10
Coreference Resolution

The CoNLL-2012 Shared Task (Pradhan et al., 2012) is dedicated to the modeling of coreference resolution for multiple languages. The participants are provided with datasets for three languages: Arabic, Chinese and English. These datasets are provided by the OntoNotes project and, besides accurate coreference information, contain various annotation layers such as part-of-speech (POS) tagging, syntax parsing, named entities (NE) and semantic role labeling (SRL). The shared task consists in the automatic identification of coreferring mentions of entities and events, given predicted information on other OntoNotes layers. The official ranking for this task is given by the mean score on the three languages. We take part in the CoNLL-2012 Shared Task *closed track*, in which training data is restricted to the information provided by the shared task organizers. We propose a language-independent approach based on ESL and submit its results to the shared task (Fernandes et al., 2012b). The developed systems obtain the very best performance among all participants. In this chapter, we describe this ESL application.

Coreference resolution consists in identifying mention clusters in a document. Mentions are textual references to real world entities, like people, companies or places. In Figure 10.1, we present an illustrative document with nine highlighted mentions. In a given document, mentions that refer to the

North Korea$_{a_1}$ opened **its**$_{a_2}$ doors to the **U.S.**$_{b_1}$ today, welcoming **Secretary of State Madeleine Albright**$_{c_1}$. **She**$_{c_2}$ says **her**$_{c_3}$ visit is a good start. The **U.S.**$_{b_2}$ remains concerned about **North Korea's**$_{a_3}$ missile development program and **its**$_{a_4}$ exports of missiles to Iran.

Figure 10.1: Document with nine highlighted mentions that refer to three different entities: North Korea is referenced by mentions $\{a_1, a_2, a_3, a_4\}$; the U.S. is referenced by $\{b_1, b_2\}$; and Madeleine Albright by $\{c_1, c_2, c_3\}$. The letter in the mention subscript identifies its entity cluster and the number uniquely identifies the mention within its cluster.

same entity are called *coreferring mentions* and form an *entity cluster*. In the example, the letter in a mention subscript indicates its entity cluster and the number uniquely identifies the mention within its cluster. There are three entity clusters in the example that are related to the following real

world entities: North Korea, which is identified by the letter a; the United States, which is identified by b; and Madeleine Albright, which is identified by c. The coreference resolution task is to identify entity mentions in a given document and to cluster the coreferring mentions. Clusters that comprise only one mention are ignored. For instance, in the example, the mention Iran is ignored.

The remainder of this chapter is organized as follows. In Section 10.1, we formalize the coreference resolution task. In this chapter, we propose a novel structure learning modeling for this task. In Section 10.2, we present the feature factorization used in this modeling. The resulting prediction problem is equivalent to the maximum branching problem, just as for dependency parsing. However, we use a slightly different loss function. These aspects are discussed in Section 10.3. We describe the basic features provided to ESL in Section 10.4. We apply our ESL-based coreference modeling to three very different languages, since our modeling is highly language independent. Nevertheless, some datasets lack basic features and we need to adapt some parts of the systems. In Section 10.5, we describe these adaptations and some additional preprocessing procedures. Finally, in Section 10.6, we present our empirical results.

10.1 Task Formalization

Regarding our ESL modeling, the input for the coreference resolution task is a set of mentions $\boldsymbol{x} = \{x_1, \ldots, x_N\}$ within a document. The task is to cluster the coreferring mentions together, that is, mentions that are references to the same entity are in the same cluster. A feasible output is then a set of non-overlapping clusters $\boldsymbol{y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K\}$, where $\boldsymbol{y}_i \subset \boldsymbol{x}$; $\boldsymbol{y}_i \cap \boldsymbol{y}_j = \emptyset$, for $i, j \in \{1, \ldots, K\}$ and $i \neq j$; and K is unknown. Additionally, *singleton* mentions are ignored. A singleton is a unique mention to its entity in the input document.

10.2 Feature Factorization

Usually, coreference systems use features that depend on pairs of mentions (x_i, x_j). We follow this idea, but we introduce a novel modeling for coreference resolution. Most clustering metrics lead to NP-hard optimization problems. Hence, we assume that an entity cluster is represented by a rooted tree. A directed edge (i, j) in this tree indicates that x_j is a reference to the more general mention x_i.

