Multilingual Mention Detection for Coreference Resolution

Olga Uryupina
DISI, University of Trento, Italy
uryupina@gmail.com

Alessandro Moschitti
QCRI, Qatar Foundation, Doha, Qatar
amoschitti@qf.org.qa

Abstract

This paper proposes a novel algorithm for multilingual mention detection: we extract mentions from parse trees via kernel-based SVM learning. Our approach allows for straightforward mention detection for any language where (not necessary perfect) parsing resources are available, without any complex language-specific rule engineering. We also investigate possibilities for incorporating automatically acquired mentions into an end-to-end coreference resolution system. We evaluate our approach on the Arabic and Chinese portions of the CoNLL-2012 dataset, showing a significant improvement over the system with the baseline mention detection.

1 Introduction

Accurate mention detection (MD) is a vital prerequisite for a variety of Natural Language Processing tasks, in particular, for Relation Extraction (RE) and Coreference Resolution (CR). If a toolkit cannot extract mentions reliably, it will obviously be unable to assign them to relations or entities.

Many studies on RE and CR report evaluation figures on *gold* mentions: in such a setting, a system is supplied with correct mention boundaries and/or semantic classes or other relevant properties. It can, in theory, be argued that such a methodology provides better insights on performance of RE and CR algorithms per se. It has been demonstrated, however, that evaluation results on gold mentions are misleading: for example, Ng (2008) shows that unsupervised CR algorithms exhibit promising results on gold mentions, that are not mirrored in a more realistic evaluation on automatically detected mentions.

The exact scope of the mention detection task varies considerably depending on the annotation guidelines. Thus, some corpora consider all the (non-embedding) NPs to be mentions, some corpora do not allow for non-referential mentions and some do not mark singleton referential mentions, that do not participate in coreference relations. In addition, some guidelines may restrict the annotation to specific semantic types.

A number of linguistic studies focus on various syntactic, semantic and discourse clues that might help identify nominal constructions that cannot participate in coreference relations. Possible features include, among others, specific syntactic constructions for expletive pronouns, negation, modality and quantification (Karttunen, 1976). Several algorithms have been proposed recently, trying to tackle some of the addressed phenomena within a computational approach. Thus, a number of algorithms have been developed recently to identify expletive usages of "it" (Evans, 2001; Boyd et al., 2005; Bergsma and Yarowsky, 2011). While these approaches are potentially beneficial for mention detection in English, for other languages, neither theoretical nor computational studies are available at the moment. In this paper, we use tree kernels to extract relevant syntactic patterns automatically, without assuming any prior knowledge of the input language.

In this paper, we propose a learning-based solution to the mention detection task. We use SVMs (Joachims, 1999) with syntactic tree kernels (Collins and Duffy, 2001; Moschitti, 2008; Moschitti, 2006) to classify parse tree nodes as $\pm$mentions. Our approach does not require any language- or corpus-specific engineering and thus can be easily adapted to cover new languages or mention annotation schemes.

The rest of paper is organized as follows. In the next section, we define the task and discuss our tree and vector representations. Section 4 presents MD evaluation figures. Finally, in Section 5 we incorporate our MD module into an end-to-end coreference resolution system.

2 Related Work

Until recently, most RE and CR toolkits have been evaluated on the ACE datasets (Doddington et al., 2004). The ACE guidelines restrict possible mentions to be considered to specific semantic types (PERSON, LOCATION and so on). Moreover, mentions are annotated with their minimal and maximal span, allowing for relaxed matching between gold and automatically extracted boundaries. In such a setting, the mention detection task can be cast as a tagging problem, similar to the named entity recognition and classification task. A number of systems have followed this scenario, demonstrating reliable performance (Florian et al., 2004; Ittycheriah et al., 2003; Zitouni and Florian, 2008).

In the past years, however, several corpora have been created from a more linguistic perspective: for example, the OntoNotes dataset (Hovy et al., 2006; Pradhan et al., 2012) provides annotation for unrestricted coreference. The guidelines differ significantly from the ACE scheme: mentions correspond to parse nodes and can be of any semantic type, the systems are expected to recover mention boundaries exactly. The OntoNotes mentions—unlike ACE ones–correspond to large NP structures (embedding NP nodes in gold parse trees), so a traditional approach (e.g., one of those mentioned above), which aims at identifying basic NP chunks, would not be applicable here. Therefore, any MD method for OntoNotes would rely on parsing.

