removing verb phrases and the resulting singletons from the system output. This way, the system will not get penalized for correctly resolving verb phrases in NP-only evaluations.

By comparing the results of "Minimum span-normalized" vs. "Minimum span-NP only", we can see that the missing mentions that affect the overall performance are not all verb phrases. Therefore, there is still room for improvement in coreference resolution by improving the recall of noun phrase mention detection.

4.6 Summary

Mentions in coreference corpora are annotated by their maximum logical span. As we show in Section 4.1, evaluations based on maximum spans will directly penalize coreference resolvers because of parsing errors. Penalizing coreference resolvers because of parsing errors is a known problem that is addressed by annotating minimum spans as well as maximum spans in several corpora including MUC, ACE, and ARRAU. However, annotating minimum span brings in an extra cost and therefore it is not a scalable solution for large coreference corpora.

In this chapter, we introduced MINA, a tool for coreference evaluations based on minimum spans. Instead of addressing the above problem at the annotation level, MINA handles varying mention boundaries by automatically extracting corresponding minimum spans at the evaluation level. Based on our analysis of the MUC and ARRAU datasets, MINA automatically extracted minimum spans are compatible with the manually annotated ones. The use of minimum spans instead of maximum spans leads to about two percent difference in coreference evaluation scores. MINA can extract minimum spans based on system-generated parse trees. The use of system parse trees instead of gold parse trees only results in about 0.2 percent variations in final evaluation scores.

Apart from minimum span evaluations, we also analyze the effect of incorporated mention detections in the given coreference resolvers. Based on our results, there is still room for improvement in the overall performance of coreference resolvers by solely improving the recall of mention detection. The MINA algorithm is also incorporated in `https://github.com/ns-moosavi/coval`.

Chapter 5

Robust Evaluation Scheme

The introduction of the CoNLL dataset enabled a significant boost in the performance of coreference resolvers. There is about 13 percent difference between the CoNLL score of the currently best coreference resolver, i.e. e2e-coref by Lee et al. (2017), and that of the winner of the CoNLL-2011 shared task, i.e. the Stanford rule-based system by Lee et al. (2013). However, this substantial improvement does not seem to be visible in downstream tasks. Worse, the difference between state-of-the-art coreference resolvers and the rule-based system drops significantly when they are applied to a new dataset, even a dataset with consistent definitions of mentions and coreference relations (Ghaddar & Langlais, 2016a).

Table 5.1 shows the results of several coreference resolvers on both the official CoNLL test set and WikiCoref [1]. As we can see, the performance of coreference resolvers constantly improves on the CoNLL test set over time. However, the superiority of supervised and more complex coreference resolvers on the CoNLL dataset does not hold on WikiCoref.

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1	Avg. F_1	R	P	F_1
	CoNLL test set												
rule-based [2011]	64.29	65.19	64.74	49.18	56.79	52.71	52.45	46.58	49.34	55.60	43.72	51.53	47.30
berkeley [2013]	67.56	74.09	70.67	53.93	63.50	58.33	53.29	56.22	54.72	61.24	49.66	59.17	54.00
cort [2015]	67.83	78.35	72.71	54.34	68.42	60.57	53.10	61.10	56.82	63.37	50.40	64.46	56.57
deep-coref [2016]	70.43	79.57	74.72	58.08	69.26	63.18	54.43	64.17	58.90	65.60	54.55	65.68	59.60
e2e-coref [2017]	74.02	77.82	75.88	62.58	67.45	64.92	59.16	62.96	61.00	67.27	58.90	63.79	61.25
	WikiCoref												
rule-based [2011]	60.42	61.56	60.99	43.34	53.53	47.90	50.89	42.70	46.44	51.77	38.79	48.92	43.27
berkeley [2013]	68.52	55.96	61.61	59.08	39.72	47.51	48.06	40.44	43.92	51.01	-	-	-
cort [2015]	70.39	53.63	60.88	60.81	37.58	46.45	47.88	38.18	42.48	49.94	-	-	-
deep-coref [2016]	57.48	70.55	63.35	42.12	60.13	49.54	41.40	53.08	46.52	53.14	38.22	55.98	45.43
e2e-coref [2017]	57.92	68.14	62.61	42.66	57.63	49.03	38.55	45.85	41.88	51.17	38.37	53.24	44.60
e2e-coref-half [2017]	60.13	64.45	62.22	45.19	51.75	48.25	38.18	43.49	40.66	50.38	40.69	47.55	43.85

Table 5.1: Performance comparison on the CoNLL test set and WikiCoref.

[1] The results of the berkeley and cort systems on WikiCoref are taken from (Ghaddar & Langlais, 2016b).

In this chapter, we show that the large gap between the performance of coreference resolvers on the CoNLL and non-CoNLL datasets is mainly due to (1) the great overlap between the CoNLL datasets, and (2) the extensive use of lexical features in recent coreference resolvers. The training, development and test sets of the CoNLL corpus share many common coreferring mentions.

This high degree of overlap between the training and test sets makes the CoNLL test set unreliable for evaluations. A coreference resolver with a better evaluation score on the CoNLL dataset may not necessarily be better at resolving coreference relations. It may be the case that the resolver is better at memorizing mentions or mention-pairs of the training data.

Lexical features enable coreference resolvers to model some linguistic phenomena implicitly, and at a finer level of granularity, instead of modeling them by heuristic features (Durrett & Klein, 2013). However, the knowledge that is mainly captured by lexical features substantially limits the generalization of the coreference model to other domains.

Finally, we argue that in order to ensure meaningful improvements in coreference resolution, we must perform out-of-domain evaluations.

5.1 Overlap in the CoNLL Dataset

In this section, we examine the role of the CoNLL dataset in the limited generalization of the models that are trained on this dataset.

As mentioned in Section 2.2.1, the CoNLL dataset has three official splits: training, development and test sets. The number of (sub)documents[2] in the training, development and test sets are 2802, 343, and 348, respectively. The CoNLL corpus has seven genres including *bc* (broadcast conversation), *bn* (broadcast news), *mz* (magazine), *nw* (newswire), *pt* (Bible), *tc* (telephone conversations), and *wb* (weblog).

We measure the overlap ratio between coreferent mention strings of the test and training sets. We report the overlap ratios for each genre of the test set separately. Since pronouns are very likely to appear as coreferent mentions in all documents, we exclude pronominal mentions for computing the overlap.

We compute the overlap ratios considering various configurations:

1. **In-genre, out-of-genre and overall overlap ratios.** In the in-genre setting, we compute the overlap only within the same genre of the test and training sets. For instance, in order to compute the overlap ratio of the *bc* genre of the test set, we only consider coreferring mentions of the *bc* genre of the training set. For out-of-genre ratios, we compute the overlap ratio of one test genre based on all non-identical genres of the training set. For

[2]Long documents are split into smaller parts to make the coreference annotation easier. However, each part is annotated independently. Therefore, they should be considered as different documents.

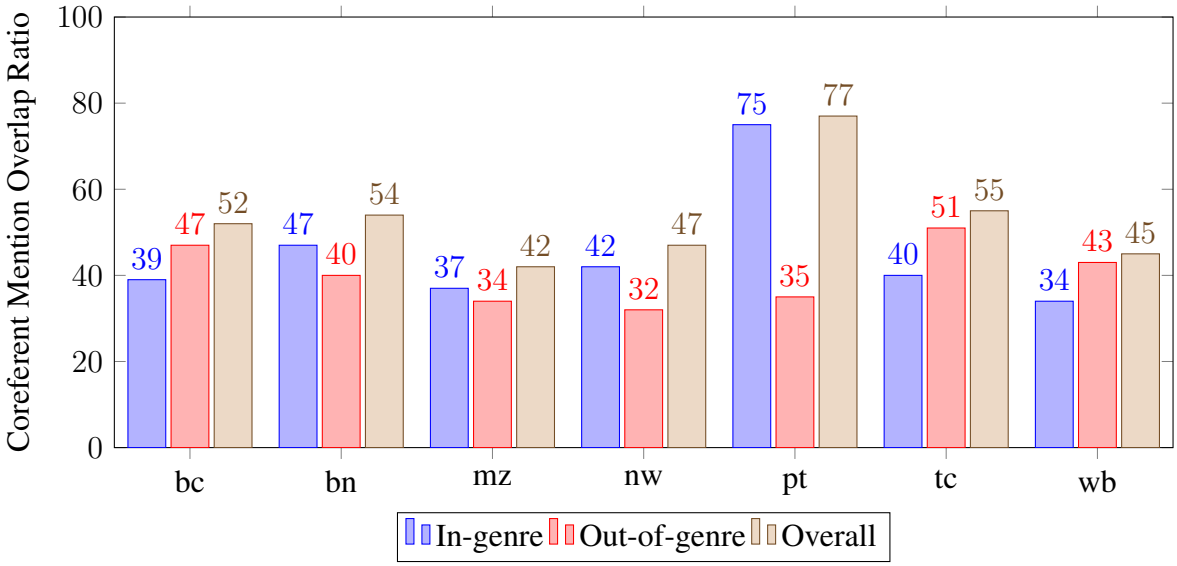


Figure 5.1: Overlap ratios between strings of non-pronominal coreferent mentions of the test set and those of the training set. In-genre overlap ratios are computed between coreferent mentions of the same genres. Out-of-genre overlap ratios are computed between coreferent mentions of one genre in the test set and those of all other genres of the training set. Overall overlap ratios are computed between coreferent mentions of a genre in the test set and those of all genres in the training set.

instance, the overlap ratio of the test *bc* genre is computed based on the *bn*, *mz*, *nw*, *pt*, *tc*, and *wb* genres of the training set. In the overall setting, the overlap of a test genre is computed considering all genres of the training set. For instance, the overlap of the *bc* genre is computed based on all genres of the training data including *bc* itself.

2. **Mention strings vs. mention heads.** In Figure 5.1, ratios are computed based on the whole mention string, i.e. the whole string of a coreferent mention should appear in both test and training sets in order to be considered as an overlap. On the other hand, we compute the overlap ratios only based on mention heads in Figure 5.2, i.e. if the head of a non-pronominal coreferent mention in the test set appears as the head of a non-pronominal coreferent mention in the training set, it is considered as an overlap.

The high overlap ratios in both Figure 5.1 and Figure 5.2 indicate that the CoNLL official test set is very similar to the training set and they share common entities. Therefore, a coreference resolver can benefit from this overlap by overtraining and memorizing more properties of the training data. This high degree of similarity between the training and test sets makes

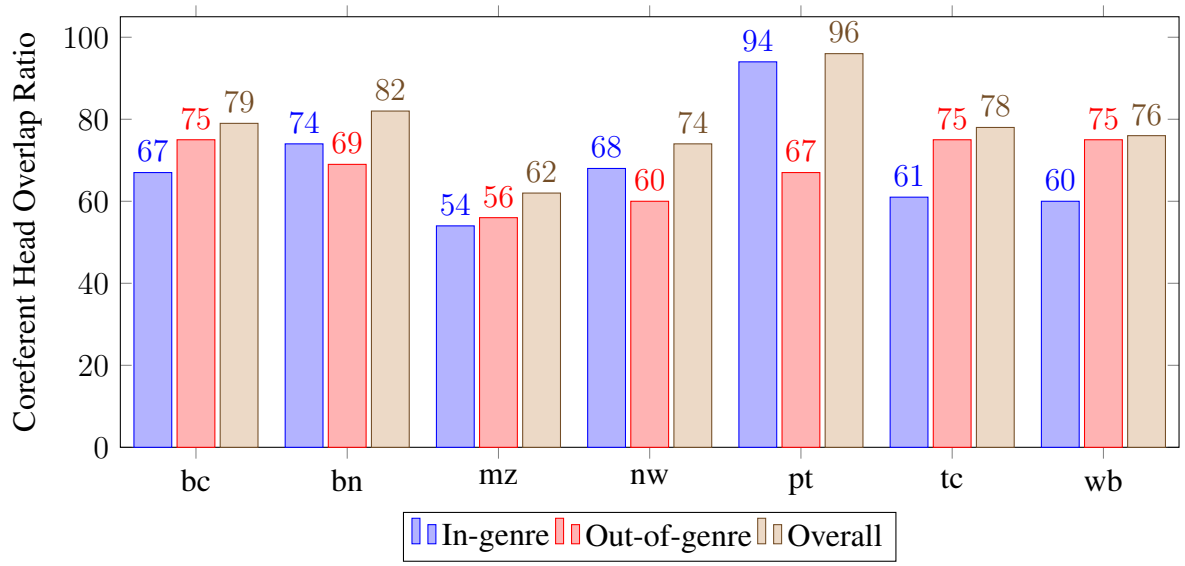


Figure 5.2: Overlap ratios between heads of non-pronominal coreferent mentions of the test set and those of the training set. In-genre overlap ratios are computed between coreferent mention heads of identical genres. Out-of-genre overlap ratios are computed between coreferent mention heads of one genre in the test set and those of all other genres in the training set. Overall overlap ratios are computed between coreferent mention heads of a genre in the test set and those of all genres in the training set.

the CoNLL official test set highly problematic for reliable evaluations.

According to the results of Figure 5.1 and Figure 5.2, the *pt* genre of the test set has the highest overlap with the training data. The overall overlap ratios of this genre are 77% and 96% based on mention strings and mention heads, respectively. Therefore, a coreference resolver can benefit greatly from memorizing mentions of this genre during training.

The most immediate effect of this large overlap is on mention identification and anaphoricity determination. Mention detection and anaphoricity determination both have crucial effects on the overall performance of a coreference resolver. The large gap between the performance of coreference resolvers on gold mentions in comparison to their performance on system mentions indicates the importance of both mention detection and anaphoricity determination modules, e.g. refer to Moosavi & Strube (2016a).

If the training and test sets share many coreferent mentions, mention detection and anaphoricity determination modules can memorize the correct boundary and anaphoricity information of seen mentions. However, this way of "getting better" by memorizing the properties of mentions from the training data is not beneficial or valuable for solving the general coreference

problem.

As we will also show in Section 5.2, this substantial overlap between training and test sets does not only affect individual mention processing. The overlap also biases the coreference resolver towards mention-pairs seen in the training data. Therefore, it limits the resolver generalization to mentions other than those that are seen during training.

The *pt*, *nw* and *mz* genres are three genres with the lowest out-of-genre overlaps. Therefore, they are good candidates for performing out-of-domain evaluations using the CoNLL corpus. In order to do so, we can choose one of these genres from the test set as the test data and remove its corresponding genre from the training and development sets.

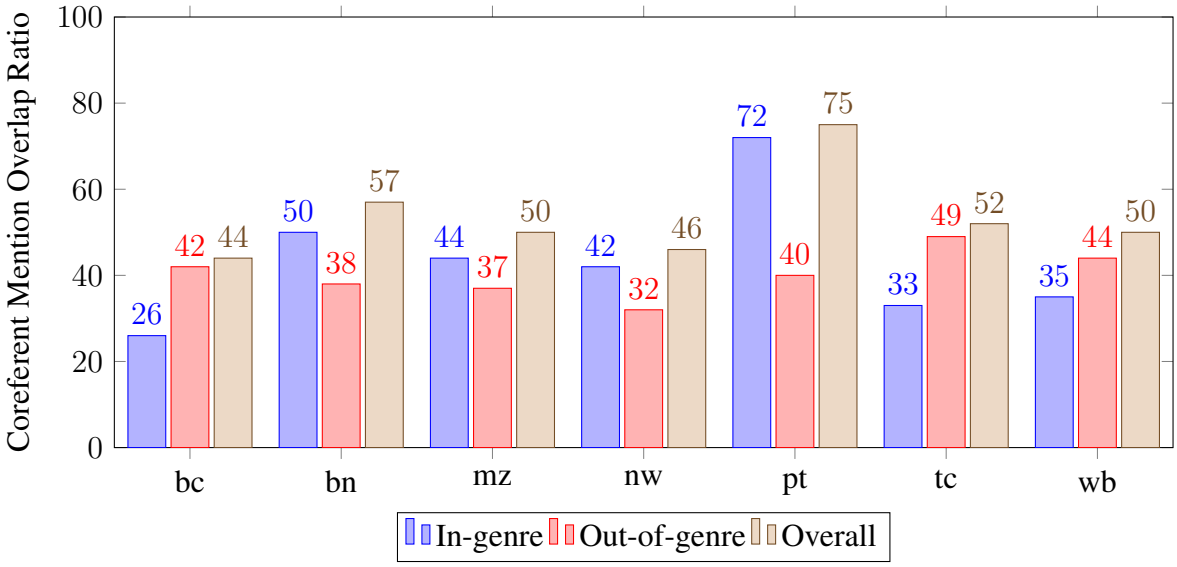


Figure 5.3: In-genre, out-of-genre and overall overlap ratios for non-pronominal coreferent mention strings of the development set and those of the training set.

We also compute the overlap ratios between the development and training sets. The results for non-pronominal coreferent mentions and mention heads are presented in Figure 5.3 and Figure 5.4, respectively.

Similar to the official CoNLL test set, coreferent mentions of the development set also have a great overlap with those of the training set. The development set is usually used for tuning the hyperparameters or for selecting the best trained model from different iterations of the training process. Because of this large overlap between the training and development sets, the best model selection can be biased towards models that memorize more specific properties of the training mentions instead of those that are focused on more general properties of coreference relations.

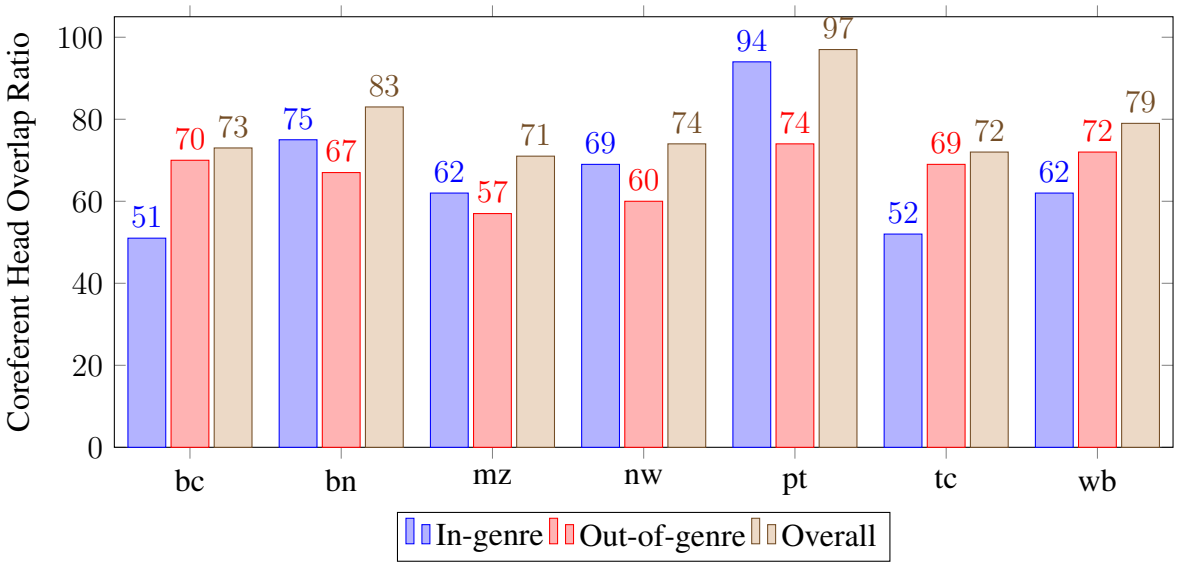


Figure 5.4: In-genre, out-of-genre and overall overlap ratios for non-pronominal coreferent mention heads of the development set and those of the training set.

5.2 Role of Lexical Features in Limited Generalization

The use of lexical features[3] in coreference resolution dates back to the early 2000s, e.g. Luo et al. (2004), Daumé III & Marcu (2005), Bengtson & Roth (2008). However, the contribution of lexical features was not significant to the overall performance of coreference resolvers. An effective use of lexical features requires a considerable number of annotated samples, which was not available before the introduction of the CoNLL dataset.