10.2.1 Coreference Trees

We introduce *coreference trees* to represent clusters of coreferring mentions. A coreference tree is a rooted tree whose nodes are the coreferring mentions and arcs represent *some* coreference relation between mentions. In Figure 10.1, we present a document with nine highlighted mentions comprising three clusters. One plausible coreference tree for the cluster $\{a_1,a_2,a_3,a_4\}$ is presented in Figure 10.2. We are not really concerned about the semantics

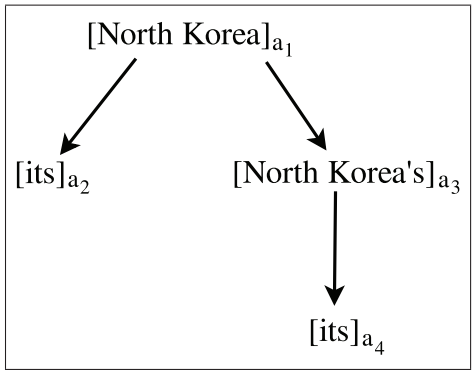

Figure 10.2: Coreference tree for the cluster a in Figure 10.1.

underlying coreference trees, since they are just auxiliary structures for the clustering task. However, we argue that this concept is linguistically plausible, since usually there is indeed a specific-to-general relation between two coreferring mentions. Observing the aforementioned example, one may agree that mention a_3 (North Korea's) is more likely to be associated with mention a_1 (North Korea) than with mention a_2 (its), even considering that a_2 and a_3 are closer to each other than a_1 and a_3, in the document text.

For a given document, we have a forest of coreference trees, one tree for each entity cluster. However, for the sake of simplicity, we link the root node of every coreference tree to an *artificial* root node, obtaining the *document tree*. In Figure 10.3, we depict a document tree for the text in Figure 10.1.

10.2.2 Latent Structure Learning

Coreference trees are not given in the training data. Thus, we assume that these structures are *latent* and make use of the latent structure perceptron (Fernandes and Brefeld, 2011; Yu and Joachims, 2009) to train our models. We introduced this algorithm earlier in Figure 2.3. Here, we describe its application to coreference resolution by using coreference trees. We decompose the original predictor into two predictors, namely the *latent predictor* $F_h(\boldsymbol{x};\boldsymbol{w})$ and the

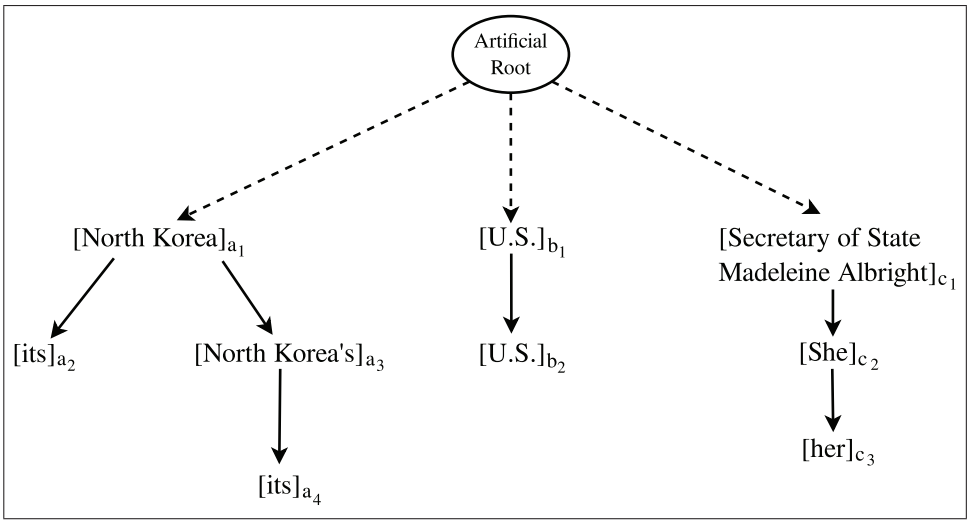

Figure 10.3: Document tree with three coreference trees that corresponds to the text in Figure 10.1. Dashed lines indicate artificial arcs.

target predictor $F_y(\boldsymbol{x}, \boldsymbol{h})$. The latent predictor is defined as

$$F_h(\boldsymbol{x}; \boldsymbol{w}) = \arg \max_{\boldsymbol{h} \in \mathcal{H}(\boldsymbol{x})} \langle \boldsymbol{w}, \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{h}) \rangle,$$

where $\mathcal{H}(\boldsymbol{x})$ is the set of feasible document trees for $\boldsymbol{x}$ and $\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{h})$ is the joint feature vector representation for the mentions $\boldsymbol{x}$ and the document tree $\boldsymbol{h}$. Hence, the latent predictor finds a maximum scoring rooted tree over the given mentions $\boldsymbol{x}$, where a tree score is given by a linear function over its features. $F_y(\boldsymbol{x}, \boldsymbol{h})$ is a straightforward procedure that creates a cluster for each subtree connected to the artificial root node in the document tree $\boldsymbol{h}$. Then, for a given input $\boldsymbol{x}$, a complete prediction is given by $F_y(\boldsymbol{x}, F_h(\boldsymbol{x}; \boldsymbol{w}))$.