The OntoNotes corpus has been used for evaluating end-to-end CR systems at two CoNLL shared tasks (2011 and 2012). At the 2011 shared task, the participants relied on rule-based modules for extracting mention boundaries from parse trees. This was relatively straightforward, as the task was devoted to CR in English and most participants could use their in-house MD modules developed and refined in the past decade. At the 2012 shared task, however, the systems were expected to provide end-to-end coreference resolution for Arabic and Chinese. As it turned out, most groups could not adapt their MD rules to cover these two languages and fell back to very simple baselines (e.g., "use all NP nodes as mentions"). Kummerfeld et al. (2011) investigated various post- and pre-filtering heuristics for adapting their mention detection algorithm to the OntoNotes English data in a semi-automatic way, reporting mixed results.

3 Mention extraction from parse trees

We recast MD as a *node filtering* task: each candidate node is classified as either mention or not. In this study, we consider all "NP" nodes to be candidates for MD. As Table 1 shows, this is a reasonable assumption for the OntoNotes dataset, as almost 90% of all the mentions for both Arabic and Chinese correspond to NP nodes. The remaining 11-14% of mentions can mostly be attributed to parsing errors: as we aim at end-to-end processing with no gold information available, we run our system on automatically extracted parse trees, it is therefore possible that a mention corresponds to a gold NP node that has not been labeled correctly in an automatic parse tree.

	train		development	
	NP-nodes	%	NP-nodes	%
ARB	24068	87.23	2916	87.91
CHN	88523	85.96	12572	88.52

Table 1: NP-nodes in OntoNotes for Arabic (ARB) and Chinese (CHN): total numbers and percentage of mentions that are NP-nodes.

Not all the NP nodes, however, correspond to a mention. Such non-mention NPs fall into several categories:

- **Embedded NPs.** When an NP is embedded into another one, only the outer NP is used to represent a mention:

  (1) $[_{MENTION-NP}[_{NP}$This type] of earthquake] has no precursors. [1]

  A number of heuristics have been proposed for English to identify and discard embedded NPs, based on available head-finding algorithms, e.g., (Collins, 1999). For other languages, however, the task of finding a head of a given NP in a constituency tree is not trivial.

- **Non-referential NPs.** Depending on the annotation guidelines, non-referential NPs can

[1]We use English OntoNotes examples throughout this paper to illustrate discussed phenomena, as our approach is language-independent. The evaluation, however, is done on Arabic and Chinese.

either be marked as mentions or not. In OntoNotes, non-referential NPs should not be annotated:

(2) This type of earthquake has [$_{NP}$no precursors].

- **Singleton NPs.** In some CR corpora (for example, ACE), mentions are annotated even if they do not participate in any coreference relations. In other corpora (MUC and OntoNotes), such *singletons* are not marked. When singletons are not marked, the MD tasks becomes considerably more difficult: the performance of an MD component cannot be measured and optimized directly, but only in conjunction with a coreference resolver.

- **Erroneous NPs.** When we evaluate an end-to-end system, we expect it to process raw input and thus rely on automatically extracted parse trees. Some NP-nodes might be incorrect, not corresponding to any NP in the gold tree. Such nodes cannot be mentions:

(3) At the meeting, Huang Xiangning read [$_{NP}$the earthquake prediction] that they had previously issued.

"The earthquake prediction" is considered to be an NP node by the parser. In the gold data, however, this node does not exist at all. And even if it existed, the mention should correspond to its embedding NP node, "the earthquake prediction that the had previously issued" (cf. example 1 above). While this problem is less crucial for English, parsing resources for other languages are still scarce and less reliable.

3.1 Tree Representation

We use kernel-based SVMs to classify nodes as ±mentions. This requires representing a relevant fragment of a tree with a specific node marked as "C-NP" (candidate). We start from a straightforward representation: using automatically generated parse trees provided within the CoNLL data distribution, we generate one example for each NP node: the example corresponds to the entire parse tree with just a single node re-labeled as "C-NP". The assigned class label reflects the fact that this particular node corresponds to some gold mention or not. For example, the full parse tree for our sentence (1-2) will generate one positive (for "This type of earthquake", shown on Figure 1) and three negative examples (for "This type", "earthquake" and "no precursors").