With the introduction of the CoNLL dataset in the CoNLL-2011 shared task, more systems start using lexical features, e.g. Björkelund & Nugues (2011), Rahman & Ng (2011), Fernandes et al. (2012), Björkelund & Farkas (2012). However, the efficacy of using lexical features in coreference resolution is mainly brought to attention by Durrett & Klein (2013).

Durrett & Klein (2013) present a simple coreference resolution system that uses a small number of lexical features. Their proposed system significantly outperforms previous state-of-the-art coreference resolvers. As mentioned in Section 2.4.2, Durrett & Klein (2013)'s lexical features describe shallow properties of mentions and their surrounding words by incorporating the head, first, last, preceding and following words of mentions (anaphor or antecedent). Durrett & Klein (2013)'s base system contains very few non-lexical features including exact string match, head match, mention length and distance features.

[3]We refer to features that include exact word forms or word embeddings as lexical features.

Durrett & Klein (2013) show that instead of hand-coding some linguistic phenomena like definiteness and syntactic roles, one can use simply lexical features. They note that in comparison to traditional features, lexical features can capture the examined phenomena at a finer level of granularity.

Recently, the shift from traditional features to lexical features goes to an extent that the non-lexical features that are used in the current state-of-the-art system of Lee et al. (2017) are only the distance between two mentions, the length of mentions, and speaker match.

In this section, we examine the role of lexical features in the limited generalization of lexicalized coreference resolvers. In other words, we investigate how much lexical features contribute to the fact that current improvements in coreference resolution do not properly apply to a new domain. We show that the extensive use of lexical features biases coreference resolvers towards the mentions that are seen during training.

5.2.1 Performance Drop in Lexicalized vs. Non-lexicalized Systems

In this section, we compare the performance drop in out-of-domain evaluations for two sets of coreference resolvers: (1) lexicalized systems: coreference resolvers that use lexical features, and (2) non-lexicalized systems: coreference resolvers that only use heuristic features.

The examined lexicalized systems are the surface and final models of the Berkeley coreference resolver, the ranking model of *cort*, and the ranking model of *deep-coref*. For in-domain evaluations we train *deep-coref*'s ranking model for 100 iterations, i.e. the setting used by Clark & Manning (2016b). However, we only train the model for 50 iterations in out-of-domain evaluations for the model to be less overfitted to the CoNLL dataset.

It is worth noting that in order to have a fair comparison, we only examine noun phrase coreference resolvers in this section. Therefore, we do not include *e2e-coref* that also resolves verb phrases in the experiments of this section.

We examined two non-lexicalized coreference resolvers: (1) Stanford's rule-based system and (2) *cort−lexical*. As mentioned in Section 2.4.3, *cort* uses both lexical features and a considerable number of non-lexical features. *cort−lexical* is a version of *cort* in which no lexical features are used.

The following observations from Table 5.2 and Table 5.3 are worth noting:

1. The performance of *deep-coref* is about nine (for *wb*) to 16 (for *mz*) percent better than those of *rule-based* on various genres in in-domain evaluations, where the test genre is included in the training data. However, the performance is only about one (for *pt*) to 12 (for *mz*) percent better when the training data does not include the test genre.

2. The results of the *pt* genre show that when there is a high overlap between the training and test datasets, the performance of all learning-based classifiers significantly im-

	CoNLL	LEA			CoNLL	LEA		
	Avg. F_1	R	P	F_1	Avg. F_1	R	P	F_1
	in-domain				out-of-domain			
pt (75% in-genre overlap ratio)								
rule-based	-	-	-	-	65.01	50.58	65.02	56.90
cort−lexical	69.48	54.26	70.33	61.26	64.32	45.63	68.51	54.77
berkeley-surface	69.15	58.57	65.24	61.73	63.01	46.56	62.13	53.23
berkeley-final	70.71	60.48	67.29	63.70	64.24	47.10	65.77	54.89
cort	72.56	61.82	70.70	65.96	64.60	46.85	67.69	55.37
deep-coref	75.61	68.48	73.70	71.00	66.06	52.44	63.84	57.58
bn (47% in-genre overlap ratio)								
rule-based	-	-	-	-	55.02	38.83	55.46	45.68
cort−lexical	59.21	42.73	63.45	51.07	58.76	42.06	63.20	50.51
berkeley-surface	59.00	44.35	58.14	50.32	57.3	41.82	57.44	48.40
berkeley-final	60.91	47.35	59.88	52.88	59.05	45.34	58.57	51.11
cort	61.88	46.06	65.57	54.11	60.45	44.59	65.73	53.13
deep-coref	65.27	51.71	66.31	58.10	61.12	46.89	61.40	53.17
nw (42% in-genre overlap ratio)								
rule-based	-	-	-	-	51.39	38.95	48.18	43.08
cort−lexical	58.71	43.95	61.91	51.41	57.57	41.88	61.70	49.89
berkeley-surface	54.69	40.29	54.48	46.32	51.03	35.94	50.71	42.07
berkeley-final	56.76	44.18	56.03	49.41	52.12	38.24	50.94	43.69
cort	59.92	45.60	61.98	52.54	57.51	41.49	62.37	49.83
deep-coref	61.54	49.92	62.06	55.34	56.90	44.35	55.96	49.48

Table 5.2: In-domain and out-of-domain evaluations for the genres of the CoNLL dataset. In in-domain evaluations the training data contains all genres including the test genre. In out-of-domain evaluations the testing genre is excluded from the training and development sets. The results are reported for genres in which the in-genre overlap ratios between the non-pronominal training and test mentions have higher values.

	CoNLL	LEA			CoNLL	LEA		
	Avg. F_1	R	P	F_1	Avg. F_1	R	P	F_1
	in-domain				out-of-domain			
tc (40% in-genre overlap ratio)								
rule-based	-	-	-	-	57.83	51.83	54.38	53.07
cort−lexical	62.90	60.13	58.47	59.29	60.54	56.50	55.42	55.95
berkeley-surface	61.50	54.31	54.58	54.44	58.79	51.21	50.63	50.92
berkeley-final	65.89	61.78	62.99	62.38	56.85	49.91	47.23	48.53
cort	64.32	58.19	64.44	61.15	60.65	51.98	59.96	55.68
deep-coref	68.49	62.58	71.95	66.90	61.24	61.29	53.49	57.13
bc (39% in-genre overlap ratio)								
rule-based	-	-	-	-	52.11	41.20	45.61	43.29
cort−lexical	56.00	43.52	55.27	48.69	55.75	43.30	54.72	48.34
berkeley-surface	54.82	41.53	52.58	46.41	55.20	43.53	49.98	46.53
berkeley-final	57.64	43.83	58.96	50.29	55.66	41.41	54.16	46.93
cort	57.98	44.33	59.35	50.75	57.26	42.80	59.21	49.68
deep-coref	60.38	45.86	63.65	53.31	58.23	46.28	55.39	50.43
mz (37% in-genre overlap ratio)								
rule-based	-	-	-	-	56.44	48.42	49.13	48.77
cort−lexical	62.36	53.16	58.45	55.68	62.05	53.53	57.38	55.39
berkeley-surface	62.73	54.25	57.49	55.82	61.17	52.92	54.99	53.94
berkeley-final	64.28	57.27	57.66	57.46	62.87	56.07	55.39	55.73
cort	65.18	56.52	61.55	58.93	63.87	55.09	60.01	57.45
deep-coref	70.36	59.52	71.34	64.89	67.28	61.77	60.61	61.19
wb (34% in-genre overlap ratio)								
rule-based	-	-	-	-	53.80	45.19	44.98	45.08
cort−lexical	56.83	51.00	47.34	49.10	57.10	51.50	47.83	49.60
berkeley-surface	56.37	45.72	47.20	46.45	55.14	45.94	44.59	45.26
berkeley-final	56.08	44.20	50.45	47.12	57.31	50.33	46.17	48.16
cort	59.29	50.37	51.56	50.96	58.87	51.47	50.96	51.21
deep-coref	61.46	48.04	60.99	53.75	57.17	50.29	47.27	48.74

Table 5.3: In-domain and out-of-domain evaluations for the genres of the CoNLL dataset. In in-domain evaluations the training data contains all genres including the test genre. In out-of-domain evaluations the testing genre is excluded from the training and development sets. The results are reported for genres in which the in-genre overlap ratios between the non-pronominal training and test mentions have lower values.

proves. *deep-coref* has the largest gain from including *pt* in the training data that is more than 13% based on the *LEA* score.

3. *cort−lexical* has the lowest performance drop when the evaluation setting changes from in-domain to out-of-domain for all genres[4].

4. *deep-coref* has the highest performance drop in out-of-domain evaluations[5]. Its performance drop is considerably higher than that of the other examined lexicalized coreference resolvers. This result indicates that as we use more complex neural networks, there is more capacity for brute-force memorization of the training dataset.

5. Performance gains and drops in out-of-domain evaluations are not entirely due to the use of lexical features, as the performance of *cort−lexical* also drops significantly in the *pt* out-of-domain evaluation. The classifier may also memorize other properties of the seen mentions in the training data. However, in comparison to features like gender and number agreements or syntactic roles, lexical features have the highest potential for overfitting.

5.2.2 Lexical Memorization

Based on the experiments of Table 5.2 and Table 5.3, *deep-coref* has the highest performance drop among the examined noun phrase coreference resolvers. We further analyze the output of this system on the development set. The *all* rows in Table 5.4 show the number of pairwise links that are created by *deep-coref* on the development set for different mention types. The *seen* rows show the ratio of each category of links for which the (antecedent head, anaphor head) pair is seen in the training set. All ratios are surprisingly high. The most worrisome cases are those in which both mentions are either a proper name or a common noun.

Table 5.5 further divides the links of Table 5.4 based on whether they are correct coreferent links. The results of Table 5.5 show that most of the incorrect links are also made between the mentions that are both seen in the training data.

These high ratios indicate that (1) there is a high overlap between the mention pairs of the training and development sets, and (2) even though *deep-coref* uses generalized word embeddings instead of exact surface forms, it is strongly biased towards the seen mentions.

We analyze the links that are created by Stanford's rule-based system and compute the ratio of the links that exist in the training set. Table 5.6 shows the ratios for the rule-based system.

[4]Except for the *bc* genre in which both *cort−lexical* and *berkeley-surface* have minor performance changes.

[5]The exception is the *tc* genre in which *berkeley-final* has the highest performance drop.

Ratios for pairs in which one mention is a proper name and another is a nominal are considerably lower than those of *deep-coref*. However, overall, corresponding ratios for the rule-based system are also very high, even though the rule-based system does not use the training data. This analysis emphasizes the overlap in the CoNLL datasets. Because of this high overlap, it is not easy to assess the generalization of a coreference resolver to unseen mentions on the CoNLL dataset given its official split.

We also compute the ratios of Table 5.5 for the recall errors of *deep-coref*, i.e. minimum number of links that are missing in the output. We compute the recall errors by *cort*'s error analysis tool (Martschat & Strube, 2014). Table 5.7 shows the corresponding ratios for recall errors. The lower ratios of Table 5.7 in comparison to those of Table 5.4 emphasize the bias of *deep-coref* towards the seen mentions during training.

		Anaphor		
Antecedent		Proper	Nominal	Pronominal
Proper	seen	80%	85%	77%
	all	3221	261	1200
Nominal	seen	75%	93%	95%
	all	69	1673	1315
Pronominal	seen	58%	99%	100%
	all	85	74	4737

Table 5.4: Ratio of links created by deep-coref for which the head-pair is seen in the training data.

As an example, *deep-coref*'s links on the development set include 31 cases in which both mentions are either proper names or common nouns and the head of one of the mentions is "country". For all these links, "country" is linked to a mention that is seen in the training data. Therefore, this raises the question how the classifier would perform on a text about countries that are not mentioned in the training data.

Memorizing mention-pairs in which one of them is a common noun could help the classifier to capture world knowledge to some extent. From seen pairs like (Haiti, his country), and (Guangzhou, the city) the classifier could learn that "Haiti" is a country and "Guangzhou" is a city. However, it is questionable how useful word knowledge is if it is mainly based on the training data.

The coreference relation of two nominal noun phrases with no head match can be very hard to resolve. The resolution of such pairs has been referred to as capturing semantic similarity (Clark & Manning, 2016a). *deep-coref* links 49 such pairs on the development set. Among all these links, only 5 pairs are unseen on the training set and all of them are incorrect links. The link between "one man in Moorhead Minnesota" and "the driver" is an example of such links.

The effect of lexical memorization is also analyzed by Levy et al. (2015) for hypernymy

		Anaphor		
		Proper	Nominal	Pronominal
Antecedent		Correct decisions		
Proper	seen	82%	85%	78%
	all	2603	150	921
Nominal	seen	76%	94%	96%
	all	42	1058	890
Pronominal	seen	63%	98%	100%
	all	49	44	3998
Incorrect decisions				
Proper	seen	73%	85%	76%
	a11	618	111	279
Nominal	sen	74%	92%	94%
	all	27	615	425
Pronominal	seen	50%	100%	100%
	all	36	30	739

Table 5.5: Ratio of links created by deep-coref for which head-pairs are seen in the training data.

		Anaphor		
Antecedent		Proper	Nominal	Pronominal
Proper	seen	80%	51%	79%
	all	3655	323	1351
Nominal	seen	59%	90%	94%
	all	139	2131	2141
Pronominal	seen	78%	92%	100%
	all	191	250	4966

Table 5.6: Number of links created by Stanford's rule-based system for which the head-pair exists in the training data.

		Anaphor		
Antecedent		Proper	Nominal	Pronominal
Proper	seen	63%	51%	75%
	all	818	418	278
Nominal	seen	44%	73%	90%
	all	168	892	538
Pronominal	seen	82%	90%	100%
	all	49	59	444

Table 5.7: Ratio of deep-coref's recall errors for which head-pairs exist in the training data.

detection and text entailment. They show that state-of-the-art classifiers memorize words from the training data. The classifiers benefit from this lexical memorization when there are common words between the training and test sets.

5.3 Summary

In this chapter, we show that there is a significant overlap between the training and validation sets in the CoNLL dataset. This large overlap makes it impossible to have a reliable evaluation on the official splits of this corpus.

As mentioned in Chapter 3, we introduce the *LEA* metric in order to make coreference evaluations more reliable. However, in order to ensure valid developments on coreference resolution, it is not enough to have reliable evaluation metrics. The validation set on which the evaluations are performed also needs to be reliable. A dataset is reliable for evaluations if a considerable improvement on this dataset indicates a better solution for the coreference problem instead of a better exploitation of the dataset itself.

Moreover, we show that the extensive use of lexical features biases coreference resolvers towards seen mentions. This bias holds us back from developing more robust and generalizable coreference resolvers. After all, while coreference resolution is an important step for text understanding, it is not an end-task. Coreference resolvers are going to be used in tasks and domains for which coreference annotated corpora may not be available. Therefore, developing generalizable systems should be brought to attention in developing coreference resolvers.

It is worth noting that we do not intend to argue against the use of lexical features. Especially, when word embeddings are used as lexical features. The incorporation of word embeddings is an efficient way for capturing semantic relatedness.

To ensure more meaningful improvements, incorporating out-of-domain evaluations in the current coreference evaluation scheme is a must. Out-of-domain evaluations could be performed by using either the existing genres of the CoNLL dataset or by using other existing coreference annotated datasets like WikiCoref.

Part III

How to Achieve Robust Improvements?

Chapter 6

Reconciling Coreference Resolution with Linguistic Features

The resolution of coreference relations requires many knowledge sources including surface properties, syntactic, contextual and semantic information, and world knowledge. Early coreference resolution systems use various information sources and complex linguistic features including syntactic and semantic parallelism, gender and number agreements, c-command constraints, e.g. Mitkov (2002), centering theory (Grosz et al., 1995), e.g. Brennan et al. (1987), Walker (1998), Kong et al. (2009), and world knowledge, e.g. Ponzetto & Strube (2006), Daumé III & Marcu (2005), Ng (2007), Bengtson & Roth (2008). However, state-of-the-art coreference resolvers no longer use such features and mainly rely on lexical features. Henceforth, we refer to features that are designed based on linguistic intuitions, e.g. string match and agreement features, or are acquired from a linguistic preprocessing module, e.g. part of speech tags and syntactic information, interchangeably as linguistic or traditional features. The use of the term "traditional" for linguistic features is to emphasize the fact that these features are fading out in recent systems.

Durrett & Klein (2013) show that lexical features are more effective at capturing a set of examined key linguistic phenomena, e.g. definiteness and syntactic roles, in comparison to the corresponding hand-engineered features. They refer to this success as "easy victories" in coreference resolution.

While Durrett & Klein (2013) argue that lexical features are better suited to model some linguistic phenomena in comparison to linguistically motivated heuristic features, they still leave a room for heuristic features in order to capture semantics. They refer to semantics in coreference resolution as "uphill battles". They conclude that the incorporation of semantics still requires complex heuristics and external knowledge sources.

Durrett & Klein (2013) use words as lexical features. However, current state-of-the-art coreference resolvers use word embeddings instead of words.

Word embeddings are shown to be effective at capturing syntactic and semantic information, e.g. Mikolov et al. (2013b), Mikolov et al. (2013a), are usually pretrained on large external corpora. Hence, the use of pretrained word embeddings introduces external knowledge to coreference resolvers to some extent. As a result, lexical features also fill the room that was left for traditional features for capturing semantics. The number of traditional features is now less than ten features in state-of-the-art coreference resolvers.

deep-coref (Clark & Manning, 2016a) and e2e-coref (Lee et al., 2017) are two recent state-of-the-art coreference resolvers that both use deep neural networks and word embeddings. The feature set of deep-coref includes string match, speaker match, whether a mention is nested into another mention, the type, position and length of mentions, the distance between two mentions based on the number of sentences and mentions, and the genre of the document. e2e-coref incorporates even fewer features including speaker match, the relative position of mentions, the distance between two mentions and the genre of the document. The rest of the required information for the complex task of coreference resolution is assumed to be implicitly inferred by deep neural networks and from word embeddings.

In this part of the dissertation, we examine the role of linguistic features in state-of-the-art coreference resolvers. We investigate the following two questions:

- **Are linguistic features still useful?**

  Existing works regarding the importance of linguistic features for coreference resolution are mainly based on analyses of simple coreference resolvers on small datasets, i.e. MUC or ACE. It is worth exploring the efficacy of linguistic features in the presence of word embeddings and deep neural networks and on larger corpora. We show the efficacy of linguistic features in state-of-the-art coreference resolvers in Section 8.5.

- **Do linguistic features improve generalization?**

  The use of lexical features as the main source of information for learning coreference relations results in a poor generalization to new domains (Section 5.2). A resolver that mainly relies on lexical features is biased towards seen mentions and does not perform well in out-of-domain evaluations. We show that the incorporation of linguistic features alongside lexical features is a promising direction for enhancing the generalization. The effect of linguistic features on generalization is examined in Section 8.6.

To show the efficiency of linguistic features in in-domain as well as out-of-domain evaluations, we first need to identify features that are useful for discriminating coreference relations. We describe our algorithm for finding discriminative features in Section 6.1.

6.1 Which Features to Use?

There is a large number of features that can be used for describing mentions or mention-pairs. For instance, each mention can be simply described by the part of speech (POS) tags of its containing words. The description can be improved by computing the syntactic role of mentions or the shortest syntactic path between two mentions. Type of mentions can be specified along the type of named entities. Similarly, one can use any other features to enrich the description of a mention pair. However, some of these features or feature-values are not discriminative for detecting coreferring relations. The incorporation of a large number of non-discriminating features may only confuse the classifier.

Some of the features may not be discriminative for coreference relations on their own. However, a coreference resolver can benefit from them by considering them in combination with other features. For instance, knowing that an anaphor is a direct object is not a discriminative feature for coreference resolution. However, this feature-value can become discriminative in combination with other feature-values, e.g. the anaphor is a pronoun and the candidate antecedent has the same dependency relation and it is also the nearest antecedent that has compatible gender, number and animacy information as those of the anaphor.