As one can observe, in this application of the latent SPerc, we do not use the target model $\boldsymbol{w}_y$ introduced in Section 2.3, since the target predictor $F_y(\boldsymbol{x}, \boldsymbol{h})$ predicts an output based exclusively on the latent structure $\boldsymbol{h}$. Thus, in this chapter, the model $\boldsymbol{w}$ corresponds to the latent model $\boldsymbol{w}_h$ presented in Chapter 2.

In Figure 10.4, we depict the latent structure perceptron algorithm for the mention clustering task. Likewise its binary counterpart (Rosenblatt, 1957), the structure perceptron is an online algorithm that iterates through the training set. For each training instance, it performs two major steps: (i) a prediction for the given input using the current model; and (ii) a model update based on the difference between the predicted and the ground truth outputs. The latent SPerc performs an additional step to predict the latent ground truth $\tilde{\boldsymbol{h}}$ by using a specialization of the latent predictor.

Golden coreference trees are not available, however, during training, for

$\boldsymbol{w}_0 \leftarrow \mathbf{0}$
$t \leftarrow 0$
while no convergence
  for each $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$
    $\tilde{\boldsymbol{h}} \leftarrow \arg\max_{h \in \mathcal{H}(\boldsymbol{x},\boldsymbol{y})} \langle \boldsymbol{w}_t, \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{h}) \rangle$
    $\hat{\boldsymbol{h}} \leftarrow \arg\max_{h \in \mathcal{H}(\boldsymbol{x})} \langle \boldsymbol{w}_t, \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{h}) \rangle + \ell(\boldsymbol{h}, \tilde{\boldsymbol{h}})$
    $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \boldsymbol{\Phi}(\boldsymbol{x}, \tilde{\boldsymbol{h}}) - \boldsymbol{\Phi}(\boldsymbol{x}, \hat{\boldsymbol{h}})$
    $t \leftarrow t + 1$
$\boldsymbol{w} \leftarrow \frac{1}{t} \sum_{k=1}^{t} \boldsymbol{w}_k$

Figure 10.4: Latent structure perceptron algorithm.

an input $\boldsymbol{x}$, we have the golden clustering $\boldsymbol{y}$. Thus, we predict the *constrained document tree* $\tilde{\boldsymbol{h}}$ for the training instance $(\boldsymbol{x}, \boldsymbol{y})$ using a specialization of the latent predictor – the *constrained latent predictor* – that makes use of $\boldsymbol{y}$. The constrained predictor finds the maximum scoring document tree among the *constrained document tree set* $\mathcal{H}(\boldsymbol{x}, \boldsymbol{y}) \subset \mathcal{H}(\boldsymbol{x})$, which includes all rooted trees of $\boldsymbol{x}$ that follow the correct clustering $\boldsymbol{y}$. That is, a constrained tree $\boldsymbol{h} \in \mathcal{H}(\boldsymbol{x}, \boldsymbol{y})$ comprises only arcs between coreferring mentions – according to $\boldsymbol{y}$ – plus one arc from the artificial node to each cluster. In that way, the constrained predictor guarantees that $F_y(\tilde{\boldsymbol{h}}) = \boldsymbol{y}$, for any $\boldsymbol{w}$. The constrained tree is then used as the ground truth on each iteration. Therefore, the model update is determined by the difference between the constrained document tree and the document tree predicted by the ordinary predictor.

The latent structure perceptron algorithm learns to predict document trees that help to solve the clustering task. Thereafter, for an unseen document $\boldsymbol{x}$, the latent predictor $F_h(\boldsymbol{x}; \boldsymbol{w})$ is employed to produce a predicted document tree $\boldsymbol{h}$ which, in turn, is fed to $F_y(\boldsymbol{x}, \boldsymbol{h})$ to give the predicted clusters.

10.3 Prediction Problem

We decompose the joint feature vector $\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{h})$ along coreference tree edges, that is, mention pairs. Thus, the prediction problem is reduced to the maximum branching problem, just as for dependency parsing, and it can be efficiently solved by the Chu-Liu-Edmonds algorithm.