While this representation might work for our toy example, for a longer sentence it would provide irrelevant information. Consider again the tree on Figure 1. To generate a training example we append "C-" to one NP node, keeping all the remaining nodes as-is. The tree kernel operates on subtrees of the given structure, so, effectively, it will consider a lot of tree fragments that do not contain the marked node. These fragments will affect the treatment of different examples, possibly with conflicting class labels. It will not only make learning slow but also introduce spurious evidence, decreasing the system's performance. We have therefore investigated two possibilities for pruning our trees.

Our first pruning algorithm ("up-down") starts from the node of interest (C-NP) and goes up for u nodes. From each node on the path, it considers all its children up to the depth d. The first part of Figure 2 shows a pruned tree for $u = 2, d = 1$ for the node "This type of earthquake."

Our second pruning algorithm ("radial") starts from the node of interest and considers all the nodes in the tree that are reachable from it via at most n edges. The second part of Figure 2 shows a pruned tree for $n = 2$ for the same node.

3.2 Vector Representation

In addition to (pruned) trees, we also provide vector representations of our NPs. For each NP, we extract its basic properties: number, gender, person, mention type (name, nominal or pronoun) and the number of other NPs in the document that have the same surface form. To extract mention properties, we have to compute the head. However, the goal of our study is to provide an MD algorithm that is adaptable to different languages without extensive engineering. We have therefore deliberately relied on an over-simplistic heuristic for finding an NP head: either the last or the first noun in an NP is considered a head, depending on some very basic information on a word order in a specific language. Given the head, we extract its properties from the CoNLL data in a straightforward way (for example, we have compiled a list of pronouns with their gender, number and person values from the training data and so on). This is done fully automatically and doesn't require any

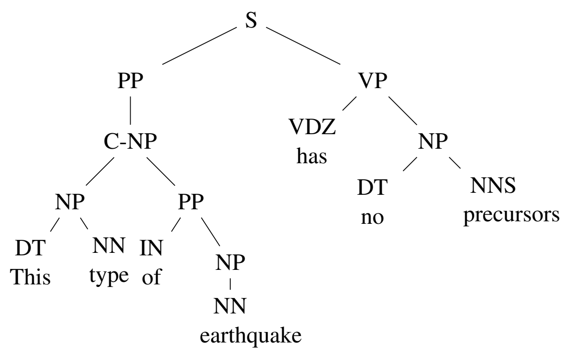
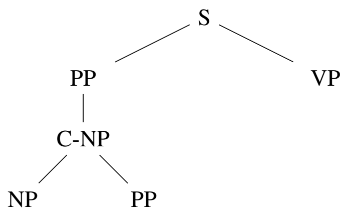


Figure 1: Parse tree for “This type of earthquake”, examples (1-2): before pruning.



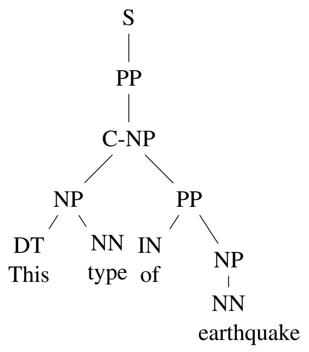


Figure 2: Up-down (left) vs. radial (right) pruning for “This type of earthquake,” examples (1-2)

language-specific manual engineering.

Table 2 lists the features for our vector representation. Nominal values are binarized, leading to 10 binary or continuous features.

feature	possible values
Gender	F,M,Unknown
Definiteness	Yes, No
Number	Sg,Pl,Du,Unknown
MentionType	Name,Nominal,Pronoun
#same-surface NPs in the doc	continuous (normalized)

Table 2: Features used for Mention Detection: each feature describes an individual NP

4 Evaluating MD

In this section we provide evaluation results on both Arabic and Chinese. We reserve a small portion of the CoNLL training data (around 20k instances for each language) for training an MD system. Another small subset (around 5k instances) is reserved for fitting the system parameters. The evaluation results are reported on the CoNLL development data. Note that we evaluate the NP-node classifier, so the system receives no penalty for missing mentions that are not NPs. In Section 5 below, however, we will assess the impact of MD on the end-to-end CR system and thus penalize for missing non-NP mentions.