Given the above motivation, instead of incorporating a whole set of features, we only incorporate feature-values that are discriminative for resolving coreference relations, either individually or in combination with other feature-values. A viable solution for finding discriminative feature-values is to use a pattern mining approach (Cheng et al., 2007; 2008; Batal & Hauskrecht, 2010). In the following sections we describe our pattern mining approach for identifying discriminative feature-values.

6.1.1 Definitions

We use the following notations and definitions throughout this section:

- $D = \{X_i, c(X_i)\}_{i=1}^{n}$: set of n training samples. X_i is the set of feature-value pairs that describes the ith sample. $c(X_i) \in C$ is the label of X_i. For instance, in the experiments of Section 8, each sample is a mention-pair and each feature describes a property of anaphor, antecedent, or their pairwise relation. The corresponding label of each sample is either coreferent or non-coreferent.
- $A = \{a_1, \ldots, a_l\}$: set of all feature-values present in D. We call each $a_i \in A$ an item. For instance, "anaphor type=proper" is an item.
- p: pattern $p = \{a_{i_1}, \ldots, a_{i_k}\}$ is a set of one or more items. For instance, p_1 ={"anaphor type=proper", "antecedent type=proper"} is a pattern.
- $D_p = \{X_i | p \in X_i\}$: the set of samples that is matched by pattern p, referred to as cover p.

For instance, D_{p_1} is the set of all samples in which both anaphor and antecedent are proper names.

- $support(p, c_i)$: the number of samples in D_p that are labeled with c_i, i.e. $|\{X_i | X_i \in D_p \wedge c(X_i) = c_i\}|$. For instance, $support(p_1, \text{coreferent})$ is the number f coreferring mention pairs in which both anaphor and antecedent are proper names.

6.1.2 Using Pattern Mining for Finding Informative Features

Using pattern mining for generating classification features originates from the association rule mining (e.g. Liu et al. (1998) and Li et al. (2001)). In this line of research, supervised pattern set mining approaches are used to construct new features for classification algorithms.

A systematic exploration of using frequent patterns in classification is conducted by Cheng et al. (2007). They use the discriminative power of patterns (evaluated by information gain or Fisher score), in order to choose a subset of frequent patterns that are useful for classification. They show that the incorporation of discriminative frequent patterns as new features can significantly improve the performance of the baseline classifier. This finding is also supported in other studies, e.g. Cheng et al. (2008), Batal & Hauskrecht (2010).

The reason for using frequent patters, instead of all patterns, is that generating all possible patterns is not computationally tractable due to combinatorial explosion. Besides, rare patterns have limited discriminative power (Cheng et al., 2007) and may lead to over-fitting. Even if a pattern with low frequency is useful for classifying few training samples, it may have limited ability to generalize. In this regard, frequent patterns have been introduced as promising candidates to be used as features that can result in more accurate models (Cheng et al., 2007).

Mining a discriminative and non-redundant set of patterns from labeled data is a well-established problem that is referred to as supervised pattern set mining. There is a large number of supervised pattern set mining approaches, e.g. Bringmann et al. (2009), Novak et al. (2009), Zimmermann & Nijssen (2014), inter alia, that are mainly grouped in two categories: (1) post-processing, and (2) iterative approaches.

Post-processing approaches first mine a set of patterns satisfying certain constraints, which is usually the minimum frequency. The mining step is then followed by a post-processing step which selects a subset of mined patterns based on discriminative power and redundancy criteria. Iterative approaches mine one or more patterns satisfying certain constraints. Based on the selected pattern(s) they modify the constraints or data and then repeat the mining process. Zimmermann & Nijssen (2014) include a detailed discussion about these two categories.

In natural language processing tasks, it is often the case that the dataset is very large and the data includes a large set of feature-values. Therefore, efficiency is a main concern for applying a pattern mining approach to NLP tasks.

Post-processing approaches are generally faster than iterative ones because they only need

one iteration of the mining process.

However, the post-processing approaches are still not efficient enough for large data sizes or large search spaces. Enumerating all frequent patterns of data has been proven to be an NP-complete problem (Yang, 2006). Besides, when the search space is large, the mining step may result in a huge number of patterns, which in turn makes post-processing also a time-consuming step.

In this section, we introduce an efficient post-processing approach for supervised pattern set mining. We consider two variations of the discriminative power and redundancy measures: (1) lenient variants that are directly checked during the mining step on individual patterns, and (2) strict variants that are checked in the post-processing step. Since we also use lenient variations of discriminative power and redundancy measures during the mining step, the number of pattens for the post-processing step decreases considerably and therefore the time efficiency of the whole algorithm increases.

6.1.3 Data Structure

For representing the input samples, we use the Frequent Pattern Tree (FP-Tree) structure that is the data structure of the FP-Growth algorithm (Han et al., 2004), i.e. one of the most common algorithms for frequent pattern mining. FP-Tree provides a structure for representing all existing patterns of data in a compressed form. Using the FP-Tree structure allows an efficient enumeration of all frequent patterns. In the FP-Tree structure, items are arranged in descending order of frequency. Frequency of an item corresponds to $\sum_{c_i \in C} \ support(a_i, c_i)$, i.e. the frequency of the item a_i in all class labels.

Except for the root of the tree, which is a null node, each node n contains an item $a_i \in A$. It also contains the support values of a_i in the subpath of the tree that starts from the root and ends with n. These support values are indicated by $support_n(a_i, c_j)$.

The FP-Tree construction method (Han et al., 2004) is as follows:

1. Scan D once to collect the set of all items, i.e. A. Compute $support(a_i, c_j)$ for each item $a_i \in A$ and label $c_j \in C$. Sort A's members in descending order according to their frequencies, i.e. $\sum_{c_i \in C} \ support(a_i, c_i)$.

2. Create a null-labeled node as the root.

3. Scan D one more time. For each $(X_i, c(X_i)) \in D$ perform the following steps:

   (a) Order all items $a_j \in X_i$ according to the order in A.

   (b) Set the current node (T) to the root.

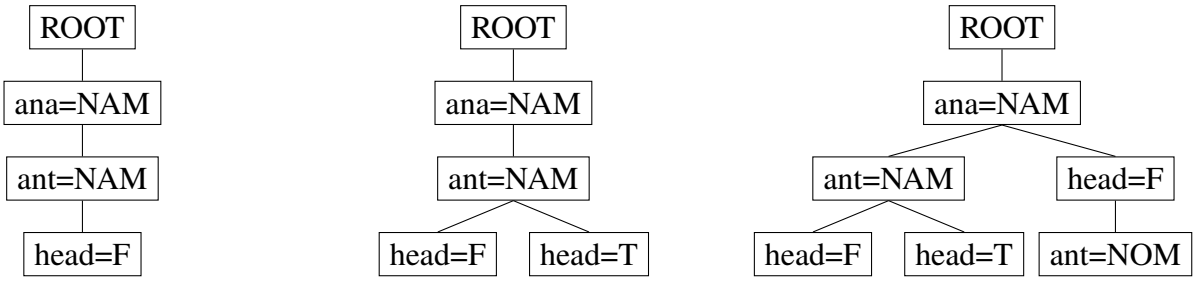


Figure 6.1: Left to right: (partial) constructed FP-Tree after adding each of the three given samples. The right-most tree is the final FP-Tree that represents all input samples.

(c) Consider $X_i = [a_k|\bar{X}_i]$, where a_k is the first (ordered) item of x_i, and $\bar{X}_i = X_i - a_k$. If T has a child n that contains a_k then increment $support_n(a_k, c(X_i))$ by one. Otherwise, create a new node n that contains a_k with $support_n(a_k, c(X_i)) = 1$. Add n to the tree as a child of T.

(d) If $\bar{X}_i$ is non-empty, set T to n. assign $X_i = \bar{X}_i$ and go to step 3c.

From an initial FP-Tree (T) that represents all existing patterns, one can easily obtain a new FP-Tree in which all patterns include a given pattern p. This can be done by only including sub-paths of T that contain pattern p. The new tree is called conditional FP-Tree of p, T_p.

Running Example Assume D contains the following three samples:

X_1={ana-type=NAM, ant-type=NAM, head-match=F}, $C(X_1) = 0$

X_2={ana-type=NAM, ant-type=NAM, head-match=T}, $C(X_2) = 1$

X_3={ana-type=NAM, ant-type=NOM, head-match=F}, $C(X_3) = 0$

Based on these three samples

- A={ana-type=NAM, ant-type=NAM, head-match=F, head-match=T, ant-type=NOM},
- $support(a_i, 0)_{a_i \in A}$= {2,1,2,0,1}, e.g. "ana-type=NAM" appeared two times in non-coreferring ($C(X_i) = 0$) samples,
- and $support(a_i, 1)_{a_i \in A}$={1,1,0,1,0}.

If we sort A based on a_i's frequencies, i.e. $support(a_i, 0) + support(a_i, 1)$, the ordering of A's items will remain the same.

Now, we need to go through the samples again to build the tree. The FP-Tree construction steps after adding each of the above samples is demonstrated in Figure 6.1. ana-type=NAM, ant-type=NAM, head-match=F, head-match=T, and ant-type=NOM are abbreviated as ana=NAM, ant=NAM, head=F, head=T, and ant=NOM, respectively in Figures 6.1 and

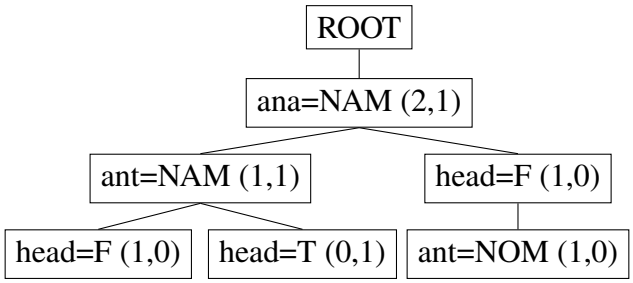


Figure 6.2: FP-Tree with corresponding support values of the nodes.

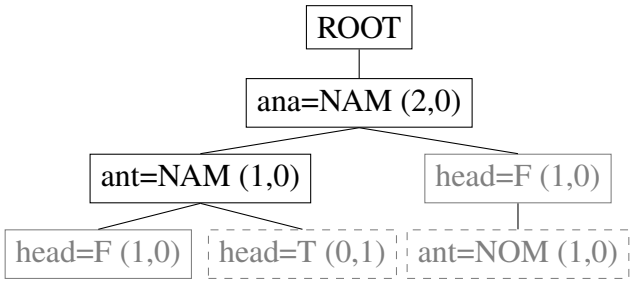


Figure 6.3: Conditional FP-Tree for the $p = \{\text{head=F}\}$ pattern.

6.2. Figure 6.2 shows the resulted FP-Tree in which corresponding support values for both classes, i.e. zero and one, are also included in each node.

Figure 6.3 and Figure 6.4 show the conditional FP-Trees that are built based on the FP-Tree of Figure 6.2 and for patterns $p = \{\text{head=F}\}$ and $p = \{\text{head=F}, \text{ant=NAM}\}$, respectively.

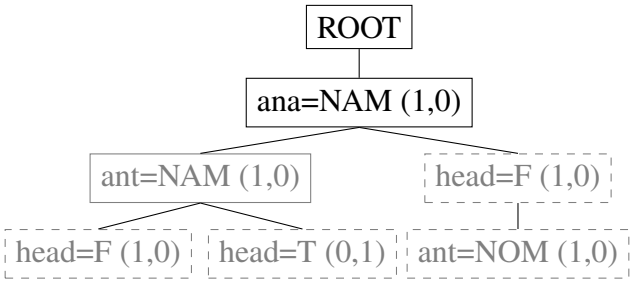


Figure 6.4: Conditional FP-Tree for the $p = \{\text{head=F}, \text{ant=NAM}\}$ pattern.

6.1.4 Which Pattern is Informative?

Similar to other supervised pattern mining approaches, our goal is to select a set of patterns that is discriminative regarding the class label. Selected patterns also should not include redundant information. We use a *discriminative power* and an *information novelty* measure in order to determine informative patterns. Similar to other pattern-based feature mining approaches, we also use a *frequency* measure. Frequency can be used to limit the search space if we are not interested in rare patterns of data. We use the statistical significance of the association of a pattern and the class label as the measure for choosing discriminative patterns. We use the binomial test for choosing patterns with novel information. The binomial test is successfully used in previous work for mining discriminative patterns (Batal & Hauskrecht, 2010).

The *discriminative power* and *information novelty* measures are partly checked during the mining process and also in the post-processing step. The difference is that the evaluation of the measures in the post-processing step is done in a more strict way.

Frequency: A pattern p is frequent if one of the following equations holds:

$$\exists c_i \in C \quad \frac{support(p, c_i)}{|\{X_i | c(X_i) = c_i\}|} \geq \lambda_1 \tag{6.1}$$

$$\exists c_i \in C \quad support(p, c_i) \geq \lambda_2 \tag{6.2}$$

λ_1 and λ_2 are the minimum support thresholds. λ_1 is specified as a fraction, i.e. $0 < \lambda_1 < 1$, and λ_2 is an absolute number, i.e. $\lambda_2 \geq 1$.

For instance, if $\lambda_1 = 0.1$ in Equation 6.1, p is frequent if it occurs in at least ten percent of the samples that belong to one of the classes. This definition is especially useful in imbalanced datasets. For Equation 6.2, p should occur at least λ_2 times in the samples of one of the classes to be considered as frequent.

Discriminative power: We use the G^2 likelihood ratio test (Agresti, 2007) in order to choose patterns whose association with the class variable is statistically significant. G^2 is an approach for hypothesis testing, and it has been successfully applied to text analysis (Dunning, 1993).

Consider Table 6.1 as a sample contingency table, e.g. X and Y can be the corresponding variables for a pattern (p) and the coreference label, respectively.

	Label	
	Y	$\bar{Y}$
X	n_{11}	n_{12}
$\bar{X}$	n_{21}	n_{22}

Table 6.1: A sample contingency table.

Assume $\{n_{ij}\}$ denote cell counts. $\pi_{ij} = Pr(X = i, Y = j)$ is the probability that (X, Y) falls in the cell that is in row i and column j. Let n be the total sample size, i.e. $\sum n_{ij}$.

Expected frequencies are then computed as $\{\mu_{ij} = n\pi_{ij}\}$. The G^2 likelihood ratio statistic is defined as:

$$G^2 = 2\sum n_{ij} log(\frac{n_{ij}}{\mu_{ij}}) \tag{6.3}$$

The larger values of G^2 provide stronger evidence against the null hypothesis. In our case, the null hypothesis is that X, i.e. an examined pattern, is independent of Y, i.e. the coreference variable.

G^2 can be unreliable for expected frequencies of less than five (Agresti, 2007). However, since we experiment on large datasets and evaluate the significance measure on patterns satisfying the frequency condition (Equation 6.1 or 6.2), this problem does not apply in our case. If one is interested in rare patterns of data, Fisher's exact test is a better choice. However, Fisher's exact test is very slow in comparison to the G^2 test for large values of n_{ij}. During the mining process, the discriminative condition is applied in a more lenient way. A pattern is considered discriminative if the p-value returned by the G^2 test is less than a fixed threshold (i.e. 0.01).

Information Novelty: A large number of patterns can be generated by adding irrelevant items to a base pattern that is discriminative itself. This can lead to a large set of redundant patterns conveying similar information regarding the class label. Many of these patterns may also be significant by themselves. However, the information they provide is redundant in comparison to the base pattern. In order to evaluate the information novelty of a pattern we check the following two conditions:

1. For pattern p, assume that D_p contains N samples, out of which N_c samples belong to target class c. Let P_c be the highest probability achieved by an included item in pattern p, i.e. $P_c = max_{a_i \in p} Pr(c|D_{a_i})$. The null hypothesis is that N_c is generated from N with a binomial distribution with probability P_c. The alternative hypothesis is that the underlying probability that generates N_c is significantly higher than P_c. The p-value of a one-sided significance test using a binomial distribution should be smaller than a significance level α, i.e. $Pr_{binomial}(x \geq N_c|N, P_c) < \alpha$. We set $\alpha = 0.01$ in our experiments. This condition is checked directly during the mining step.

2. The second condition is similar to the first one. However, instead of individual items that are included in p, we evaluate p against all its sub-patterns that are discriminative (satisfying *discriminative power*), frequent (satisfying *frequency*) and novel according to their containing items (satisfying the first information novelty condition). Therefore, for the second condition, P_c is the maximum probability of all the $p' \subset p$ that have the above three conditions. This condition is checked in the post-processing step.

6.1.5 Mining Algorithm

Our mining algorithm, i.e. Efficient Pattern Miner (EPM), is summarized in Algorithm 2. It takes FP-Tree T, pattern p on which T is conditioned, and items ($A_j \subset A$) whose combinations with p will be examined. Initially, p is empty and the FP-Tree is constructed based on all frequent items of data and $A_j = A$.

For each $a_i \in A_j$, the algorithm builds new pattern q by combining a_i with p. $frequent(q)$ checks whether q meets the frequency condition of Equation 6.1 or 6.2. If q is frequent, the algorithm continues the search process. Otherwise, q itself is not qualified as a useful pattern and it is not possible to build any interesting pattern out of q.

Discriminative power and the first condition of *information novelty* are then checked for pattern q.

Input: T: input $FP\text{-}Tree$
Input: p: pattern base on which T is conditioned
Input: A_j: set of items to be combined with p
Output: P: set of output patterns
Algorithm EPM (T, p, A_j)

```
foreach a_i ∈ A_j do
    q = p ∪ {a_i}
    if Frequent(q) then
        if Discriminative(q) then
            testCount(|q|)+ = 1
            if Novel(q) then
                P = P ∪ q
            end
        end
        if |q| >= Θ_l then
            continue
        end
        construct T_q = q's conditional tree
        EPM(T_q, q, ancestors(a_i))
    end
end
```

Algorithm 2: The EPM algorithm

We use a threshold (Θ_l) for the maximum length of mined patterns. Θ_l can be set to large values if more complex and specific patterns are desirable. This threshold considerably

reduces the search space, especially for datasets with many features.

If $|q|$ is smaller than Θ_l, the conditional FP-Tree T_q is built that represents patterns of T that belongs to D_q, i.e. patterns of T that are matched by pattern q. The mining algorithm then continues to recursively search for more specific patterns by combining q with the items included in $ancestors(a_i)$. $Ancestors$ is built while constructing the original FP-Tree. For each item a_i, $ancestors$ keeps the list of all ancestors of a_i in the original FP-Tree.

It is worth noting that all frequent patterns of up to length Θ_l will be examined by our mining algorithm.

6.1.6 Post-Processing

The post-processing step checks *discriminative power* and *information novelty* measures in a more strict way.

If we use a statistical test multiple times, the risk of making false discoveries increases (Webb, 2006). In the post-processing step, we, therefore, apply the Bonferroni correction for multiple tests in order to reduce search errors. Similar to Bay and Pazzani (2001), we set the p-value threshold for all patterns of length l as follows:

$$\Theta_{p_l} = min(\frac{\Theta_{p_0}}{2^l \times testCount(l)}, \Theta_{p_{l-1}}),$$

where $testCount(l)$ is the number of times that the G^2 test is applied on a pattern of length l in the mining step.

Now, we have the set of frequent and discriminative patterns of data up to length Θ_l. Therefore, we can easily check the second information novelty condition of Section 6.1.4 on the resulting pattern set.

6.1.7 Extracting Useful Feature-Values from Informative Patterns

Instead of using informative patterns, i.e. patterns that are frequent, discriminative and novel, as new features, we use the set of mined patterns to select feature-values that appear in at least one informative pattern.

The reason is that recent coreference resolvers use deep neural networks and they have a fully automated feature generation process, which is referred to as representation learning (Bengio et al., 2013). Therefore, we use our algorithm to choose informative values, if any, of the existing features. For instance, instead of using the pattern {“nearest compatible antecedent=true”, “antecedent dependency=subject”, “anaphor=he”} as one feature, we incorporate the three feature-values that are included in this pattern as individual features. If a feature does not occur in any informative pattern, we exclude it from the set of features.