We use a loss function that is similar to the one used for dependency parsing

$$\ell(\boldsymbol{h}, \hat{\boldsymbol{h}}) = \sum_{(i,j) \in \hat{\boldsymbol{h}}; i > 0} \mathbf{1}[(i,j) \notin \boldsymbol{h}] + \sum_{(i,j) \in \hat{\boldsymbol{h}}; i = 0} r \cdot \mathbf{1}[(i,j) \notin \boldsymbol{h}]$$

where $(0, j)$ is an artificial edge and r is a meta-parameter denoted *root loss value*. This loss function just counts how many predicted edges are not present in the constrained document tree. Additionally, for arcs from the artificial root node, we use a different loss value r.

10.4 Basic Features

We use 70 basic features to describe a candidate edge. All of them give hints on the coreference strength of individual edges. These features provide lexical, syntactic, semantic, and positional information. They have been adapted from previously proposed features dos Santos and Carvalho (2011); Sapena et al. (2010); Ng and Cardie (2002). All features have been transformed into categorical, even the integer ones.

In Table 10.1, we briefly describe the set of basic features used in our system. The *Id* column identifies each feature. The *Type* column indicates the value type of each feature, such as boolean (*yes*, *no*) or ternary (*yes*, *no*, *not applicable*). The # column indicates how many basic features correspond to each description.

10.5 Data Preparation

In this section, we present some specific procedures that are performed before the application of the ESL system to coreference resolution.

10.5.1 Mention Detection

The CoNLL-2012 shared task datasets do not explicitly provide entity mentions; the system needs to detect them. For each document, we generate a list of candidate mentions using the strategy of dos Santos and Carvalho (2011). The basic idea is to use all noun phrases and, additionally, pronouns and named entities, even if they are inside larger noun phrases. We do not include verbs as mentions.

10.5.2 Coreference Arcs Generation

The input for the prediction problem is a graph whose nodes are the mentions in a document. Ideally, we could consider the complete graph for each document, thus every mention pair would be an option for building the document tree. However, since the total number of mentions is huge and a big

Id	Description	Type	#
	Lexical Features		*25*
L1	Head word of x_i (x_j)	word	2
L2	String matching of x_i and x_j	boolean	1
L3	String matching of the head words of x_i and x_j	boolean	1
L4	Both x_i and x_j are pronouns and their strings match	ternary	1
L5	Both x_i and x_j are *not* pronouns and their string match	ternary	1
L6	Previous and next two words of x_i (x_j)	word	8
L7	Length of x_i (x_j)	integer	2
L8	Edit distance of head words x_i and x_j	integer	1
L9	Edit distance of x_i and x_j after removing determiners	integer	1
L10	x_i (x_j) is a definitive noun phrase	boolean	2
L11	x_i (x_j) is a demonstrative noun phrase	boolean	2
L12	The head word of both x_i and x_j are proper nouns	boolean	1
L13	Both x_i and x_j are proper names and their strings match	ternary	1
L14	Both x_i and x_j are proper names and their head word strings match	ternary	1
	Syntactic Features		*28*
Sy1	POS tag of the head word of x_i (x_j)	POS tag	2
Sy2	Previous and next two POS tags of x_i (x_j)	POS tag	8
Sy3	x_i (x_j) is a pronoun	boolean	2
Sy4	Gender of x_i (x_j), if pronoun	f, m, n/a	2
Sy5	x_i and x_j are both pronouns and agree in gender	ternary	1
Sy6	x_i and x_j are both pronouns and agree in number	ternary	1
Sy7	x_i (x_j) is a proper name	boolean	2
Sy8	x_i and x_j are both proper names	boolean	1
Sy9	Previous and next predicate of x_i (x_j)	verb	4
Sy10	x_i and x_j are pronouns and agree in number, gender and person	ternary	1
Sy11	Noun phrase embedding level of x_i (x_j) in the syntactic parse	integer	2
Sy12	Number of embedded noun phrases in x_i (x_j)	integer	2
	Semantic Features		*13*
Se1	The prediction of the baseline system proposed in dos Santos and Carvalho (2011)	binary	1
Se2	Sense of the head word of x_i (x_j)	sense	2
Se3	Named entity type of x_i (x_j)	NE tag	2
Se4	x_i and x_j have the same named entity	ternary	1
Se5	Semantic role for the previous and next words of x_i (x_j)	SRL tag	4
Se6	Concatenation of semantic roles of x_i and x_j for the same predicate, if they are in the same sentence	(SRL tag)^2	1
Se7	x_i and x_j have the same speaker	ternary	1
Se8	x_j is an alias of x_i	boolean	1
	Positional Features		*4*
P1	Distance between x_i and x_j in number of sentences	integer	1
P2	Distance between x_i and x_j in number of mentions	integer	1
P3	Distance between x_i and x_j in number of person names (applies only for the cases where x_i and x_j are both pronouns or one of them is a person name)	integer	1
P4	One mention is in apposition to the other	boolean	1

Table 10.1: Description of all 70 basic features.

portion of arcs can be easily identified as incorrect, we filter the arcs and, thus, include only candidate mention pairs that are more likely to be coreferent.