As a baseline (“all-NP”), we consider all the NP nodes to be mentions. Table 3 below compares this baseline against mentions extracted automatically from different representations. We use Syntactic Tree Kernels (TK) implemented within the SVM-TK toolkit[2] to induce the classification.

As our results suggest, vector representation does not provide enough information for robust mention detection.[3] Indeed, without tree kernels, the system is only able to learn a major class labeling. This highlights the importance of a model that is able to handle structured input, learning relevant patterns directly from parse trees.

As discussed in Section 3 above, full trees contain too much misleading evidence. A single parse tree might contain several dozens of NP nodes, so,

[2]`http://disi.unitn.it/moschitti/Tree-Kernel.htm`

[3]As a pilot experiment, we also added bag-of-words features to our vector representations, but this didn’t yield any improvement.

representation	pruning	R	P	F
Arabic				
all-NP	N/A	100	18.0	30.5
vectors	N/A	0	N/A	N/A
trees	-	53.9	45.5	49.4
trees	d=3, u=1	59.5	50.4	54.6
trees	n=2	59.6	64.7	62.0
trees+vectors	d=3, u=1	65.1	52.9	58.4
trees+vectors	n=2	66.0	66.1	66.1
Chinese				
all-NP	N/A	100	27.9	43.6
vectors	N/A	0	N/A	N/A
trees	-	65.0	55.8	60.0
trees	d=1, u=1	73.0	56.9	63.9
trees	n=2	69.7	63.7	66.6
trees+vectors	d=1, u=1	75.6	60.7	67.4
trees+vectors	n=3	71.3	68.9	70.1

Table 3: Performance of the MD classifier on the development set

using a full sentence tree to represent a particular candidate node provides confusing input for the classifier. This is reflected with a low classifier performance on full representations.

Both pruning strategies have resulted in a substantial improvement in the performance level. The radial pruning has significantly outperformed the up-down strategy. Moreover, the radial pruning depends on just one parameter and can therefore be optimized faster.

Finally, joint vector and tree representation further outperforms a plain tree-based model. It must be noted, however, that our MD features (Table 2) require at least some minimal amount of language-specific engineering.

5 Incorporating TK-based Mention Detection into an end-to-end coreference resolution system

For our experiments, we use BART – a modular toolkit for coreference resolution that supports state-of-the-art statistical approaches to the task and enables efficient feature engineering (Versley et al., 2008). BART has originally been created and tested for English, but its flexible modular architecture ensures its portability to other languages and domains.

In our evaluation experiments, we follow a very simple model of coreference, namely, the mention-pair approach advocated by Soon et al. (2001) and adopted in many studies ever since. We believe, however, that more complex models of coreference will also benefit from our MD algorithm: most state-of-the-art CR systems treat mention detection as a preprocessing step that is not affected by further processing and therefore we expect them to yield better performance when such a preprocessing is achieved in a more robust way.

Creating a robust coreference resolver for a new language requires linguistic expertise and language-specific engineering. This cannot and, moreover, should not be avoided by fully language-agnostic methods. Our approach to end-to-end coreference resolution relies on a universal MD component that requires no linguistic engineering – it facilitates the development of coreference resolvers in the narrow sense, by providing them with input mentions. We must stress that the resolvers themselves are not supposed to be universal: in fact, a number of linguistic studies on coreference address various language-specific challenging problems (e.g., zero pronouns, different marking of information status etc).

Below we describe the adjustments we made to BART to cover Arabic and Chinese and then report on our experiments for integrating kernel-based MD into BART to provide an end-to-end coreference resolution for these languages.

5.1 Adapting BART to Arabic and Chinese

The modularity of the BART toolkit enables its straightforward adaptation to different languages. This includes creating meaningful linguistic representations of mentions ("mention properties") and, optionally, some experiments on feature selection and engineering.

We extracted some properties (sentence boundaries, lemmata, speaker id) for Arabic and Chinese directly from the CoNLL/OntoNotes layers[4]. Mention types are inferred from PoS tags.