6.2 Historical Value of Features in Coreference Resolution

Uryupina (2007)'s thesis is one of the most thorough analyses of linguistically motivated features in learning-based coreference resolvers. She examines a large set of linguistic features that are suggested by theoretical studies and investigates their interaction with coreference relations. In particular, she examines string match, syntactic knowledge, semantic compatibility, discourse structure and salience features for individual mentions and mention-pairs and analyzes their effect on coreference resolution. She shows that even imperfect linguistic features that are automatically extracted using error-prone preprocessing modules, boost the performance of coreference resolvers. As a result, she argues that coreference resolvers could and should benefit from linguistic theories.

The examined features set by Uryupina (2007) includes:

- **String match features:** The naive string match feature, i.e., whether the whole string of one mention matches the whole string of the other, is extended by
  - considering normalized forms of a string by lowercasing, removing punctuations, or removing determiners
  - considering only a subpart of a string, i.e. first, head, last, and the least frequent word in string match features
  - considering various forms of matching functions including exact match, the minimum edit distance, the number of overlapping words and whether one string is the abbreviation of the other

  Uryupina (2007) shows that sophisticated and carefully tuned string match features can drastically improve the performance in their experiments.

  The importance of well-designed string match features is corroborated by the success of the Stanford rule-based system (Raghunathan et al., 2010; Lee et al., 2011). The success of the rule-based system is mainly attributed to its sieve architecture. In Moosavi & Strube (2014), we use Stanford's features in an unsupervised setting in which all features are used in a single step. Our single-step approach leads to on-par performance with that of the sieve model. As a result, the rule-based system owes its success to its well-tuned features and as described in Section 2.4.1, these are mainly string match features.

- **Syntactic knowledge:** Uryupina (2007) examines various syntactic properties of mentions including (1) mention type, (2) the inclusion of determiners in mentions, (3) head word of mentions, (4) the internal structure of mentions, e.g. inclusion of coordination, pre-modifiers, and post-modifiers, (5) command relations (Barker & Pullum, 1990), (6) apposition, (7) predicate nominative, (8) syntactic roles, and (9) number and gender agreements. The syntactic features only help the resolution of few mentions.

- **Semantic information:** Semantic information is encoded using gender match and WordNet-based features. WordNet features include:
  - same semantic class: the first WordNet senses of head words are exactly the same
  - compatible semantic class: the first WordNet senses of head words are in a hyponymy/hyperonymy relation
  - WordNet similarity: the tree-based distance between synsets of head words

  The semantic features over-relate too much and therefore decrease precision.

- **Discourse structure and salience:** Discourse-level features contain:
  - discourse level: whether a mention is within a sub-discourse [1]
  - proximity: distance based on the number of mentions, sentences, and paragraphs between two mentions
  - backward-looking center (CB): A discourse includes a sequence of utterances, i.e. $U_1, \ldots, U_n$. According to the Centering theory (Grosz et al., 1995), each utterance has a ranked list of forward-looking centers ($CF(U_i)$). $CF(U_i)$ is the list of discourse entities that are directly realized in U_i. The backward-looking center is the highest-ranked element of $CF(U_i)$ that is also realized in U_{i-1}. Based on Uryupina (2007)'s analyses, CB is a reliable indicator for mentions to be antecedents. However, it has a low coverage.
  - salience: salience is encoded using features like whether an antecedent is a subject, the first mention of a sentence, the first mention of a paragraph, a CB, or the closest mention to the examined anaphor. Uryupina (2007) finds that the combination of proximity, salience and agreement features is an effective factor for guiding the search through likely antecedents, especially for pronouns.
  - coreferential properties of candidate antecedents: this information is modeled using three types of features: (1) whether the antecedent is discourse-old, (2) the coreference chain size of the antecedent and (3) a set of features describing the nearest antecedent of the candidate antecedent.

    The effectiveness of coreferential information of candidate antecedents for pronoun resolution is also investigated by Yang et al. (2004). They include a set of backward features describing lexical, syntactic and semantic properties of the closest antecedents of candidate antecedents. The efficacy of backward features

[1] A naive algorithm is used for sub-discourse identification. Explicitly quoted opinions are specified using a regular expression. Sentences that have the verb "say" in their main clause are identified as implicitly quoted opinions.

depends on the accuracy of determining the closest antecedents of candidate antecedents. However, overall, they show that the coreferential information of candidate antecedents is a useful source of information for pronoun resolution.

Overall, Uryupina (2007) shows that discourse properties and salience features alone are very effective for pronoun resolution. They are also useful for other types of noun phrases when they are used in combination with other knowledge sources.

It is notable that Uryupina (2007) claims are based on analyses on the MUC dataset. Apart from this thesis, there are also other studies regarding the importance of individual features in coreference resolution. Ng & Cardie (2002) are the first to investigate the incorporation of various knowledge sources in a machine learning based coreference resolver. They first incorporate a set of 53 new features including:

- various string match features, e.g. strings match and both mentions are pronouns, or substrings match and both mentions are non-pronominal
- mention type information, e.g. both mentions are prenominal modifiers, or both mentions are definite noun phrases
- syntactic role of mentions, e.g. one of the mentions is a subject or both mentions are the grammatical subject
- agreement features, e.g. gender, number and animacy agreements
- heuristic features regarding the grammatical form of mentions, e.g. a mention is a part of a quoted string or is nested in other mentions
- semantic features, e.g. semantic class compatibility and WordNet distance
- distance between two mentions
- coreference decisions of a naive pronoun resolver and a rule-based coreference resolver

According to Ng & Cardie (2002)'s experiments, the performance of the base coreference resolver drops significantly after the incorporation of all features. They mention that the overall performance drop is partially due to the performance drop on the resolution of common nouns. Hence, they manually select a smaller subset of the initial feature set in a way to improve precision for common noun resolution. For instance, none of their semantic features are included in the refined set. They show that the manually selected features significantly improve the performance. This shows the importance of removing irrelevant features that can mislead the classifier. Ng & Cardie (2002) left the automation of the precision-oriented feature selection for future work.

Ponzetto & Strube (2006) incorporate semantic and world knowledge in coreference resolution. They enrich the mention-pair representation by extracting additional information from WordNet and Wikipedia. They use the extracted information for computing various similarity measures for a given mention-pair. The semantic role information of each mention is also included in the features set. They use a feature selection algorithm to determine useful features. The suggested semantic features improve recall. The WordNet and Wikipedia features increase the performance on common nouns while the semantic role information improves the resolution of pronouns.

Bengtson & Roth (2008) also point out the importance of features in coreference resolution by showing that if we use a well-designed feature set, a simple mention-pair model will outperform more complex models. Their features include string match, lexical and semantic information.

Above studies are mainly focused on features for English coreference resolution. There are also studies for the importance of features for coreference resolution in languages other than English. Recasens & Martí (2009) is an example for the Spanish language.

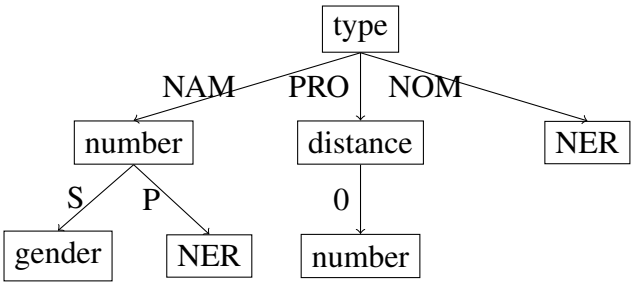
Apart from the above studies, which are mainly about the importance of linguistically motivated features, there are other works in which new features are generated automatically by combining basic features of mentions or mention-pairs, e.g. Björkelund & Farkas (2012), Fernandes et al. (2012), Uryupina & Moschitti (2015). For instance, based on "mention type" and "string match" features, one can create new features by combining these two basic features, i.e. "anaphor mention type $\wedge$ string match", "antecedent mention type $\wedge$ string match" and "anaphor mention type $\wedge$ antecedent mention type $\wedge$ string match". If "mention type" and "string match" have four and two different values, respectively, the "anaphor mention type $\wedge$ string match" feature will have eight values, i.e. four times two.

Björkelund & Farkas (2012) do not follow a systematic approach to create feature combinations. They evaluate a large number of feature combinations and select the subset which performs the best. They note that feature combination is crucial for their overall performance.

Fernandes et al. (2012) and Uryupina & Moschitti (2015) on the other hand, follow a systematic approach for creating combinatorial features.

Fernandes et al. (2012) use the Entropy guided Feature Induction (EFI) approach (Fernandes & Milidiú, 2012) to automatically generate discriminative feature combinations. The first step in this approach is to train a decision tree on a dataset. Each sample of the dataset consists of the basic features related to the decision variable of the prediction problem. For instance, for coreference resolution, each sample consists of features describing a mention pair.

Once the tree is built, the EFI approach traverses the tree from the root node in a depth-first order and recursively builds conjunction features. Figure 6.5 demonstrates a sample decision tree and the set of feature conjunctions that are extracted based on this tree. In constructing the conjunction features, Fernandes et al. (2012) abstract over feature values, i.e. ignore the



type+number

type+number+gender

type+number+NER

type+distance

type+distance+number

type+NER

Figure 6.5: A sample decision tree and the list of all extracted feature conjunctions based on Fernandes et al. (2012)'s approach.

values of connecting edges. Each extracted feature conjunction will be used as a template feature. By considering all value combinations of basic features, Fernandes et al. (2012) use each feature template for generating numerous binary features. for instance, if "type" has four values and number has three values, "type+number" results in 12 new binary features.

In order to limit the maximum template length, Fernandes et al. (2012) prune the initial decision tree at the depth five.

In comparison to EPM, the entropy guided feature induction approach also uses a tree to explore combinations of features. However, Fernandes & Milidiú (2012) use a decision tree. A decision tree does not represent all patterns of data. Therefore, it is not possible to explore all feature combinations from a decision tree. Each pattern that is generated by the EFI approach starts from the root node. Therefore, EFI tends to generate long patterns. Finally, Björkelund & Farkas (2012) use the EFI approach in order to generate templates for feature combinations instead of exploring informative feature-values.

Uryupina & Moschitti (2015) propose an alternative approach to EFI. Similar to our work, they formulate the problem of automatically generating feature combinations as a pattern mining approach.

Uryupina & Moschitti (2015) use the Jaccard item mining (JIM) algorithm[2] (Segond & Borgelt, 2011) for mining patterns of feature-values. They abstract away the set of mined patterns to generate feature templates, e.g. pattern {type=proper, number=singular} is abstracted to the feature template "type+number". Similar to Fernandes et al. (2012), all feature templates are then converted to binary features.

Uryupina & Moschitti (2015) compare the efficacy of features that are generated by JIM in comparison to those of EFI using the same base classifier. Based on their experiments, EFI

[2]Available at `http://www.borgelt.net/jim.html`

generated around 20 times more features in comparison to JIM. Besides, the EFI generated features tend to be longer than those of JIM. Uryupina & Moschitti (2015) show that the base classifier that uses the JIM features significantly outperforms the one that incorporates the EFI features.

The Jaccard item mining approach uses the Jaccard index, for selecting useful patterns. For a given pattern, the Jaccard index is defined as:

$$J(p) = \frac{\cap_{a_i \in p} D_{a_i}}{\cup_{a_i \in p} D_{a_i}} \tag{6.4}$$

where D_{a_i} is the set of samples that include a_i.

The Jaccard index is a redundancy measure. It is not an indicator of the relevance of a given pattern to the class label. In order to make the mined patterns more discriminative regarding the class label, Uryupina & Moschitti (2015) use a heuristic approach. They mine two separate sets of patterns from positive and negative samples. Let $J^+(p)$ and $J^-(p)$ be the Jaccard index of pattern p in positive and negative sets of samples, respectively. Uryupina & Moschitti (2015) use the following two scores for ranking the mined patterns:

$$score^+(p) = \frac{J^+(p)}{J^-(p)} \tag{6.5}$$

$$score^-(p) = \frac{J^-(p)}{J^+(p)} \tag{6.6}$$

They then use patterns for which $score^+$ or $score^-$ are above predefined thresholds. Similar to Björkelund & Farkas (2012), Uryupina & Moschitti (2015) use patterns for generating feature templates. We compare the efficacy of Uryupina & Moschitti (2015)'s patterns to those of EPM in Section 8.5.4.

6.3 Summary

In this chapter, we briefly reviewed related work regarding the importance of features in coreference resolution. Based on the literature, there is a general agreement that features are important for coreference resolution. However, the efficacy of features is evaluated based on various corpora and different coreference resolvers. It is not clear whether the examined features are beneficial for all corpora and all coreference resolvers.

We present an efficient method to automatically investigate the efficacy of various features for discriminating coreference relations in given corpora and independently from baseline coreference resolvers.

In the following two chapters we perform two sets of experiments:

1. In chapter 7, we evaluate EPM on standard machine learning datasets. We show that EPM is efficient, and is therefore applicable to large NLP corpora. Besides, we show that EPM's patterns are discriminative and their incorporation improves the accuracy of the baseline classifier. In order to evaluate the discriminative power of patterns, we use a linear classifier and incorporate the mined patterns as new features. We then evaluate whether new features improve the overall performance of the base classifier. We compare EPM with two other supervised pattern set mining approaches that are shown to be efficient and result in discriminative patterns.

2. We evaluate EPM's patterns in coreference resolution. For coreference resolution experiments, the data includes all coreferring and non-coreferring mention-pairs of the CoNLL training set. The mined patterns are a set of informative features for coreference relations. Since we use a deep neural network as the baseline classifier, which performs the feature combination automatically, we do not include each pattern as a new feature. Instead, as described in Section 6.1.7, we employ mined patterns for recognizing the set of informative feature-values, i.e. we only incorporate feature-values that appear in at least one informative pattern. We show that linguistic features are still beneficial for state-of-the-art coreference resolvers if we select an informative subset of feature-values (Section 8.5). Finally, we show that the incorporation of these features significantly improves the generalization of the baseline coreference resolver across domains (Section 8.6).

Chapter 7

Evaluation on General Benchmarks

In this section, we evaluate the EPM algorithm in terms of mining time efficiency and classification accuracy on standard machine learning datasets. We compare EPM with two other algorithms that also mine a discriminative set of patterns for improving the performance of baseline classifiers. We show that EPM scales best and compares favorably based on accuracy. Due to its efficiency, EPM can handle large datasets similar to ones that are commonly used in various NLP tasks.

7.1 Compared Methods

We compare EPM with two other discriminative pattern mining approaches that are shown to be more efficient than the post-processing approaches, i.e. Minimal Predictive Patterns (MPP) and Direct Discriminative Pattern Mining (DDPMine). MPP (Batal & Hauskrecht, 2010) directly mines a compact set of predictive patterns. MPP does not have several iterations of the mining algorithm or any post-processing step. It selects patterns that are significantly more predictive than all of their sub-patterns. For each pattern of length l, there are $2^l - 1$ sub-patterns to be checked. MPP's pattern selection criterion is as follows:

$$\exists c_i \in C \quad Pr(c_i|D_p) \succ max_{p' \subset p}\, Pr(c_i|D_{p'}) \tag{7.1}$$

where the probability $Pr(c_i|D_p)$ is used as the discriminative power of pattern p for label c_i. $\succ$ indicates that $Pr(c_i|D_p)$ significantly improves the right term of Equation 7.1 and the significance is measured by the binomial distribution.

DDPMine (Cheng et al., 2008) is an iterative approach. In practice, it requires a very small number of iterations over input samples and it progressively shrinks the search space. As the name implies, DDPMine select patterns according to their discriminative power that is evaluated by information gain. At each iteration, the algorithm selects the most discriminative pattern. Then the search space of the next iteration is reduced by removing all samples that

are covered by the selected pattern. This process continues iterating until all of the samples are covered by at least one discriminative pattern or no more pattern can be mined.

EPM and MPP evaluate the discriminative power of each pattern against the whole set of samples while DDPMine evaluates each pattern on a subset of them. Therefore, a pattern that is selected as discriminative by DDPMine may not be discriminative if all samples are included.

Both DDPMine and EPM use the FP-Tree structure while the MPP approach uses Apriori (Agrawal & Srikant, 1994) in order to generate patterns. Approaches which use the FP-Tree structure are better suited for handling large datasets that contain a large number of frequent items (Heaton, 2017). The major advantage of using FP-Tree for generating patterns compared to Apriori is that FP-Tree iterates over samples only twice while Apriori requires multiple iterations to enumerate patterns. Indeed, the execution time of FP-Tree based algorithms for generating patterns is less than that of Apriori.

In the following sections, we compare the accuracy and time efficiency of EPM to those of MPP and DDPMine.

7.2 Experimental Setup

For evaluation, we implement both DDPMine (Cheng et al., 2008) and MPP (Batal & Hauskrecht, 2010). We use approximated MPP in our experiments. It is a variant of MPP that achieves higher efficiency by using a lossy pruning technique (Batal & Hauskrecht, 2010). DDPMine and EPM both use the same FP-Tree implementation.

In all of the evaluated feature mining methods, Equation 6.1 is used as the frequency condition. The minimum support for the frequency condition, λ_1 in Equation 6.1, is set to 0.1.

For evaluating MPP and EPM, we use the maximum pattern size threshold, Θ_l, for a fair comparison. In this way, both approaches search through the pattern space only up to length Θ_l. Θ_l is set to 3 for both approaches.

We perform 5-times repeated 5-fold cross validation and the results are averaged in order to reduce the possibility of poor estimation due to chance divisions of datasets. In each validation, all experiments are performed on the same split.

A linear SVM, i.e. LIBLINEAR 2.11[1] (Fan et al., 2008), is used as the baseline classifier. The performance is evaluated using micro- and macro-averaged F_1. Assume there are l different class labels and tp_i, tn_i, fp_i and fn_i are the number of true positives, true negatives, false positives, and false negatives for the class label i, respectively. The macro-averaged F_1

[1] Available at `https://www.csie.ntu.edu.tw/~cjlin/liblinear/`

is computed as:

$$macro - F = \frac{1}{l}\sum_{i=1}^{l} F_1(tp_i, fp_i, tn_i, fn_i)$$

The micro-averaged F_1 is computed as:

$$micro - F = F_1(\sum_{i=1}^{l} tp_i, \sum_{i=1}^{l} fp_i, \sum_{i=1}^{l} tn_i, \sum_{i=1}^{l} fn_i)$$

7.3 Datasets

We evaluate EPM on several real-world datasets from the UCI machine learning repository[2] (Lichman, 2013) as well as the *fars* dataset from the KEEL dataset repository[3] (Alcalá-Fdez et al., 2011). In order to demonstrate the scalability of EPM to large datasets, we choose datasets with a large number of features or samples.

Table 7.1 presents characteristics of the selected datasets. The second column shows the number of (real/integer/nominal) features. The third column shows the number of frequent items if $\lambda_1 = 0.1$, i.e. a pattern should occur in at least ten percent of the samples of one of the classes in order to be considered frequent. The number of frequent items is an indicator of the search space complexity. The fourth and the fifth columns show the number of samples and the number of classes in each dataset. Datasets with more than two classes are binarized using a one-vs-all scheme, i.e. by considering a minority class as the first class and all other classes as the second class. We do not use any binning method for converting real or integer features to nominal features in the experiments of this section.

7.4 Evaluation of Time Efficiency

Figure 7.1 shows the running time (in seconds) of EPM on the test datasets with different parameters. EPM(0.1,3) shows the running time when $\Theta_l = 3$ and $\lambda_1 = 0.1$. EPM(0.1,4) uses the same λ_1, but Θ_l is set to 4. The $\lambda_1 = 0.01$ and $\Theta = 3$ parameters are used for EPM(0.01,3). All reported times for *EPM* include the post-processing time.