We filter arcs by simply adapting the sieves method proposed by Lee et al. (2011) to English coreference resolution. Lee et al. propose a list of handcrafted

rules that are sequentially applied to mention pairs in order to iteratively merge mentions into entity clusters. These rules are denoted *sieves*, since they filter the correct mention pairs. In Lee et al.'s system, sieves are applied from higher to lower precision. However, in our filtering strategy, precision is not a concern and the application order is not important. The objective here is to build a small set of candidate arcs that shows good recall. Additionally, we do not have interest on sieves that are strongly language dependent, since our target is multilingual coreference resolution. We thus select the most general sieves, which can be easily applied to the Arabic and Chinese datasets provided in the CoNLL-2012 shared task.

Given a mention pair (x_i, x_j), where x_i appears before x_j in the text, we create a directed arc (i, j) if at least one of the following conditions holds:

1. *Distance* – The number of mentions between x_i and x_j is not greater than a given parameter.

2. *Alias* – If both mentions are people, check if the head word of one mention is part of the other mention, like Dilma and Dilma Rousseff. If both mentions are organizations, check if the head word of one mention is contained in the other, or if one is the acronym of the other.

3. *Relaxed String Match* – There is a match of both mentions up to their head words.

4. *Head Word Match* – The head word of x_i matches the head word of x_j.

5. *Shallow Discourse* – Test if shallow discourse attributes match for both mentions. For instance, two first person pronouns assigned to the same speaker are considered coreferent.

6. *Pronouns* – Check if x_j is a pronoun and x_i has the same gender, number, speaker and animacy of x_j. For this filter, we use number and gender data provided by Bergsma and Lin (2006).

7. *Pronouns/NE* – Check if x_j is a pronoun and x_i is a compatible pronoun or proper name (named entity).

Sieves 2 to 7 are adapted from Lee et al. (2011). Most of these sieves are relaxed versions of the ones proposed by Lee et al. (2011). Sieve 1 is introduced by us to lift recall, yet avoiding strongly language-dependent sieves.

10.5.3 Language Specifics

Our system can be easily adapted to different languages. In our experiments, only minor changes are needed to train and apply the system to three different languages. The adaptations are due to: (i) lack of input features for some languages; (i) different POS tagsets across datasets; and (iii) creation of static lists of language specific pronouns. The necessary adaptations are restricted to only two preprocessing steps: basic features and coreference arcs generation.

Some input features available in the English dataset are not available in the Arabic nor in the Chinese datasets. The Arabic dataset does not contain named entity, semantic role labeling and speaker features. Therefore, for Arabic, we do not derive the following basic features: Sy9, Se3, Se4, Se5, Se6, Se7, and P3. For Chinese, information related to named entity is not provided. Thus, we do not derive the following basic features: Se3, Se4, and P3. Additionally, the Chinese dataset uses a different POS tagset. Hence, some mappings are used during the basic feature derivation stage.

The lack of input features for Arabic and Chinese also impact the sieve based arc generation. For Chinese, we do not use sieve 6, and, for Arabic, we only use sieves 1, 3, 4 and 7. Sieve 7 is not used for the English dataset, since it is a specialization of sieve 6. The first sieve threshold is 4 for Arabic and Chinese, and 8 for English.

In the arc generation and basic feature derivation steps, our system makes use of static lists of language specific pronouns. In our experiments, we use the POS tagging information and the golden entity clusters to automatically extract these pronoun lists from training data.

Our system submitted to the CoNLL-2012 Shared Task ignores arcs linking *nested* mentions. While this kind of mentions are never coreferent in Arabic nor in English, the Chinese datasets include many nested coreferring mentions. Hence, in the latest version of our system, we include such arcs for the Chinese language.

10.5.4 EFG Setting

We experiment with different template sets for each language. The difference between these sets is the training data given as input to EFG. We obtain better results when merging different template sets. For the English language, it is better to use a set of 196 templates obtained by merging the output of two independent EFG executions. These two runs are fed with

training datasets comprising: (a) mention pairs produced by sieves 2 to 6; and (b) mention pairs produced by all sieves. For Chinese and Arabic, it is better to use template sets generated specifically for these languages and merge them with the template set (a), generated for the English language. The final set for Chinese comprises 197 templates, while the final set for Arabic comprises 223. All these templates conjoin from two to seven basic features.

10.5.5 Evaluation Metrics

Evaluating coreference systems is a hard task. The main issue is that coreference information is highly faceted and the value of each facet varies a lot from one application to another. Thus, when reporting and comparing coreference performances, it is really hard to define *one* metric that fits all purposes. Therefore, we follow the methodology proposed in the CoNLL-2012 Shared Task to assess our systems, since it combines three of the most popular metrics. The metrics used are the link based MUC metric (Vilain et al., 1995), the mention based B^3 metric (Bagga and Baldwin, 1998) and the entity based $CEAF_e$ metric (Luo, 2005). All these metrics are based on precision and recall measures, which are combined to give an F-score value. The mean F-score of these three metrics gives a unique score for each language. Additionally, when appropriate, the *official* CoNLL-2012 Shared Task score is reported, which is the average of the F-scores for all languages. We denote this metric as *CoNLL score*.

Another important aspect of coreference evaluation is mention matching. Some methodologies, like the ones used in MUC or ACE evaluations, consider approximate matching of mention spans. However, the CoNLL-2012 Shared Task evaluation considers only exact span matching. We use the latter in our performance measures. In fact, the experimental results reported in this work are generated by the official CoNLL-2012 Shared Task evaluation scripts.

10.6 Empirical Results

In this section, we present five sets of empirical findings on the CoNLL-2012 Shared Task datasets. Namely, (i) we show our system overall quality, that is, the best one for Arabic, Chinese and English; (ii) we assess the EFG impact, showing that it significantly improves the resulting system quality; (iii) we assess the root loss value impact, also showing that it significantly improves system quality; (iv) we show that by enhancing our Chinese modeling with nested mentions, we achieve state-of-the-art quality

for this language; and (v) we present the supplementary results provided by the shared task organizers. These empirical findings highlight the main contributions of this work regarding multilingual unrestricted coreference resolution on OntoNotes.

10.6.1 State-of-the-art Systems

In Table 10.2, we present per-language and CoNLL scores of the best performing systems on the CoNLL-2012 test sets. The first row of this table

Reference	AR	CH	EN	CoNLL Score
This work	**54.22**	**62.87**	**63.37**	**60.15**
Fernandes et al. (2012a)	**54.22**	58.49	**63.37**	58.69
Björkelund and Farkas (2012)	53.55	59.97	61.24	58.25
Chen and Ng (2012)	47.13	62.24	59.69	56.35

Table 10.2: State-of-the-art systems for multilingual unrestricted coreference resolution in OntoNotes. Performances on the CoNLL-2012 Shared Task test sets.

corresponds to the last version of our system and the second row corresponds to our official entry in the CoNLL-2012 Shared Task. The difference between these two versions is the inclusion of candidate arcs linking *nested* mentions for the Chinese language. By including such arcs, the score increases almost 4.5 points for that language.

The last two rows of Table 10.2 correspond to the competitors that are ranked second Björkelund and Farkas (2012) and third Chen and Ng (2012) in the shared task. Our system obtains a remarkable performance on the English language, outperforming the runner-up by more than two points. We also achieve the highest performance on Arabic and Chinese.

The detailed performance of our systems is presented in Table 10.3, where we report recall, precision and F-score for all metrics and languages considered in the CoNLL-2012 Shared Task. We can observe that the mean scores on Chinese and English are similar. On the other hand, the performance on the Arabic language is much lower. Given the smaller size of the Arabic training corpus, this variation is expected.

Lang	MUC			B^3			$CEAF_e$			Mean
	R	P	F	R	P	F	R	P	F	
Arabic	43.63	49.69	46.46	62.70	72.19	67.11	52.49	46.09	49.08	**54.22**
Chinese	59.20	71.52	64.78	67.17	80.55	73.25	57.46	45.20	50.59	**62.87**
English	65.83	75.91	70.51	65.79	77.69	71.24	55.00	43.17	48.37	**63.37**
									CoNLL Score	**60.15**

Table 10.3: Detailed performance of our system on the CoNLL-2012 Shared Task test sets.

10.6.2
Entropy Guided Feature Generation

In this work, we employ entropy guided feature generation to automatically generate non-linear features that conjoin the used 70 basic features. In this section, we compare our EFG-based system with a system trained with basic features alone. It is important to notice that among these 70 basic features there are several complex features. Some of these features are even conjunctions of other simpler basic features, and others provide complex task dependent information, like head words and agreement on number and gender, for instance. These 70 basic features were manually generated by domain experts and encode valuable coreference information.

In Table 10.4, we present the performances of four systems on the English development set. In the upper half of this table, we report the performance of our EFG-based system (first row) and the performance of a model trained with basic features alone (second row). We can notice that the EFG system outperforms the baseline by 7.31 points. Moreover, EFG consistently outperforms the baseline on all metrics.