As can be seen in Figure 7.1, increasing Θ_l, or decreasing λ_1 does not notably affect the running time of EPM on the datasets with smaller search spaces. However, on the datasets with a larger number of frequent items, decreasing λ_1 affects the processing time more than increasing Θ_l.

Figure 7.2 shows the running time of *EPM(0.1,3)* in comparison to those of *DDPMine* and *MPP*. λ_1 is set to 0.1 for all approaches. Because of the large variability of the running times

[2]https://archive.ics.uci.edu/ml/datasets.html

[3]http://sci2s.ugr.es/keel/datasets.php

Dataset	#Features	#FI	#Samples	#Classes
car	(0/0/6)	21	1728	4
cmc	(0/2/7)	24	1473	3
ticTacToe	(0/0/9)	27	958	2
flare	(0/0/11)	27	1066	6
nursery	(0/0/8)	27	12690	5
crx	(3/3/9)	28	653	2
sick	(6/1/22)	36	2800	2
kr-v-k	(0/0/16)	40	28056	17
german	(0/7/13)	51	1000	2
fars	(5/0/24)	67	100968	8
connect-4	(0/0/42)	75	67557	3
census	(1/12/28)	76	299284	3
poker	(0/10/0)	85	1025010	10

Table 7.1: Data characteristics.

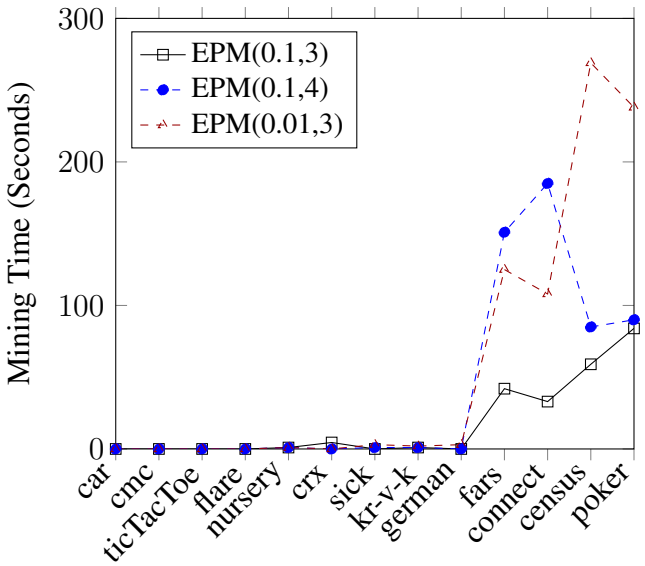


Figure 7.1: Experimental analyses of the EPM mining time.

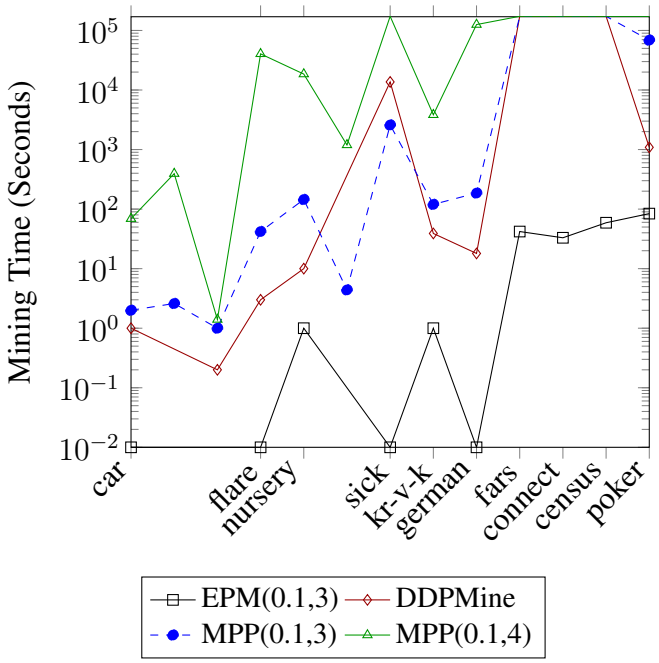


Figure 7.2: Mining time of the EPM, DDPMine and MPP algorithms.

of EPM and other approaches (especially MPP), the running times in Figure 7.2 are reported on a logarithmic scale. *MPP(0.1,3)* and *MPP(0.1,4)* show the results of the approximated MPP approach when $\Theta_l = 3$ and $\Theta_l = 4$, respectively. When the running time exceeds more than two days, the experiments are not included in the plot.

As can be seen in Figure 7.2, EPM is notably faster in comparison to the other two approaches. For the datasets in the lower part of Table 7.1, EPM is the only feature miner that mines the feature set for all the datasets in less than a day. DDPMine and MPP were terminated after two days for the fars, connect-4, census and poker datasets. These datasets are way smaller than the coreference data that includes 8-16 millions of samples and more than 200 frequent items

7.5 Evaluation of Discriminative Power

As mentioned before, we do not use the mined patterns as new features. Instead, we break down the mined patterns into their contained items for selecting feature-values, i.e. using contained feature-values as features instead of using the patterns themselves as features.

However, in order to evaluate the discriminative power of mined patterns, for the experiments of this section, we consider each pattern as a new feature. We then evaluate the effectiveness of these features for improving the performance of the baseline classifier. This is the

	# Patterns			Micro-F				Macro-F			
Dataset	DDP	MPP	EPM	Orig	DDP	MPP	EPM	Orig	DDP	MPP	EPM
car	3	95	65	96.2	96.1	99.9	99.4	49.0	49.0	99.8	96.1
cmc	4	99	23	77.5	77.4	76.2	77.3	57.3	57.1	57.7	59.4
ticTacToe	11	270	92	98.3	98.3	100	100	98.1	98.1	100	100
flare	2	430	86	92.3	93.3	92.7	92.6	49.6	48.3	66.3	61.6
nursery	4	258	198	97.5	98.2	99.9	99.8	49.4	79.4	99.8	98.8
crx	3	145	55	83.7	85.7	86.3	86.2	83.9	85.9	86.4	86.3
sick	5	627	89	94.6	94.7	96.1	95.8	62.6	64.8	81.0	75.6
kr-v-k	7	71	63	99.1	99.1	99.6	99.6	49.8	49.8	87.8	88.4
german	8	548	97	70.7	70.9	73.1	72.7	49.6	55.2	65.3	64.2
fars	-	-	2071	99.2	-	-	99.5	70.5	-	-	85.3
connect-4	-	-	907	90.5	-	-	90.5	47.5	-	-	56.6
census	-	-	5618	93.8	-	-	93.8	48.4	-	-	51.6
poker	-	-	14216	23.1	-	-	49.6	22.4	-	-	44.5

Table 7.2: Evaluating the quality of DDPMine, MPP and EPM patterns on standard datasets.

common practice for evaluating discriminative pattern mining approaches. All patterns are considered as binary features, i.e. the feature is true for samples that include the corresponding pattern. For instance, assume a dataset includes five features, $f_1 \dots f_5$, and pattern p is $\{f_1 = v_1, f_4 = v_6\}$. The value of the corresponding feature of p is true for all samples of the dataset in which the values of f_1 and f_4 are v_1 and v_6, respectively.

The effect of incorporating patterns of the DDPMine, MPP and EPM algorithms in the overall accuracy of the baseline classifier is presented in Table 7.2. The *Orig* columns in Table 7.2 show the results of the baseline classifier, i.e. linear SVM, on the test datasets with original features. The *DDP*, *MPP*, and *EPM* columns show the results of the linear SVM on the datasets for which the feature set is extended by the new binary features mined by DDPMine, MPP, and EPM, respectively.

The first three columns of Table 7.2 show the number of patterns mined by DDPMine, MPP, and EPM.

The results of the 5-repeated 5-fold cross validation evaluation are reported if they take up to ten hours for each single validation, otherwise, they are specified by $-$.

Based on the results of Table 7.2, the following points are worth noting:

- Among the evaluated feature mining approaches, EPM efficiently scales to large datasets.
- The incorporation of patterns generated by both MPP and EPM considerably improves

the performance of the base classifier.

- When all methods are applicable, MPP results in the highest improvements on five out of nine datasets, i.e. car. flare, nursery, sick and german. EPM outperforms or has on-par results with MPP on the rest of datasets. However, MPP becomes very slow if either the size or the dimensionality of the dataset increases.
- The number of features mined by EPM is considerably smaller than that of MPP. It is not clear if the better performance of MPP in some of the datasets should be attributed to the better quality of features or the larger number of features.
- DDPMine always mines only a small number of patterns, regardless of the size or the dimensionality of the dataset.

7.6 Summary

In this chapter, we evaluate EPM both in terms of time efficiency and classification accuracy. Enumerating all frequent patterns of data is an NP-complete problem (Yang, 2006). Therefore, time efficiency is an important factor for a pattern mining algorithm. Otherwise, it wouldn't be practical to use the mining algorithm for extracting informative patterns of large datasets that include many frequent items. As the experiments show, among the examined algorithms, which are shown to be efficient, EPM scales best to large datasets. Besides, the incorporation of EPM's informative patterns results in comparable improvements in the overall accuracy of the baseline classifier.

Chapter 8

Evaluation on Coreference Resolution

In order to investigate the effect of traditional features on state-of-the-art coreference resolvers, we use the EPM algorithm for selecting feature-values that are included in informative patterns. As we will see, the feature-values that are selected by EPM significantly improve the performance in both in-domain and out-of-domain evaluations.

We apply EPM on coreferring and non-coreferring mention-pairs of the CoNLL training set. We use a set of shallow linguistic features (Section 8.2) to describe each mention-pair. We show that (1) the EPM algorithm results in interpretable patterns that provide linguistic insights regarding coreference relations, some of which corroborate previous findings based on smaller datasets (Section 8.4), (2) the incorporation of feature-values that are selected by the EPM algorithm significantly improves the performance of the baseline coreference resolver on the official CoNLL test set (Section 8.5), and (3) the selected feature-values notably improve the generalization of the baseline coreference resolver (Section 8.6). The resulting coreference model that is only trained on the CoNLL data achieves on-par performance with the state-of-the-art system on the WikiCoref dataset, which is specially developed for this dataset and uses domain-specific knowledge.

8.1 Baseline Coreference Resolver

deep-coref (Clark & Manning, 2016a) and e2e-coref (Lee et al., 2017) are the two current state-of-the-art coreference resolvers from which e2e-coref has the best performance on the CoNLL test set. deep-coref is a pipelined system, i.e. a mention detection first determines the list of candidate mentions and then a set of features are extracted based on candidate mentions and their corresponding heads. e2e-coref, on the other hand, is an end-to-end system that jointly models mention detection and coreference resolution. As mentioned in Section 2.4.5, e2e-coref considers all possible (start, end) word spans of each sentence as candidate mentions.

Extracting features of Section 8.2 for every candidate mention of e2e-coref is not computationally efficient, e.g. we need to compute head, NER, gender, and number for all candidate spans of words. One possible solution would be to limit e2e-coref candidate mentions to those that are also detected by a mention detection.

The performance of the e2e-coref single model in which the candidate mentions are restricted to those that are also detected by deep-coref's mention detection[1] is presented as *restricted* in Table 8.1. This result shows that the higher scores of e2e-coref on the CoNLL test set are due to resolving mentions that are not detected by the mention detection. In other words, on the same set of candidate mentions, the single model of e2e-coref does not outperform deep-coref.

	MUC	B^3	CEAF_e	CoNLL	LEA		
	F_1	F_1	F_1	Avg. F_1	R	P	F_1
	CoNLL development set						
deep-coref	74.31	64.23	59.73	66.09	55.42	66.52	60.47
e2e-coref	75.33	65.72	61.04	67.36	59.33	64.77	61.93
restricted	74.29	64.16	58.99	65.81	55.42	66.51	60.46

Table 8.1: Performance of e2e-coref when it is restricted to resolving mentions that are also detected by deep-coref's mention detection.

As a result, we choose deep-coref as the baseline coreference resolver in our experiments. As mentioned in Section 2.4.4, deep-coref includes various coreference models including the mention-pair, mention-ranking, and entity-based models. The mention-ranking model of deep-coref has three variations. The mention-ranking model uses the slack-rescaled max-margin training objective of Wiseman et al. (2015), we refer to this model as "ranking" in the experiments of this section. We refer to the variation of deep-coref's mention-ranking model in which the best hyper-parameters of the loss function are set in a reinforcement learning framework (Sutton & Barto, 1998) as "reinforce". Finally, deep-coref contains a simpler variation of the mention-ranking model that uses a probabilistic instead of a max-margin objective function. This model is referred to as "top-pairs". deep-coref models are explained in more details in Section 2.4.4. We use the top-pairs and ranking models as baselines in the experiments of this chapter.

In the next section, we describe the set of features on which we apply the EPM algorithm

[1] deep-coref preprocessing module includes a liberal mention detection flag. If this flag is set, the number of detected mentions increases considerably while the performance of pairwise, top-pairs and ranking models decreases significantly. We only use this flag for the "restricted" experiment in order to restrict e2e-coref candidate mentions as less as possible.

for mining informative patterns.

8.2 Base Features

The set of features that we use in our experiments includes the following syntactic and shallow semantic features for both anaphor and antecedent:

- mention type, i.e. proper name, nominal, pronominal and other
- fine mention type that determines whether the mention is a proper name, definite or indefinite nominal, or one of the he, she, it, they, I, you, or we pronouns
- gender, number, animacy and named entity type of each mention
- dependency relation of the head word to its governor
- POS tags of the first, last, head, two preceding and two following words of each mention

We also include the following mention-pair features, i.e. mainly string match features, in our feature set:

- head match
- string of one mention is contained in the string of the other one
- head of one mention is contained in the string of the other one
- compatible gender, compatible number, and compatible animacy, i.e. corresponding values matches or one of the values is unknown

Above features are all extracted using the Stanford CoreNLP preprocessing modules[2].

We only employ features that are not already incorporated in deep-coref [3]. The feature set of deep-coref are:

- speaker features
- exact match
- refined head match, i.e. two mentions have the same entity type and the head of one mention is included in the tokens of the other one

[2]Available at `https://stanfordnlp.github.io/CoreNLP/coref.html`

[3]The only exception is for the experiments of Section 8.4, in which we also include the distance of two mentions as one of the features.

- relaxed string match, i.e. if there is a word with one of the "NN", "NNS", "NNP", "NNPS" POS tags in one mention it should also exist in the other mention
- binned values of mention lengths and the distance of two mentions

deep-coref also incorporates the embeddings of the head, first, last, two preceding, and two following words of each mention as well as the averaged word embeddings of the five preceding, five following, all mention words, all sentence words, and all document words.

	MUC	B^3	CEAF$_e$	CoNLL	LEA		
	F$_1$	F$_1$	F$_1$	Avg. F$_1$	R	P	F$_1$
	top-pairs model						
base	73.95	63.98	59.52	65.82	54.25	67.29	60.07
-avg. embedding	73.84	63.62	59.42	65.63	54.45	66.11	59.71
	ranking model						
base	74.31	64.23	59.73	66.09	55.42	66.52	60.47
-avg. embedding	74.05	63.82	59.78	65.88	55.26	65.65	60.01

Table 8.2: Effect of excluding averaged word embeddings from deep-coref on the CoNLL-2012 development set for both top-pairs and ranking models.

As we show in Section 5.2, coreference resolvers that use more lexical features are more prone to overfitting. In order to reduce the effect of lexical memorization to some extent, we do not incorporate averaged word embeddings in our experiments. In the same line of reasoning, we also do not consider any lexical features in EPM in order to mine more generalizable features. The performance of deep-coref with, i.e. "base", and without averaged word embeddings, i.e. "-avg. embeddings" on the CoNLL development set is presented in Table 8.2. By excluding averaged word embeddings, the performance drops slightly in in-domain evaluations.

8.3 EPM Experimental Setup

To extract informative patterns for coreference resolution, we use the set of mention-pairs from the CoNLL-2012 training set. deep-coref's mention detection is employed for extracting candidate mentions. For creating mention-pairs, each mention is paired with all of its previous mentions, which is the most common approach for enumerating mention-pairs in coreference resolution. Each pair is then described by the features of Section 8.2.

We consider a pattern as frequent if it occurs in coreference relations of at least λ_2 different coreferring anaphors. This way we exclude patterns that only occur in the coreferent relations of a specific anaphor that has more than λ_2 antecedents. λ_2 is set to 20 in our experiments.

Apart from *frequency*, *discriminative power* and *information novelty* measures, we also consider the coreference probability of patterns in the post-processing step. The coreference probability is the number of all coreferring mention-pairs that contain pattern p to the number all mention-pairs that contain p:

$$\frac{|\{X_i|p \in X_i \wedge c(X_i) = coreferent\}|}{|\{X_i|p \in X_i\}|} \tag{8.1}$$

The coreference probability of a pattern should be higher than a threshold in order to be selected as an informative pattern. We set this threshold to 60% so we mine patterns that are informative for coreferring mentions and not for non-coreferring ones[4].

The p-value thresholds for both G^2 and binomial tests are set to 0.01. For the experiments of this chapter, we set the pattern length threshold (Θ_l) to five[5].

Following previous studies that show different features are of different importance for various types of mentions (Denis & Baldridge, 2008; Lassalle & Denis, 2013; Moosavi & Strube, 2017b), we mine a separate set of features for each type of anaphor, i.e. proper, nominal and pronominal. To do so, we separate mention-pairs based on the type of anaphor and mine a separate set of patterns for each type of anaphor.

8.4 Linguistic Insights from EPM Patterns

We can use EPM patterns to gain linguistic insights about coreference relations since they are frequent patterns that are discriminative for coreference relations, contain novel information and are more probable to occur in coreferring mention-pairs than non-coreferring ones. We can use this insight to design new features. Besides, EPM can be used as a data-driven approach for confirming or refining earlier linguistic findings, which were based on smaller datasets, on more recent and larger datasets.

Our findings based on the EPM patterns in the CoNLL training data are as follows:

1. The feature values "compatible gender=true", "compatible number=true", and "compatible animacy=true" occur very frequently together and in patterns in which the distance of two mentions is very small. Based on these feature-values, we design the feature "nearest compatible antecedent", i.e. the nearest antecedent that has compatible gender, number, and animacy as those of the anaphor. This feature can be considered as a

[4]Setting the coreference probability to higher values results in only long and specific patterns.

[5]The use of larger Θ_l values increases the number of informative patterns considerably and results in many specific patterns.

modified version of Uryupina (2007)'s *closest-closest-agree* feature, which indicates an antecedent that is the closest to the anaphor with the matching number and person characteristics. Based on Uryupina (2007)'s definition, two person characteristics match if they are equal or at least one of them belong to a quoted mention.

2. The three above feature-values also occur frequently in patterns in which the dependency relation of the antecedent is either subject or direct object. Therefore, we create the features "nearest compatible subject" and "nearest compatible object" features based on these patterns. It is worth noting that these features and the "nearest compatible antecedent" feature occur mostly in informative patterns of pronouns. These features are similar to Uryupina (2007) *closest-subject-agreement* feature, i.e. the closest antecedent that is both a subject and has compatible number and person characteristics. However, she only considers the subject relation while based on EPM patterns, the incorporation of the object dependency relation is also beneficial. Uryupina (2007) creates such features based on the concept of combining *proximity*, *salience* and *agreement* properties.

3. The feature-values "compatible modifiers=true" and "head match=true" occur frequently together for patterns in which the distance of two mentions is small. Based on these patterns, we create the feature "nearest head match with compatible modifiers", i.e. the nearest antecedent that has the same head and compatible pre-modifiers with those of the anaphor.

4. The set of EPM patterns for pronouns contains several patterns in which the POS tag of the preceding word is a quotation mark. This pattern confirms the efficacy of the "mention is quoted" features that is used in earlier coreference resolvers, e.g. Ng & Cardie (2002), Uryupina (2007).

5. There are several informative patterns that contain the feature-value "antecedent preceding POS=START". This feature-value specifies antecedents that appear at the beginning of a sentence. The occurrence of this feature-value in informative patterns indicates the efficacy of the feature "first mention of a sentence" that is used by Uryupina (2007) to model salience.

6. Based on the informative patterns that are mined for different types of anaphors, we see that various features are of different importance for resolving different types of anaphors. This observation confirms previous findings that different features, or even models, are required for resolving different types of mentions Denis & Baldridge (2008), Lassalle & Denis (2013), Moosavi & Strube (2017b).

   For instance, if we add the "nearest compatible antecedent", "nearest compatible subject", "nearest compatible object" and "nearest head match with compatible modifiers"

features to the feature set of Section 8.2 and repeat the mining process, about 98% of the informative patterns for pronouns would include at least of one of the first three features, i.e. "nearest compatible antecedent", "nearest compatible subject" or "nearest compatible object". However, this ratio is only around 0.01% and 0.07% for patterns of proper names and common nouns, respectively. This indicates that distance is a distinguishing feature for resolving pronouns while it is of less importance for resolving proper names or nominals. As another example, more than 98% of the informative patterns for proper names contain one of the string match features, which in turn shows the importance of these features for resolving proper names.

7. The number of informative patterns that are mined for nominals are considerably lower compared to proper names and pronouns, i.e. less than one percent. This indicates that the features of Section 8.2 do not contain enough information for resolving nominals. Therefore, other sources of information are required to improve the resolution of nominals. For instance, the use of word embeddings is shown to be useful for improving the performance for nominals (Clark & Manning, 2016a; Moosavi & Strube, 2017b).

8.5 Are Linguistic Features Still Useful?

In this section, we show that the incorporation of linguistic features is indeed beneficial in a state-of-the-art coreference resolver.

In the first experiment (Section 8.5.1), we employ the feature set of Section 8.2 as it is. As the results show, the inclusion of these features with all their corresponding values does not improve the baseline coreference resolver, i.e. deep-coref.

In the second experiment (Section 8.5.2), we only employ the feature-values that are selected by EPM. The inclusion of EPM feature-values results in a significant improvement. This result shows that in order to benefit from linguistic features we need to select them properly, even in a deep neural network.

8.5.1 Incorporating All Features

In our first set of experiments, we add all features of Section 8.2, which includes various string match, syntactic and shallow semantic features, to the feature set of deep-coref. From the above features, string match and agreement features are pairwise features with binary values. The gender, number, animacy and mention type features, which have very few values, i.e. less than five, are also converted to binary features, i.e. each feature=value is considered as a binary feature. The rest of the features, i.e. named entity and POS tags, and dependency relations, are represented as learned embeddings.

The performances of the two mention-ranking models of deep-coref, i.e. top-pairs and ranking, with, i.e. "+all", and without, i.e. "base", these features are presented in Table 8.3 .

	MUC			B^3			CEAF$_e$			CoNLL	LEA		
	R	P	F$_1$	R	P	F$_1$	R	P	F$_1$		R	P	F$_1$
CoNLL development set													
top-pairs													
base	69.05	79.61	73.95	58.13	71.13	63.98	55.25	64.50	59.52	65.82	54.25	67.29	60.07
+all	70.17	78.34	74.03	59.17	69.82	64.06	56.73	63.84	60.08	66.06	55.40	65.99	60.23
ranking													
base	69.88	79.34	74.31	59.18	70.21	64.23	55.34	64.89	59.73	66.09	55.42	66.52	60.47
+all	70.99	77.23	73.98	60.49	68.07	64.06	57.30	63.28	60.14	66.06	56.71	64.19	60.22

Table 8.3: Impact of linguistic features on the overall performance. The "base" rows show the performance of the baseline models. The "+all" rows show the performance of the baseline in which all features are incorporated.

As we can see from the results, incorporating the linguistic features helps bridge the gap between the performance of the top-pairs and ranking models. However, it does not improve over the baseline ranking model.

8.5.2 Incorporating Informative Feature-Values

In this section, we show that linguistic features improve the performance if we only incorporate informative feature-values[6] instead of all feature-values. We use EPM in order to select informative feature-values.

In the experiments of this section and Section 8.6, we extend the features of Section 8.2 with the "nearest compatible antecedent", "nearest compatible subject", "nearest compatible object features" and "nearest head match with compatible modifiers" features (Section 8.4), which are designed based on the EPM patterns. We then repeat the mining process on the extended feature set.

EPM results in 194 informative feature-values[7] including 13 pairwise feature-values, e.g. "nearest compatible antecedent=true", 53 and 59 POS tags, 13 and 12 dependency relations, 11 and 15 mention types (mention types or fine mention types), and finally six and eight named entity tags, for anaphors and antecedents, respectively. None of the values of the gender, number and animacy features are among the selected feature-values while the corresponding compatibility feature-values, i.e. "compatible number=true", "compatible gender=true" and "compatible animacy=true", are informative.

[6]We refer to feature-values that form informative patterns as informative feature-values.

[7]It takes from two to six hours for mining informative patterns for different types of anaphors.

We then add these feature-values as binary features in deep-coref. deep-coref itself has six pairwise features, i.e. three speaker match, exact string match, refined head match and relaxed string match. It has a hidden layer of size 1000 on top of the pairwise features. The output of the hidden layer is then combined with the output of the hidden layer of word embeddings. Apart from the number of pairwise features, we do not change any other parameter in deep-coref. The deep-coref model in which the EPM feature-values are incorporated is referred to as "+EPM" in our experiments.

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1		R	P	F_1
CoNLL development set													
top-pairs													
base	69.05	79.61	73.95	58.13	71.13	63.98	55.25	64.50	59.52	65.82	54.25	67.29	60.07
+EPM	70.92	79.41	74.92	60.08	70.86	65.03	57.05	65.26	60.88	66.95	56.41	67.22	61.34
ranking													
base	69.88	79.34	74.31	59.18	70.21	64.23	55.34	64.89	59.73	66.09	55.42	66.52	60.47
+EPM	70.53	79.87	74.91	60.03	71.03	65.07	56.32	65.99	60.77	66.92	56.45	67.46	61.46

Table 8.4: Impact of informative feature-values. The "base" rows show the baseline results while the "+EPM" rows presents the performance when EPM feature-values are added. The F_1 gains of "+EPM" are statistically significant for all metrics.

The impact of informative feature-values, which are selected based on the EPM algorithm, is shown in Table 8.4. In this table, the "base" rows show the baseline results while the "+EPM" rows show the results of the baseline in which the EPM feature-values are also incorporated.

As we can see, informative feature-values yield considerable improvements over the baseline. The F_1 gains of "+EPM" in both top-pairs and ranking models are statistically significant for all metrics. In all experiments of this chapter, the statistical significance is measured by the approximate randomization test (Noreen, 1989). We consider an improvement statistically significant if $p < 0.05$. It is worth noting that, as mentioned in Section 8.2, averaged word embeddings are not included in any of the "+EPM" experiments.

Similar to the results of Table 8.3, when the feature set is enhanced with the informative feature-values, the top-pairs and ranking models have on-par performance while the top-pairs model uses a simpler objective function, and it is used for pretraining the ranking model. For the rest of the experiments of this chapter, we use the top-pairs model of deep-coref as the baseline, i.e. "+EPM" refers to the top-pairs model with EPM feature-values.

The performance of the "+EPM" model compared to recent state-of-the-art coreference resolvers on the CoNLL test set is presented in Table 8.5. The F_1 gains of "+EPM" compared to all "top-pairs", "ranking", and "reinforce" models are statistically significant. The only

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1		R	P	F_1
CoNLL test set													
deep-coref													
top-pairs	69.41	79.90	74.29	57.01	70.80	63.16	54.43	63.74	58.72	65.39	53.31	67.09	59.41
ranking	70.43	79.57	74.72	58.08	69.26	63.18	54.43	64.17	58.90	65.60	54.55	65.68	59.60
reinforce	69.84	79.79	74.48	57.41	70.96	63.47	55.63	63.83	59.45	65.80	53.78	67.23	59.76
+EPM	71.16	79.35	75.03	59.28	69.70	64.07	56.52	64.02	60.04	66.38	55.63	66.11	60.42
e2e-coref													
single	74.02	77.82	75.88	62.58	67.45	64.92	59.16	62.96	61.00	67.27	58.90	63.79	61.25
ensemble	73.73	80.95	77.17	61.83	72.10	66.57	60.11	65.62	62.74	68.83	58.48	68.81	63.23

Table 8.5: Performance comparisons on the CoNLL-2012 test set. The F_1 gains of "+EPM" compared to "top-pairs" and "ranking" are statistically significant based on all metrics. The F_1 gains of "+EPM" compared to "reinforce" are statistically significant based on MUC, B^3 and LEA. The F_1 gains of "+EPM" compared to "single" are only significant based on MUC and B^3. The F_1 gains of "ensemble" compared to all other systems are statistically significant according to all metrics.

exception is the difference between “+EPM” and “reinforce” based on the *CEAF*$_e$ metric, which is not significant.

The “single” and “ensemble” rows represent the results of the e2e-coref coreference resolver in which a single model and an ensemble of five different models are used, respectively.

The minimum span evaluation of the aforementioned systems on the CoNLL test set is shown in Table 8.6.

8.5.3 Impact of Individual Features

In this section, we investigate the effect of each group of feature-values, i.e. pairwise features, mention types, dependency relations, named entity tags and POS tags, on the overall performance. Table 8.7 shows the results of the feature ablation studies. “+EPM” represents the results of the top-pairs model in which all informative feature-values are incorporated. The performance of “+EPM” from which each of the above feature groups is removed, i.e. one feature group at a time, is represented as “-pairwise”, “-types”, “-dependencies”, “-NER”, and “-POS”, respectively.

The POS and named entity tag feature-values have the least effect on the performance. On the other hand, the pairwise features have the most significant effect. “+pairwise” shows the performance of the top-pairs model in which only pairwise features are incorporated. The

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1		R	P	F_1
CoNLL test set													
deep-coref													
top-pairs	70.92	81.63	75.90	58.61	72.79	64.93	55.80	65.35	60.20	67.01	55.07	69.36	61.39
ranking	71.65	81.80	76.39	59.27	72.17	65.09	55.88	65.58	60.34	67.27	55.84	68.83	61.66
reinforce	71.42	81.60	76.17	59.06	73.04	65.31	57.10	65.51	61.02	67.50	55.62	69.60	61.83
+EPM	71.89	81.93	76.58	59.82	72.61	65.60	57.48	66.05	61.46	67.88	56.37	69.29	62.17
e2e-coref													
single	74.64	79.95	77.20	62.92	70.31	66.41	60.09	64.54	62.23	68.62	59.43	66.92	62.95
ensemble	74.45	82.74	78.37	62.76	74.38	68.08	61.64	66.84	64.13	70.20	59.47	71.33	64.86

Table 8.6: The minimum span evaluation of the systems of Table 8.5.

results of "-pairwise" compared to "+pairwise" show that pairwise feature-values have a significant impact, but only in combination with other feature-values.

	MUC	B^3	$CEAF_e$	CoNLL	LEA		
	F_1	F_1	F_1		R	P	F_1
CoNLL development set							
+EPM	74.92	65.03	60.88	66.95	56.41	67.22	61.34
-pairwise	74.37	64.55	60.46	66.46	55.71	66.70	60.71
-types	74.71	64.87	61.00	66.86	55.99	67.17	61.07
-dependencies	74.57	64.79	60.65	66.67	56.97	65.65	61.01
-NER	74.61	65.05	60.93	66.86	56.12	67.46	61.27
-POS	74.74	65.04	60.88	66.89	57.13	66.13	61.30
+pairwise	74.25	64.33	60.02	66.20	54.85	67.62	60.57

Table 8.7: Feature ablation experiments. "-pairwise", "-types", "-dependency", "-NER", and "-POS" represent the performance of the "+EPM" model in which the corresponding feature group, i.e. pairwise, mention type, dependency relations, named entity tags, or POS tags feature-values, is removed from EPM feature-values. At each experiment, only one set of features is removed. "+pairwise" represents the performance of the top-pairs model in which only the informative pairwise feature-values are incorporated.

Some feature-values capture competing information. For instance, POS tags implicitly capture whether a mention is a subject or an object, i.e. if the following or previous word of a mention is tagged as a verb, while this information is explicitly captured by dependency relations. Hence, while POS tags do not seem to have a big impact on the overall performance of the development set, we keep them among the set of feature-values. The inclusion of

competing features can result in more robust models on test data in which the values of some of these features are noisy or missing (Sutton et al., 2006).

8.5.4 Comparison to Uryupina & Moschitti's (2015) Patterns

In this section, we compare EPM with the pattern mining approach that is used by Uryupina & Moschitti (2015), i.e. Jaccard index mining. We run the Jaccard index mining algorithm on mention-pairs of the CoNLL-2012 training set, in which each pair is described with the same set of features as that of EPM in experiments of Section 8.5.2.

The impact of feature-values that are selected by EPM compared to the mining algorithm of Uryupina & Moschitti (2015) is shown in Table 8.8. For the experiments of this table, we set the minimum frequency, maximum pattern length and $score^{+}$ threshold parameters to 20, 5 and 0.6 for JIM. This results in 356 selected feature-values. The selected feature values include nine pairwise features, 260 POS tags, 38 dependency relations, 32 mention type information, and 18 named entity tags.

The "+JIM" row shows the results of deep-coref top-pairs model in which these 356 feature-values are incorporated. The results of Table 8.8 shows that EPM feature-values result in better performance than those of JIM. The difference between the performance of "+JIM" and "top-pairs" is statistically significant only based on *MUC*, and $CEAF_e$ metrics. The performance differences between "+EPM" and "+JIM" are statistically significant based on all metrics.

	MUC	B^3	$CEAF_e$	CoNLL	LEA		
	F_1	F_1	F_1		R	P	F_1
CoNLL test set							
top-pairs	74.29	63.16	58.72	65.39	53.31	67.09	59.41
+JIM	74.81	63.52	59.45	65.93	53.46	67.97	59.85
+EPM	75.03	64.07	60.04	66.38	55.63	66.11	60.42

Table 8.8: Impact of feature-values that are mined by EPM compared to those of JIM.

As mentioned in Section 8.4, the use of discriminative power and coreference probability in EPM results in linguistically interpretable patterns. However, the mining algorithm of Uryupina & Moschitti (2015) (Section 6.2) relies on a redundancy measure for selecting patterns and therefore, resulting patterns are not necessarily informative. For instance, in our informative patterns the "head match =true" features appears in patterns of at least length three for proper names and length four for nominals. It also does not occur in any patterns for pronouns. This shows that head match alone is not a strong feature for resolving any types of anaphors. On the other hand, the {"head match=true", "nearest compatible object=false"},

{“antecedent type=nominal”, “head match=true”}, and {”nearest compatible object=false“, ”nearest compatible subject=false“, ”head match=true“} patterns are among JIM patterns with the highest $score^{+}$ value.

8.6 Do Linguistic Features Improve Generalization?

In this section, we investigate the effect of informative feature-values on the generalization of coreference resolvers across domains.

In order to do so, we perform two sets of out-of-domain evaluations: (1) using the CoNLL-2012 training and development sets for training the model and tuning the parameters, and then testing the model on the WikiCoref dataset, and (2) using each genre of the CoNLL test data as the test set while removing the corresponding genre from both training and development sets, e.g. using the *pt* genre of the test data as the test set while excluding *pt* from both training and development sets.

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1		R	P	F_1
WikiCoref													
top-pairs	56.31	71.74	63.09	39.78	61.85	48.42	40.80	52.85	46.05	52.52	35.87	57.58	44.21
ranking	57.72	69.57	63.10	41.42	58.30	48.43	42.20	53.50	47.18	52.90	37.57	54.27	44.40
reinforce	62.12	58.98	60.51	46.98	45.79	46.38	44.28	46.35	45.29	50.73	42.28	41.70	41.98
+EPM	58.23	74.05	65.20	43.33	63.90	51.64	43.44	56.33	49.05	55.30	39.70	59.81	47.72
e2e-single	60.14	64.46	62.22	45.20	51.75	48.25	38.18	43.50	40.67	50.38	40.70	47.56	43.86
e2e-ensemble	59.58	71.60	65.04	44.64	60.91	51.52	40.38	49.17	44.35	53.63	40.73	56.97	47.50
Ghaddar	66.06	62.93	64.46	57.73	48.58	52.76	46.76	49.54	48.11	55.11	-	-	-

Table 8.9: Comparison of the results on the WikiCoref dataset. Except for “Ghaddar”, all models use the CoNLL-2012 training and development sets during training and do not incorporate any information from WikiCoref.

The results of the first set of experiments are shown in Table 8.9. Similar to the experiments of Section 8.5, “top-pairs”, “ranking”, “reinforce” and “+EPM” are various models of deep-coref. “e2e-single” and “e2e-ensemble” represent the results of the single and ensemble models of e2e-coref, respectively. The best performance on WikiCoref has been achieved by Ghaddar & Langlais (2016a), i.e. represented as “Ghaddar” in Table 8.9, who introduced WikiCoref and design a domain-specific coreference resolver that makes use of the Wikipedia markup of a document and links to external knowledge bases such as Freebase.

As the results show, while “+EPM” does not use the WikiCoref data during training, and unlike “Ghaddar”, it does not employ any domain-specific features, it achieves on-par performance with that of “Ghaddar” on this dataset. This indeed shows the effectiveness of informa-

	MUC			B^3			$CEAF_e$			CoNLL	LEA		
	R	P	F_1	R	P	F_1	R	P	F_1		R	P	F_1
WikiCoref													
top-pairs	58.20	74.12	65.20	41.54	64.68	50.59	42.32	54.82	47.76	54.52	37.69	60.71	46.51
ranking	59.94	72.30	65.54	43.62	61.41	51.01	43.88	55.63	49.06	55.20	39.77	57.65	47.07
reinforce	65.98	63.03	64.47	50.97	49.79	50.37	47.03	49.22	48.10	54.32	46.31	45.95	46.13
+EPM	59.99	76.32	67.18	44.98	66.54	53.67	44.77	58.05	50.56	57.14	41.38	62.78	49.88
e2e-single	61.55	66.46	63.91	46.60	53.87	49.97	39.74	45.26	42.32	52.07	42.12	49.90	45.68
e2e-ensemble	60.77	73.17	66.40	45.82	62.73	52.96	41.72	50.80	45.81	55.06	41.98	59.10	49.09

Table 8.10: Comparison of the results on the WikiCoref dataset based on minimum spans.

tive feature-values in improving the generalization of the baseline coreference resolver, and therefore making the resolver robust across domains.

Table 8.11 shows the in-domain vs. out-of-domain evaluations for different genres of the CoNLL-2012 test set. As before, "ranking" is the ranking model of deep-coref and "+EPM" is deep-coref's top-pairs model that is enhanced with the EPM feature-values. For "in-domain" evaluations, coreference resolvers are trained on all genres of the training and development sets and the results are reported for each test genre separately. In "out-of-domain" evaluations, the test genre is excluded from both training and development sets. Table 8.12 reports the results of the same set of experiments using minimum spans instead of maximum spans.

Based on the results of Table 8.11 and 8.12, we can see that:

1. For all genres "+EPM" outperforms "ranking" in both in-domain and out-of-domain evaluations. The only exception is the "mz" genre, in which the incorporation of the EPM feature-values does not seem to have a considerable impact.

2. Based on maximum span evaluations, the use of the CoNLL score and LEA F_1 results in different rankings in "mz", "wb" and "tc" genres, e.g. for the "tc" genre, "e2e-coref" outperforms "+EPM" more than one percent based on the CoNLL score while "+EPM" has a slightly better LEA F_1 score than "e2e-coref". However, by using minimum span evaluations, both CoNLL score and LEA F_1 result in the same rankings.