Basic Feats.	**EFG**	**MUC**			**B^3**			**$CEAF_e$**			**Mean**
		R	P	F_1	R	P	F_1	R	P	F_1	
70	Yes	61.34	75.71	67.77	62.94	79.59	70.29	56.23	40.36	46.99	**61.68**
	No	51.32	73.28	60.37	54.85	78.06	64.43	50.87	30.71	38.30	**54.37**
54	Yes	60.86	74.82	67.12	62.50	78.83	69.72	54.53	39.46	45.79	**60.88**
	No	36.65	73.44	48.90	45.25	82.26	58.38	49.97	22.09	30.64	**45.97**

Table 10.4: EFG effect on system performance for the English development set.

We perform another experiment to assess EFG. We remove 16 basic features out of the 70 original ones and perform the same experiment as before. That is, we evaluate an EFG-based system trained with the remaining 54 basic features and compare it to another system trained with the same 54 basic features alone. Namely, we remove the following basic features: L2, L3, L4, L5, L8, L9, L12, L13, L14, Sy5, Sy8, Sy10, Se1, Se4, Se7, P4. In the lower half of Table 10.4, we present the performance of these two systems. We can see that, while the EFG-based system performance (third row) drops only 0.8 point when the 16 features are removed, the performance of the baseline system (fourth row) drops impressive 8.4 points. The difference between the two systems doubles in respect to the difference when using all 70 basic features. These findings highlight two important points. First, the removed features are very informative. Moreover, EFG is able to almost completely overcome the omission of these informative features by automatically generating conjunctions of the remaining 54 basic features.

10.6.3 Root Loss Value

Just as some coreference metrics can be more important than others for some applications, precision and recall have different values for applications. Specifically for the CoNLL score – which is based on the $F_{\beta=1}$ score – the balance between precision and recall is important. For this reason, we introduce one important parameter in our system: the *root loss value*. This parameter specifies a different loss function value for outgoing arcs in the artificial root node. Observe that, in a document tree, each arc from the root node corresponds to a cluster. The effect of a root loss value larger than one is to reduce the creation of new clusters, stimulating larger clusters. Therefore, one can use this parameter to adjust the balance between precision and recall.

In the upper half of Table 10.5, we present our system performances on the development sets when we set this parameter to one, which is equivalent to not use this parameter at all. We can notice that in this case recall and precision have very distinct values, lowering the F-score values. Using the

Root Loss	**Lang**	**MUC**			**B^3**			**$CEAF_e$**			**Mean**
		R	P	F	R	P	F	R	P	F	
Off	Arabic	34.18	58.85	43.25	50.61	82.13	62.63	57.37	33.75	42.49	**49.45**
	Chinese	49.17	76.03	59.72	58.16	86.33	69.50	57.56	34.38	43.05	**57.42**
	English	62.75	77.41	69.31	63.88	81.34	71.56	57.46	41.08	47.91	**62.92**
										CoNLL Score	**56.59**
On	Arabic	43.00	47.87	45.30	61.41	70.38	65.59	49.42	44.19	46.66	**52.52**
	Chinese	54.40	68.19	60.52	64.17	78.84	70.76	51.42	38.96	44.33	**58.54**
	English	64.88	74.74	69.46	66.53	78.28	71.93	54.93	43.68	48.66	**63.35**
										CoNLL Score	**58.14**

Table 10.5: Root loss value effect on development set performances.

development sets for tuning, we set the root loss value to 6, 2 and 1.5 for Arabic, Chinese and English, respectively. In the lower half of Table 10.5, we present the performances when we use these values for the root loss value parameter. We can observe that this parameter really causes a better balancing between precision and recall, consequently increasing the F-score values. Its effect is accentuated on Arabic and Chinese, since the unbalancing issue is worse on these languages. The increase in the CoNLL score is over 1.5 point.

10.6.4 Chinese Nested Mentions

Nested noun phrases are very common. For instance, the noun phrase the smart boy includes the nested noun phrase boy. Whether to consider these two noun phrases as coreferring mentions or only consider the longer noun phrase as a mention is an annotation design choice. OntoNotes mostly consider only the

longer noun phrase. However, in many Chinese documents, nested mentions are annotated as coreferring. Thus, in this work, we evaluate the effect of whether arcs linking nested mentions are considered or not. In Table 10.6, we present the detailed results when such arcs are ignored (first row) and when they are included (second row). To consider these arcs remarkably increases our system score by almost 4 points on the Chinese language.