Overall, the results of Table 8.9-Table 8.12 show the incorporation informative feature-values considerably, i.e. two to three percent, improve generalization across domains. These improvements are despite the fact that all the feature-values are extracted from error-prone preprocessing modules.

	in-domain				out-of-domain			
	CoNLL	LEA			CoNLL	LEA		
	Avg. F_1	R	P	F_1	Avg. F_1	R	P	F_1
pt (Bible)								
ranking	75.61	68.48	73.70	71.00	66.06	52.44	63.84	57.58
+EPM	76.08	68.14	74.40	71.13	**68.14**	52.14	72.74	**60.74**
e2e-coref	**77.80**	73.59	73.87	**73.73**	65.22	51.87	66.44	58.26
bn (broadcast news)								
ranking	65.27	51.71	66.31	58.10	61.12	46.89	61.40	53.17
+EPM	65.65	51.96	66.97	58.52	62.42	46.45	66.41	54.66
e2e-coref	**67.84**	56.11	67.02	**61.08**	**62.91**	49.00	64.37	**55.64**
nw (newswire)								
ranking	61.54	49.92	62.06	55.34	56.90	44.35	55.96	49.48
+EPM	**62.96**	50.36	65.09	**56.78**	**58.79**	44.42	60.86	**51.36**
e2e-coref	61.88	53.00	58.44	55.59	55.73	44.49	52.82	48.30
tc (telephone conversations)								
ranking	68.49	62.58	71.95	66.90	61.24	61.29	53.49	57.13
+EPM	69.01	64.08	71.47	**67.57**	62.83	55.15	62.05	**58.40**
e2e-coref	**70.22**	66.72	68.20	67.45	**64.21**	61.11	55.89	58.38
bc (broadcast news)								
ranking	60.38	45.86	63.65	53.31	58.23	46.28	55.39	50.43
+EPM	61.15	48.49	62.25	54.51	59.01	46.12	57.81	51.31
e2e-coref	**62.66**	50.69	62.40	**55.94**	**60.13**	46.89	60.61	**52.87**
mz (magazine)								
ranking	70.36	59.52	71.34	64.89	**67.28**	61.77	60.61	61.19
+EPM	71.81	60.10	73.86	66.27	66.94	61.50	61.03	**61.27**
e2e-coref	**72.99**	63.83	72.75	**68.00**	66.85	59.56	63.07	**61.27**
wb (weblog)								
ranking	61.46	48.04	60.99	53.75	57.17	50.29	47.27	48.74
+EPM	61.97	47.89	61.72	**53.93**	**61.52**	50.58	57.41	**53.78**
e2e-coref	**62.02**	49.54	57.19	53.09	60.69	53.00	52.39	52.69

Table 8.11: Performance comparison for all genres of the CoNLL test set in two different settings: (1) the corresponding genre of the test set is included in the training and development sets (in-genre), and (2) the corresponding genre of the test set is excluded from both training and development sets (out-of-genre). The highest CoNLL and LEA F_1 scores are boldfaced.

	in-domain				out-of-domain			
	CoNLL	LEA			CoNLL	LEA		
	Avg. F_1	R	P	F_1	Avg. F_1	R	P	F_1
pt (Bible)								
ranking	76.68	69.71	75.00	72.26	67.07	53.49	65.23	58.78
+EPM	77.12	69.26	75.74	72.36	**69.06**	53.13	73.97	**61.84**
e2e-coref	**78.56**	74.32	75.05	**74.68**	67.07	53.26	67.39	59.50
bn (broadcast news)								
ranking	67.47	54.13	69.71	60.94	63.24	49.09	64.58	55.78
+EPM	67.66	54.11	70.15	61.09	64.24	48.31	69.37	56.95
e2e-coref	**69.69**	58.13	69.93	**63.48**	**64.53**	51.46	65.23	**57.53**
nw (newswire)								
ranking	63.37	51.85	64.71	57.57	58.56	46.20	58.12	51.48
+EPM	**64.65**	52.18	67.56	**58.88**	**60.36**	46.16	62.99	**53.28**
e2e-coref	63.52	54.77	60.61	57.54	57.82	46.68	55.68	50.78
tc (telephone conversations)								
ranking	68.82	62.93	72.26	67.28	61.55	61.63	53.77	57.43
+EPM	69.34	64.22	71.77	67.78	63.07	55.28	62.35	58.60
e2e-coref	**70.64**	67.00	68.77	**67.87**	**64.83**	61.76	56.59	**59.06**
bc (broadcast news)								
ranking	61.88	47.35	65.90	55.10	59.80	48.00	57.42	52.29
+EPM	62.58	50.15	64.14	56.29	60.53	47.78	60.00	53.20
e2e-coref	**63.84**	51.90	64.00	**57.31**	**61.38**	49.32	60.06	**54.16**
mz (magazine)								
ranking	72.70	62.27	74.75	67.94	**70.08**	65.31	64.08	**64.69**
+EPM	73.79	62.42	76.67	68.82	69.35	64.55	64.18	64.37
e2e-coref	**74.74**	65.81	75.36	**70.26**	67.72	60.46	64.36	62.35
wb (weblog)								
ranking	63.27	49.86	63.59	55.89	59.34	52.62	49.74	51.14
+EPM	**63.89**	49.63	64.41	**56.07**	**63.42**	52.59	60.01	**56.05**
e2e-coref	63.63	51.23	59.52	55.07	62.09	54.70	52.33	53.49

Table 8.12: Corresponding comparisons of Table 8.11 based on minimum span evaluations. The highest CoNLL and LEA F_1 scores are boldfaced. "ranking" and "+EPM" are models of deep-coref. "e2e-coref" represents the results of the single model of e2e-coref.

8.7 Summary

Due to the success of word embeddings and deep neural networks in various NLP tasks, there is a shift from pipelined to end-to-end systems. This shift is mainly based on the assumption that the learned representations in hidden layers capture the required knowledge, and is, therefore, no need for intermediate representations from the pipelined modules, e.g. Ringgaard et al. (2017).

We show that traditional features, which are obtained from error-prone pipelined modules, encode auxiliary information to what is captured by the learned representations, and their incorporation results in significant improvements.

We also show that traditional features considerably improve the generalization of the baseline coreference resolver across domains.

According to our experiments, a key requirement to the success of linguistic features in coreference resolution is to only choose informative feature-values. For instance, features like lexicalized dependency relations or POS tags have a large value set, from which many values are irrelevant for discriminating coreference relations. We cast the problem of finding informative feature-values as a pattern mining approach and introduce an efficient pattern mining approach, called EPM, that scales to large datasets. By using EPM we can only choose informative values of each feature, which in combination with other feature-values, are discriminative for coreference relations. The informativeness of patterns is measured based on the frequency, discriminative power and information novelty measures of Section 6.1.4.

Besides, as we show in Section 8.4, because of the quality of resulting patterns, we can use EPM to gain linguistic insights about coreference relations.

Part IV

Conclusions

Chapter 9

Conclusions

In this thesis, we introduce robustness in coreference resolution. In this regard, we emphasize the importance of robustness in coreference resolution systems and provide an evaluation framework to ensure it. We then show that using a selected subset of linguistic feature-values is a promising solution for making coreference resolvers robust across domains.

In this chapter, we summarize our contributions and discuss two possible future directions.

9.1 Contributions

Our contributions in this thesis are as follows:

1. **Introducing a reliable evaluation metric**

   We analyze existing evaluation metrics and show that apart from the known drawbacks, B^3, CEAF and BLANC metrics suffer from the mention identification effect, in which the metric has a bias towards rewarding system outputs that include more gold mentions, even in completely wrong system entities. The only metric that results in interpretable recall and precision values with regard to the mention identification effect is MUC, which is the least discriminative metric for coreference resolution. We introduce LEA, i.e. the Link-Based Entity-Aware metric, that overcomes the drawbacks of previous metrics. We perform various analyses on the LEA metric and show that it is a reliable metric for coreference resolution.

2. **Introducing a mechanism for making minimum span evaluations available for every English corpus**

   Evaluating coreference resolution algorithms based on minimum spans instead of maximum spans is a way to disentangle coreference evaluations from parsing complexities. However, minimum spans are only annotated for small corpora and are not available

in large corpora including the standard dataset, i.e. CoNLL-2012. We propose an algorithm for automatically extracting minimum spans during evaluations. We show that minimum spans that are extracted by our algorithm are compatible with those that are manually annotated by human experts.

3. **Identifying factors responsible for the poor generalization of state-of-the-art systems**

   Since the introduction the CoNLL-2012 dataset, it has been used as the main dataset for developing coreference resolution algorithms and their evaluations. We show that there is a significant overlap between the training and test sets of the CoNLL dataset, which in turn rewards systems that have a better memorization of the training data. This notable overlap in combination with using lexical features as the main source of information, strongly biases state-of-the-art coreference resolvers towards resolving mentions that also appear in the training data. As a result, they have limited generalization in resolving mentions or mention-pairs that do not exist in the CoNLL training set. This problem indicates that performing out-of-domain evaluations is inevitable in coreference evaluation for ensuring robust improvements.

4. **Introducing an approach to benefit from linguistic features in state-of-the-art coreference resolvers**

   We use a simple set of linguistic features including string match, POS tags, syntactic and shallow semantic features and examine their effect on the overall performance of a state-of-the-art coreference resolver. Our experiments show that the incorporation of these features with all their corresponding values does not improve the performance. We introduce an efficient approach, called EPM, that casts the problem of finding feature-values that, in combination with other feature-values, are discriminative for coreference relations, as a pattern mining approach. We show that in comparison to previous discriminative pattern mining approaches, EPM is very efficient, and is therefore applicable for large datasets with a large number of features and training examples. By employing the feature-values that are selected by EPM, the performance improves significantly. We show that incorporating the mined feature-values makes the coreference resolver robust across domains and improves the out-of-domain performance significantly.

9.2 Future Work

In this section we discuss two possible future venues for the work that is presented in this thesis:

- **Incorporating semantic knowledge in coreference resolution**

  In this work, we only apply EPM on shallow semantic features, e.g. gender and NER tags. The incorporation of semantics in coreference resolution has a contradicting history in coreference resolution. Based on linguistic intuitions, semantic information is supposed to be useful for coreference resolution. However, there have been found arguments for and against this intuition, e.g. Hobbs (1978) and Kehler et al. (2004).

  Selectional preference is one of these semantic features. For instance, in "The Titanic hit an iceberg. It sank quickly.", a ship can sink, but an iceberg cannot, i.e. "sink" as the governor of "it" creates certain preferences for the "it" candidate antecedents. By using selectional preferences we can correctly resolve the pronoun "it".

  In Heinzerling et al. (2017), we model selectional preferences using a dependency-based embedding model. We compute a similarity value between the acquired embeddings of anaphor and those of candidate antecedents and incorporate the computed similarities as pairwise features. While our selectional preference model allows fine-grained compatibility judgments with high coverage, its incorporation in a state-of-the-art coreference resolver does not result in a significant improvement.

  In future work, we incorporate the similarity values of the selectional preference model among the EPM features. Thus, we can verify for which types of anaphor and in combination with which other features, the employment of selectional preference similarities is beneficial for coreference resolution.

- **Applying EPM to incorporate linguistic features in other NLP tasks**

  The use of linguistic features in state-of-the-art NLP systems that use deep neural networks and word embeddings does not have a consistent effect on the overall performance and on the generalization of the baseline model. For instance, for the task of sentence compression, Wang et al. (2017) show that the model gets more robust across domains by incorporating linguistic features, i.e. part of speech tags and syntactic relations. On the other hand, Marcheggiani & Titov (2017) propose a graph convolutional network to incorporate syntactic information in semantic role labeling. They show that their model performs worse than its syntactic-agnostic counterpart in out-of-domain evaluations.

  The use of EPM for selecting an informative set of linguistic feature-values for other NLP tasks is an interesting future direction. This way we can investigate whether the question is the efficacy of linguistic features themselves in state-of-the-art NLP tasks or the efficacy of the mechanisms by which we incorporate linguistic features.

List of Figures

List of Tables

List of Algorithms

Bibliography

Agrawal, Rakesh & Ramakrishnan Srikant (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pp. 487–499.

Agresti, Alan (2007). *An Introduction To Categorical Data Analysis*. John Wiley & Sons.

Alcalá-Fdez, Jesús, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez & Francisco Herrera (2011). Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17.

Bagga, Amit & Breck Baldwin (1998). Algorithms for scoring coreference chains. In *Proceedings of the 1st International Conference on Language Resources and Evaluation,* Granada, Spain, 28–30 May 1998, pp. 563–566.

Bahdanau, Dzmitry, Kyunghyun Cho & Yoshua Bengio (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Barker, Chris & Geoffrey K Pullum (1990). A theory of command relations. *Linguistics and Philosophy*, 13(1):1–34.

Batal, Iyad & Milos Hauskrecht (2010). Constructing classification features using minimal predictive patterns. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pp. 869–878.

Bay, Stephen D. & Michael J. Pazzani (2001). Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery*, 5(3):213–246.

Bengio, Yoshua, Aaron Courville & Pascal Vincent (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Bengtson, Eric & Dan Roth (2008). Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing,* Waikiki, Honolulu, Hawaii, 25–27 October 2008, pp. 294–303.

Björkelund, Anders & Richárd Farkas (2012). Data-driven multilingual coreference resolution using resolver stacking. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pp. 49–55.

Björkelund, Anders & Jonas Kuhn (2014). 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),* Baltimore, Md., 22–27 June 2014, pp. 47–57.

Björkelund, Anders & Pierre Nugues (2011). Exploring lexicalized features for coreference resolution. In *Proceedings of the Shared Task of the 15th Conference on Computational Natural Language Learning,* Portland, Oreg., 23–24 June 2011, pp. 45–50, Association for Computational Linguistics.

Brennan, Susan E., Marilyn W. Friedman & Carl J. Pollard (1987). A centering approach to pronouns. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics,* Stanford, Cal., 6–9 July 1987, pp. 155–162.

Bringmann, Björn, Siegfried Nijssen & Albrecht Zimmermann (2009). Pattern-based classification: A unifying perspective. In *Proceedings of the ECML PKDD 2009 Workshop 'From Local Patterns to Global Models'*.

Chaimongkol, Panot, Akiko Aizawa & Yuka Tateisi (2014). Corpus for coreference resolution on scientific papers. In *Proceedings of the 9th International Conference on Language Resources and Evaluation,* Reykjavik, Iceland, 26–31 May 2014, pp. 3187–3190.

Chang, Kai-Wei, Rajhans Samdani & Dan Roth (2013). A constrained latent variable model for coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing,* Seattle, Wash., 18–21 October 2013, pp. 601–612.

Chang, Kai-Wei, Rajhans Samdani, Alla Rozovskaya, Mark Sammons & Dan Roth (2012). Illinois-Coref: The UI system in the CoNLL-2012 shared task. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pp. 113–117.

Chen, Chen & Vincent Ng (2013). Linguistically aware coreference evaluation metrics. In *Proceedings of the 6th International Joint Conference on Natural Language Processing,* Nagoya, Japan, 14–18 October 2013, pp. 1366–1374.

Cheng, Hong, Xifeng Yan, Jiawei Han & Chih-Wei Hsu (2007). Discriminative frequent pattern analysis for effective classification. In *Proceedings of the IEEE 23rd International Conference on Data Engineering (ICDE 2007)*, pp. 716–725.

Cheng, Hong, Xifeng Yan, Jiawei Han & Philip S Yu (2008). Direct discriminative pattern mining for effective classification. In *Proceedings of the IEEE 24th International Conference on Data Engineering (ICDE 2008)*, pp. 169–178.

Choi, Eunsol, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste & Jonathan Berant (2017). Coarse-to-fine question answering for long documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Vancouver, B.C., Canada, 30 July –4 August 2017, Vol. 1, pp. 209–220.

Clark, Kevin & Christopher D. Manning (2015). Entity-centric coreference resolution with model stacking. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Beijing, China, 26–31 July 2015, pp. 1405–1415.

Clark, Kevin & Christopher D. Manning (2016a). improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Berlin, Germany, 7–12 August 2016.

Clark, Kevin & Christopher D. Manning (2016b). deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,* Austin, Tex., 1–5 November 2016, pp. 2256–2262.

Cohen, K. Bretonnel, Arrick Lanfranchi, William Corvey, William A. Baumgartner Jr., Christophe Roeder, Philip V. Ogren, Martha Palmer & Lawrence E. Hunter (2010). Annotation of all coreference in biomedical text: Guideline selection and adaptation. In *Proceedings of the 2nd Workshop on Building and Evaluating Resources for Biomedical Text Mining,* Valetta, Malta, 17 May 2010, pp. 37–41.

Collins, Michael (1999). *Head-Driven Statistical Models for Natural Language Parsing*, (Ph.D. thesis). Philadelphia, Penn.: University of Pennsylvania.

Culotta, Aron, Michael Wick & Andrew McCallum (2007). First-order probabilistic models for coreference resolution. In *Proceedings of Human Language Technologies 2007: The*

Conference of the North American Chapter of the Association for Computational Linguistics, Rochester, N.Y., 22–27 April 2007, pp. 81–88.

Daumé III, Hal & Daniel Marcu (2005). A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proceedings of the Human Language Technology Conference and the 2005 Conference on Empirical Methods in Natural Language Processing,* Vancouver, B.C., Canada, 6–8 October 2005, pp. 97–104.

Denis, Pascal & Jason Baldridge (2008). Specialized models and ranking for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing,* Waikiki, Honolulu, Hawaii, 25–27 October 2008, pp. 660–669.

Denis, Pascal & Jason Baldridge (2009a). Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, 42:87–96.

Denis, Pascal & Jason Baldridge (2009b). Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, (42):87–96.

Dhingra, Bhuwan, Zhilin Yang, William W Cohen & Ruslan Salakhutdinov (2017). *Linguistic Knowledge as Memory for Recurrent Neural Networks*.

Dunning, Ted (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Durrett, Greg, Taylor Berg-Kirkpatrick & Dan Klein (2016). Learning-based single-document summarization with compression and anaphoricity constraints. pp. 1998–2008.

Durrett, Greg & Dan Klein (2013). Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing,* Seattle, Wash., 18–21 October 2013, pp. 1971–1982.

Elsner, Micha & Eugene Charniak (2008). Coreference-inspired coherence modeling. In *Proceedings ACL-HLT 2008 Conference Short Papers,* Columbus, Ohio, 15–20 June 2008, pp. 41–44.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang & Chih-Jen Lin (2008). LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

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

Fernandes, Eraldo Rezende, Cícero Nogueira dos Santos & Ruy Luiz Milidiú (2012). Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pp. 41–48.

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

Finkel, Jenny Rose & Christopher Manning (2008). Enforcing transitivity in coreference resolution. In *Companion Volume to the Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics,* Columbus, Ohio, 15–20 June 2008, pp. 45–48.

Finley, Thomas & Thorsten Joachims (2005). Supervised clustering with support vector machines. In *Proceedings of the 22nd international conference on Machine learning*, pp. 217–224, ACM.

Ghaddar, Abbas & Philippe Langlais (2016a). Coreference in Wikipedia: Main concept resolution. In *Proceedings of the 20th Conference on Computational Natural Language Learning,* Berlin, Germany, 11–12 August 2016, pp. 229–238.

Ghaddar, Abbas & Philippe Langlais (2016b). WikiCoref: An English coreference-annotated corpus of Wikipedia articles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation,* Portorož, Slovenia, 23–28 May 2016.

Grosz, Barbara J., Aravind K. Joshi & Scott Weinstein (1995). Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Guillou, Liane, Christian Hardmeier, Preslav Nakov, Sara Stymne, Jörg Tiedemann, Yannick Versley, Mauro Cettolo, Bonnie L Webber & Andrei Popescu-Belis (2016). Findings of the 2016 wmt shared task on cross-lingual pronoun prediction. In *Proceedings of the First Conference on Machine Translation , Volume 2: Shared Task Papers*, pp. 525–542.

Han, Jiawei, Jian Pei, Yiwen Yin & Runying Mao (2004). Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1-5):53–87.