Nested Mentions	**MUC**			**B^3**			**CEAF$_e$**			**Mean**
	R	P	F_1	R	P	F_1	R	P	F_1	
Yes	60.35	70.56	65.05	67.37	79.49	72.93	55.15	44.94	49.52	**62.50**
No	54.40	68.19	60.52	64.17	78.84	70.76	51.42	38.96	44.33	**58.54**

Table 10.6: Effect whether nested coreferring mentions are considered or not for the Chinese language.

10.6.5 Supplementary Results

We report in Table 10.7 the supplementary results provided by the CoNLL-2012 Shared Task organizers on the test sets. These additional

Lang	**Config**	**MUC**			**B^3**			**CEAF$_e$**			**Mean**
		R	P	F_1	R	P	F_1	R	P	F_1	
AR	A/A	43.63	49.69	46.46	62.70	72.19	67.11	52.49	46.09	49.08	**54.22**
	A/GB	45.18	47.39	46.26	64.56	69.44	66.91	49.73	47.39	48.53	**53.90**
	A/GM	57.25	76.48	65.48	60.27	79.81	68.68	72.61	46.00	56.32	**63.49**
	G/A	46.38	51.78	48.93	63.53	72.37	67.66	52.57	46.88	49.56	**55.38**
	G/GB	46.38	51.78	48.93	63.53	72.37	67.66	52.57	46.88	49.56	**55.38**
	G/GM	56.89	76.27	65.17	60.07	80.02	68.62	72.24	45.58	55.90	**63.23**
CH	A/A	52.69	70.58	60.34	62.99	80.57	70.70	53.75	37.88	44.44	**58.49**
	A/GB	58.76	71.46	64.49	66.62	79.88	72.65	54.09	42.02	47.29	**61.48**
	A/GM	61.64	90.81	73.43	63.55	89.43	74.30	72.78	39.68	51.36	**66.36**
	G/A	59.35	74.49	66.07	66.31	81.43	73.10	55.97	41.50	47.66	**62.28**
	G/GB	59.35	74.49	66.07	66.31	81.43	73.10	55.97	41.50	47.66	**62.28**
	G/GM	61.70	91.45	73.69	63.57	89.76	74.43	72.84	39.49	51.21	**66.44**
EN	A/A	65.83	75.91	70.51	65.79	77.69	71.24	55.00	43.17	48.37	**63.37**
	A/GB	64.92	77.53	70.67	64.25	78.95	70.85	56.48	41.69	47.97	**63.16**
	A/GM	70.69	91.21	79.65	65.46	85.61	74.19	74.71	42.55	54.22	**69.35**
	G/A	67.73	77.25	72.18	66.42	78.01	71.75	56.16	44.51	49.66	**64.53**
	G/GB	65.65	78.26	71.40	64.36	79.09	70.97	57.36	42.23	48.65	**63.67**
	G/GM	71.18	91.24	79.97	65.81	85.51	74.38	74.93	43.09	54.72	**69.69**

Table 10.7: Supplementary results on the test sets with different configurations (Config) for parse quality and mention candidates (parse/mentions). Parse quality can be automatic (A) or golden (G); and mention candidates can be automatically identified (A), golden mention boundaries (GB) or golden mentions (GM).

experiments investigate two key aspects of any coreference resolution system: the parse feature and the mention candidates that are given to the clustering procedure. In these results, we alternate the parse feature between the official

automatic parse (A in the results table) and the *golden* parse from OntoNotes (G). Regarding mention candidates, we use three different strategies: automatic mentions (A), golden mention boundaries (GB) and golden mentions (GM). Automatic mentions are the ones detected by our system. Golden mention boundaries comprise all noun phrases in the *golden* parse tree, even when the automatic parse is used as input feature. Golden mentions are all non-singleton mentions, i.e., all mentions that take part in some entity cluster. It is important to notice that golden mention information is much stronger than just golden boundaries.

By observing Table 10.7, it is clear that the most beneficial information is golden mentions (compare A/GM to A/A rows, for each language). The mean F-score over all languages when using golden mentions is almost 8 points higher than the official score. These results are not surprising since to identify non-singleton mentions accounts to a significant part of the final task. Golden mention boundaries (A/GB) increase the Chinese score by almost 3 points. Conversely, for the other two languages, the results are decreased when this information is given. This is probably due to parameter tuning, since any additional information potentially changes the learning problem and, nevertheless, we use exactly the same three models – one per language – to produce both the official and the supplementary results. One can observe, for instance, that the recall/precision balance greatly varies among the different configurations in these experiments. The golden parse feature (G/A) causes big improvements on all languages, specially Chinese.