Hardmeier, Christian, Preslav Nakov, Sara Stymne, Jörg Tiedemann, Yannick Versley & Mauro Cettolo (2015). Pronoun-focused mt and cross-lingual pronoun prediction: Findings of the 2015 discomt shared task on pronoun translation. In *Second Workshop on Discourse in Machine Translation (DiscoMT), 17 September 2015, Lisbon, Portugal*, pp. 1–16, Association for Computational Linguistics.

Heaton, Jeff (2017). *Comparing Dataset Characteristics that Favor the Apriori, Eclat or FP-Growth Frequent Itemset Mining Algorithms*. arXiv:1701.09042.

Heinzerling, Benjamin, Nafise Sadat Moosavi & Michael Strube (2017). Revisiting selectional preferences for coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* Copenhagen, Denmark, 7–11 September 2017, pp. 1343–1350.

Hendrickx, Iris, Gosse Bouma, Frederik Coppens, Walter Daelemans, Veronique Hoste, Geert Kloosterman, Anne-Marie Mineur, Joeri Van Der Vloet & Jean-Luc Verschelde (2008). A coreference corpus and resolution system for dutch. In *Proceedings of the 6th International Conference on Language Resources and Evaluation,* Marrakech, Morocco, 26 May – 1 June 2008.

Hinrichs, Erhard W, Sandra Kübler & Karin Naumann (2005). A unified representation for morphological, syntactic, semantic, and referential annotations. In *Proceedings of the Workshop on Frontiers in Corpus Annotations II: Pie in the Sky*, pp. 13–20, Association for Computational Linguistics.

Hirschman, Lynette & Nancy Chinchor (1997). *MUC-7 Coreference Task Definition,* `http://www.muc.saic.com/proceedings/`.

Hobbs, Jerry R. (1978). Resolving pronominal references. *Lingua*, 44:311–338.

Hochreiter, Sepp & Jürgen Schmidhuber (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Holen, Gordana Ilic (2013). Critical reflections on evaluation practices in coreference resolution. In *Proceedings of the 2013 NAACL HLT Student Research Workshop,* Atlanta, Georgia, 9-14 June 2013, pp. 1–7.

Hummel, Robert A & Steven W Zucker (1983). On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (3):267–287.

Kehler, Andrew, Douglas Appelt, Lara Taylor & Aleksandr Simma (2004). The (non)utility of predicate-argument frequencies for pronoun interpretation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics,* Boston, Mass., 2–7 May 2004, pp. 289–296.

Klein, Dan & Christopher D. Manning (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics,* Sapporo, Japan, 7–12 July 2003, pp. 423–430.

Klenner, Manfred (2007). Enforcing consistency on coreference sets. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing,* Borovets, Bulgaria, 27–29 September 2007, pp. 323–328.

Klenner, Manfred & Don Tuggener (2011). An incremental model for coreference resolution with restrictive antecedent accessibility. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pp. 81–85, Association for Computational Linguistics.

Kong, Fang, GuoDong Zhou & Qiaoming Zhu (2009). Employing the centering theory in pronoun resolution from the semantic perspective. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing,* Singapore, 6–7 August 2009, pp. 987–996, Association for Computational Linguistics.

Kuhn, Harold W (1955). The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Lassalle, Emmanuel & Pascal Denis (2013). Improving pairwise coreference models through feature space hierarchy learning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Sofia, Bulgaria, 4–9 August 2013, pp. 497–506.

Lassalle, Emmanuel & Pascal Denis (2015). Joint anaphoricity detection and coreference resolution with constrained latent structures. In *Proceedings of the 29th Conference on the Advancement of Artificial Intelligence,* Austin, Texas, 25–30 January 2015, pp. 2274–2280.

Le, Phong & Ivan Titov (2017). Optimizing differentiable relaxations of coreference evaluation metrics. *Proceedings of CoNLL.*

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

Lee, Heeyoung, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu & Dan Jurafsky (2011). Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proceedings of the Shared Task of the 15th Conference on Computational Natural Language Learning,* Portland, Oreg., 23–24 June 2011, pp. 28–34.

Lee, Kenton, Luheng He, Mike Lewis & Luke Zettlemoyer (2017). End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* Copenhagen, Denmark, 7–11 September 2017, pp. 188–197.

Levy, Omer, Steffen Remus, Chris Biemann & Ido Dagan (2015). Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* Denver, Col., 31 May – 5 June 2015, pp. 970–âĂŞ976.

Li, Wenmin, Jiawei Han & Jian Pei (2001). CMAR: Accurate and efficient classification based on multiple class-association rules. In *Proceedings of the IEEE International Conference on Data Mining (ICDM 2001)*, pp. 369–376.

Lichman, M. (2013). *UCI Machine Learning Repository*.

Liu, Bing, Wynne Hsu & Yiming Ma (1998). Integrating classification and association rule mining. In *Proceedings of the 1998 International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pp. 80–86.

Luo, Xiaoqiang (2005). On coreference resolution performance metrics. In *Proceedings of the Human Language Technology Conference and the 2005 Conference on Empirical Methods in Natural Language Processing,* Vancouver, B.C., Canada, 6–8 October 2005, pp. 25–32.

Luo, Xiaoqiang, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla & Salim Roukos (2004). A mention-synchronous coreference resolution algorithm based on the Bell Tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics,* Barcelona, Spain, 21–26 July 2004, pp. 136–143.

Luo, Xiaoqiang & Sameer Pradhan (2016). Evaluation metrics. In M. Poesio, R. Stuckardt & Y. Versley (Eds.), *Anaphora Resolution: Algorithms, Resources, and Applications*. Springer. To appear.

Luo, Xiaoqiang, Sameer Pradhan, Marta Recasens & Eduard Hovy (2014). An extension of BLANC to system mentions. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 24–29. Baltimore, Maryland: Association for Computational Linguistics.

Ma, Chao, Janardhan Rao Doppa, J. Walker Orr, Prashanth Mannem, Xiaoli Fern, Tom Dietterich & Prasad Tadepalli (2014). Prune-and-score: Learning for greedy coreference resolution. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing,* Doha, Qatar, 25–29 October 2014, pp. 2115–2126.

Marcheggiani, Diego & Ivan Titov (2017). Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1506–1515. Copenhagen, Denmark: Association for Computational Linguistics.

Martschat, Sebastian & Michael Strube (2014). Recall error analysis for coreference resolution. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing,* Doha, Qatar, 25–29 October 2014, pp. 2070–2081.

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

Matsuzaki, Takuya, Takumi Ito, Hidenao Iwane, Hirokazu Anai & Noriko H Arai (2017). Semantic parsing of pre-university math problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Vancouver, B.C., Canada, 30 July –4 August 2017, Vol. 1, pp. 2131–2141.

McCallum, Andrew & Ben Wellner (2003). Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of the IJCAI-03 Workshop on Information Integration on the Web,* Acapulco, Mexico, August 9–10, 2003, pp. 79–86.

Mikolov, Tomas, Kai Chen, Greg Corrado & Jeffrey Dean (2013a). Efficient estimation of word representations in vector space. In *Proceedings of the ICLR 2013 Workshop Track*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado & Jeff Dean (2013b). Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. Curran Associates, Inc.

Mitchell, Alexis, Stephanie Strassel, Mark Przybocki, JK Davis, George Doddington, Ralph Grishman, Adam Meyers, Ada Brunstain, Lisa Ferro & Beth Sundheim (2002). *ACE-2 Version 1.0*. LDC2003T11, Philadelphia, Penn.: Linguistic Data Consortium.

Mitkov, Ruslan (2002). *Anaphora Resolution*. London, U.K.: Longman.

Moosavi, Nafise Sadat, Leo Born, Massimo Poesio & Michael Strube (2019). Using automatically extracted minimum spans to disentangle coreference evaluation from boundary detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4168–4178. Florence, Italy: Association for Computational Linguistics.

Moosavi, Nafise Sadat & GholamReza GhassemSani (2014). *Unsupervised Coreference Resolution Using a Graph Labeling Approach*, pp. 93–103. Cham: Springer International Publishing.

Moosavi, Nafise Sadat & Michael Strube (2014). Unsupervised coreference resolution by utilizing the most informative relations. In *Proceedings of the 25th International Conference on Computational Linguistics,* Dublin, Ireland, 23–29 August 2014, pp. 644–655.

Moosavi, Nafise Sadat & Michael Strube (2016a). Search space pruning: A simple solution for better coreference resolvers. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* San Diego, Cal., 12–17 June 2016, pp. 1005–1011.

Moosavi, Nafise Sadat & Michael Strube (2016b). Which coreference evaluation metric do you trust? A proposal for a link-based entity aware metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Berlin, Germany, 7–12 August 2016, pp. 632–642.

Moosavi, Nafise Sadat & Michael Strube (2017a). Lexical features in coreference resolution: To be used with caution. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers),* Vancouver, B.C., Canada, 30 July –4 August 2017.

Moosavi, Nafise Sadat & Michael Strube (2017b). Use generalized representations, but do not forget surface features. In *Proceedings of the 2nd Workshop on Coreference Resolution Beyond OntoNotes,* Valencia, Spain, 4 April 2017, pp. 1–7.

Moosavi, Nafise Sadat & Michael Strube (2018). Using linguistic features to improve the generalization capability of neural coreference resolvers. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 193–203. Brussels, Belgium: Association for Computational Linguistics.

Munkres, James (1957). Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38.

Ng, Vincent (2007). Shallow semantics for coreference resolution. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence,* Hyderabad, India, 6–12 January 2007, pp. 1689–1694.

Ng, Vincent (2009). Graph-cut-based anaphoricity determination for coreference resolution. In *Proceedings of Human Language Technologies 2009: The Conference of the North American Chapter of the Association for Computational Linguistics,* Boulder, Col., 31 May – 5 June 2009, pp. 575–583.

Ng, Vincent & Claire Cardie (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics,* Philadelphia, Penn., 7–12 July 2002, pp. 104–111.

Nicolae, Cristina & Gabriel Nicolae (2006). BestCut: A graph algorithm for coreference resolution. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing,* Sydney, Australia, 22–23 July 2006, pp. 275–283.

Noreen, Eric W. (1989). *Computer Intensive Methods for Hypothesis Testing: An Introduction*. New York, N.Y.: Wiley.

Novak, Petra Kralj, Nada Lavrač & Geoffrey I Webb (2009). Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. *The Journal of Machine Learning Research*, 10:377–403.

Peng, Haoruo, Kai-Wei Chang & Dan Roth (2015a). A joint framework for coreference resolution and mention head detection. In *Proceedings of the 19th Conference on Computational Natural Language Learning,* Beijing, China, 30–31 July 2015, pp. 12–21.

Peng, Haoruo, Kai-Wei Chang & Dan Roth (2015b). A joint framework for coreference resolution and mention head detection. In *Proceedings of the 19th Conference on Computational Natural Language Learning,* Beijing, China, 30–31 July 2015, pp. 12–21.

Poesio, Massimo (2016). Linguistic and cognitive evidence about anaphora. In M. Poesio, R. Stuckardt & Y. Versley (Eds.), *Anaphora Resolution: Algorithms, Resources, and Applications*. Springer.

Ponzetto, Simone Paolo & Michael Strube (2006). Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics,* New York, N.Y., 4–9 June 2006, pp. 192–199.

Pradhan, Sameer, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng & Michael Strube (2014). Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers),* Baltimore, Md., 22–27 June 2014, pp. 30–35.

Pradhan, Sameer, Alessandro Moschitti, Nianwen Xue, Olga Uryupina & Yuchen Zhang (2012). CoNLL-2012 Shared Task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pp. 1–40.

Pradhan, Sameer, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel & Nianwen Xue (2011). CoNLL-2011 Shared Task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Shared Task of the 15th Conference on Computational Natural Language Learning,* Portland, Oreg., 23–24 June 2011, pp. 1–27.

Raghunathan, Karthik, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky & Christopher Manning (2010). A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing,* Cambridge, Mass., 9–11 October 2010, pp. 492–501.

Rahman, Altaf & Vincent Ng (2011). Narrowing the modeling gap: A cluster-ranking approach to coreference resolution. *Journal of Artificial Intelligence Research*, 40:469–521.

Rand, William R. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

Recasens, Marta & Eduard Hovy (2011). BLANC: Implementing the Rand index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.

Recasens, Marta, Lluís Màrquez, Emili Sapena, M Antònia Martí, Mariona Taulé, Véronique Hoste, Massimo Poesio & Yannick Versley (2010). Semeval-2010 task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 1–8, Association for Computational Linguistics.

Recasens, Marta & M. Antónia Martí (2009). AnCoraCo: coreferentially annotated corpora for Spanish and Catalan. *Language Resources and Evaluation*, 43(4).

Recasens, Marta & Marta Vila (2010). On paraphrase and coreference. *Computational Linguistics*, 36(4):639–647.

Ringgaard, Michael, Rahul Gupta & Fernando C. N. Pereira (2017). *SLING: A framework for frame semantic parsing*. arXiv:1710.07032.

Rodrıguez, Kepa Joseba, Francesca Delogu, Yannick Versley, Egon W Stemle & Massimo Poesio (2010). Anaphoric annotation of wikipedia and blogs in the live memories corpus. In *Proceedings of the 7th International Conference on Language Resources and Evaluation,* La Valetta, Malta, 17–23 May 2010, pp. 157–163.

Sapena, Emili, Lluís Padró & Jordi Turmo (2010). A global relaxation labeling approach to coreference resolution. In *Proceedings of Coling 2010: Poster Volume,* Beijing, China, 23–27 August 2010, pp. 1086–1094.

Schäfer, Ulrich, Christian Spurk & Jörg Steffen (2012). A fully coreference-annotated corpus of scholarly papers from the ACL anthology. In *Proceedings of Coling 2012: Poster Volume,* Mumbai, India, 8–15 December 2012, pp. 1059–1070.

Segond, Marc & Christian Borgelt (2011). Item set mining based on cover similarity. *Advances in Knowledge Discovery and Data Mining*, pp. 493–505.

Sil, Avirup & Radu Florian (2017). One for all: Towards language independent named entity linking. pp. 2255–2264.

Song, Yang, Jing Jiang, Wayne Xin Zhao, Sujian Li & Houfeng Wang (2012). Joint learning for coreference resolution with markov logic. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1245–1254, Association for Computational Linguistics.

Soon, Wee Meng, Hwee Tou Ng & Daniel Chung Yong Lim (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Stede, Manfred (2004). The potsdam commentary corpus. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation*, pp. 96–102, Association for Computational Linguistics.

Stoer, Mechthild & Frank Wagner (1997). A simple min-cut algorithm. *Journal of the ACM (JACM)*, 44(4):585–591.

Stoyanov, Veselin & Jason Eisner (2012). Easy-first coreference resolution. In *Proceedings of the 24th International Conference on Computational Linguistics,* Mumbai, India, 8–15 December 2012, pp. 2519–2534.

Stoyanov, Veselin, Nathan Gilbert, Claire Cardie & Ellen Riloff (2009). Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing,* Singapore, 2–7 August 2009, pp. 656–664.

Stuckhardt, Roland (2003). Coreference-based summarization and question answering: A case for high precision anaphor resolution. In *Proceedings of the 2003 International Symposium on Reference Resolution and Its Applications to Question Answering and Summarization,* Venice, Italy, 23–24 June 2003, pp. 33–42.

Sun, Xu, Takuya Matsuzaki, Daisuke Okanohara & Jun'ichi Tsujii (2009). Latent variable perceptron algorithm for structured classification. In *Proceedings of the 21th International Joint Conference on Artificial Intelligence,* Pasadena, Cal., 14–17 July 2009, Vol. 9, pp. 1236–1242.

Sutton, Charles, Michael Sindelar & Andrew McCallum (2006). Reducing weight undertraining in structured discriminative learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, San Diego, Cal., 12–17 June 2016, pp. 89–95, Association for Computational Linguistics.

Sutton, Richard S & Andrew G Barto (1998). *Reinforcement learning: An introduction*, Vol. 1. MIT press Cambridge.

Tuggener, Don (2014). Coreference resolution evaluation for higher level applications. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, Volume 2: Short Papers,* Gothenburg, Sweden, 26–30 April 2014, pp. 231–235.

Uryupina, Olga (2007). *Knowledge acquisition for coreference resolution*, (Ph.D. thesis). Saarland University.

Uryupina, Olga, Ron Artstein, Antonella Bristot, Federica Cavicchio, Kepa Rodriguez & Massimo Poesio (2016). Arrau: Linguistically-motivated annotation of anaphoric descriptions. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk & Stelios Piperidis (Eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Paris, France: European Language Resources Association (ELRA).

Uryupina, Olga & Alessandro Moschitti (2015). A state-of-the-art mention-pair model for coreference resolution. In *Proceedings of STARSEM 2015: The Fourth Joint Conference on Lexical and Computational Semantics,* Denver, Col., 4–5 June 2015, pp. 289–298.

van Deemter, Kees & Rodger Kibble (2000). On coreferring: Coreference in MUC and related annotation schemes. *Computational Linguistics*, 26(4):629–637.

Vilain, Marc, John Burger, John Aberdeen, Dennis Connolly & Lynette Hirschman (1995). A model-theoretic coreference scoring scheme. In *Proceedings of the 6th Message Understanding Conference (MUC-6)*, pp. 45–52. San Mateo, Cal.: Morgan Kaufmann.

Walker, Marilyn A. (1998). Centering, anaphora resolution, and discourse structure. In M.A. Walker, A.K. Joshi & E.F. Prince (Eds.), *Centering Theory in Discourse*, pp. 401–435. Oxford, U.K.: Oxford University Press.

Wang, Liangguo, Jing Jiang, Hai Leong Chieu, Chen Hui Ong, Dandan Song & Lejian Liao (2017). Can syntax help? improving an lstm-based sentence compression model for new domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1385–1393. Vancouver, Canada: Association for Computational Linguistics.

Webb, Geoffrey I. (2006). Discovering significant patterns. *Machine Learning*, 68(1):1–39.

Weischedel, Ralph, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini et al. (2013). Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*.

Wiseman, Sam, Alexander M. Rush & Stuart Shieber (2016). 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,* San Diego, Cal., 12–17 June 2016, pp. 994–1004.

Wiseman, Sam, Alexander M. Rush, Stuart Shieber & Jason Weston (2015). Learning anaphoricity and antecedent ranking features for coreference resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Beijing, China, 26–31 July 2015, pp. 1416–1426.

Yang, Guizhen (2006). Computational aspects of mining maximal frequent patterns. *Theoretical Computer Science*, 362(1-5):63–86.

Yang, Xiaofeng, Jian Su, Jun Lang, Chew Lim Tan, Ting Liu & Sheng Li (2008). An entity-mention model for coreference resolution with Inductive Logic Programming. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* Columbus, Ohio, 15–20 June 2008, pp. 843–851.

Yang, Xiaofeng, Jian Su, Guodung Zhou & Chew Lim Tan (2004). Improving pronoun resolution by incorporating coreferential information of candidates. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics,* Barcelona, Spain, 21–26 July 2004, pp. 128–135.

Yang, Xiaofeng, Guodung Zhou, Jian Su & Chew Lim Tan (2003). Coreference resolution using competition learning approach. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics,* Sapporo, Japan, 7–12 July 2003, pp. 176–183.

Yu, Chun-Nam John & Thorsten Joachims (2009). Learning structural SVMs with latent variables. In *Proceedings of the 26th International Conference on Machine Learning,* Montréal, Québec, Canada, 14–18 June 2009, pp. 1169–1176.

Yu, Dian & Heng Ji (2016). Unsupervised person slot filling based on graph mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Berlin, Germany, 7–12 August 2016, pp. 44–53.

Zhou, Guodong & Fang Kong (2009). Global learning of noun phrase anaphoricity in coreference resolution via label propagation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing,* Singapore, 6–7 August 2009, pp. 978–986.

Zimmermann, Albrecht & Siegfried Nijssen (2014). Supervised pattern mining and applications to classification. In Charu C. Aggarwal & Jiawei Han (Eds.), *Frequent Pattern Mining*, pp. 425–442. Springer International Publishing.