We compiled lists of pronouns for both Arabic and Chinese from the training and development data. For Arabic, we used gold PoS tags to classify pronouns into subtypes, person, number and gender. For Chinese, no such information is available, so we consulted several grammar sketches and lists of pronouns on the web. Finally, we extracted a list of gender affixes for Arabic along

[4]Recall that all the layers, apart from the Arabic lemma, were computed using state-of-the-art preprocessing tools by the CoNLL organizers and do not contain gold information

with a list of gender-classified lemmata from the training data.

We assessed the list of features, supported by BART, discarding those that require unavailable information (for example, the `aliasing` feature relies on semantic types for named entities that are not available within the CoNLL/OntoNotes distribution for languages other than English). We also created two additional features: `LemmataMatch` (similar to string match, but uses lemmata instead of tokens) and `NumberAgreementDual` (similar to commonly used number agreement features, but supports dual number). Both features are expected to provide important information for coreference in Arabic, a morphologically rich language.

We ran a feature selection experiment to further remove irrelevant features (BART were only tested on European languages, thus several features reflected patterns more common for Germanic and Romance languages). This resulted in two feature sets, one for each language, listed in Table 4. For comparison, we also show the baseline features (cf. below).

5.2 Incorporating Kernel-based MD into a Coreference Resolver

Coreference resolution systems have different tolerance for precision and recall MD errors. If a spurious mention is introduced, the CR system might still assign it to no coreference chain and thus discard from the output partition. If a correct mention is missed, however, the system has no chance of recovering it as it does not even start processing such a mention. This suggests that an MD module should be tuned to yield better recall.

To assess the impact of MD precision and recall errors on the performance of our coreference resolver, we run a simulation experiment. We start from the upper bound baseline: the MD module considers all the true (gold) NP mentions to be positive and all the spurious ones – to be negative. We then randomly distort this baseline, adding spurious mentions and removing correct ones, to arrive at a predefined performance level. The resulting MD output is then sent to our coreference resolution system and its performance is measured. As a measure of the CR system performance, we use the MELA F-score – an average of MUC, B^3 and $CEAF_e$ metrics, the official performance measure at the CoNLL shared task (Pradhan et al., 2012).

Figure 3 shows the results of our simulation experiment on the development data. Each line on the figure corresponds to a single MD recall level (varying from 100% to 70%). On the horizontal axis, we plot the MD precision (from 10% to 100%) and on the vertical axis – the end-to-end system MELA F-score. The curves support our intuition that reliable MD recall is crucial for coreference: when the MD recall drops to around 70%, the MELA score remains at the baseline level even for very high MD precision. It must be noted that our simulation experiment relies on an unrealistic assumption: we assume all the errors to be independent. In a more practical setting, the MELA F-score for a given combination of MD precision and recall can be higher, because the coreference system might fail to resolve the same NPs that are problematic for the MD module. Nevertheless, the curves illustrate the fact that any MD module should be strongly biased towards recall in order to be useful for coreference resolution.

We therefore reran our optimization experiments to fit more parameters of the MD module. Recall from Section 4 that we already used a small amount of CoNLL training data to fit our d, u and n values. We expanded the set of parameters, using the end-to-end performance (MELA F-score) to select optimal values on the same subset. Table 5 lists all the parameters of our MD module.

d, u	up-down pruning thresholds
n	radial pruning threshold
j	precision-recall trade-off (SVM-TK)
c	cost factor (SVM-TK)
s	size of MD vs. CR data splits
r	tree vs. tree+vector representation

Table 5: Parameters optimized on a held-out data

Our experiments reveal that, indeed, a recall-oriented version of our MD classifier yields the most reliable end-to-end resolution. Table 6 shows the MD performance of the best classifier selected according to the MELA score. While the F-scores of these biased classifiers are, obviously, much lower than their unbiased counterparts, they still manage to filter out a substantial amount of noun phrases, at the same time maintaining a very high recall level.

Finally, Tables 7 and 8 show the MELA score of BART on the CoNLL-2012 development and test sets respectively. To evaluate the impact of our

feature		Baseline	Arabic	Chinese
StringMatch	M_i and M_j have the same surface form	+	+	+
MentionType	relevant types of M_i and M_j, (cf. Soon et al.)	+	+	+
GenderAgree	M_i and M_j agree in gender	+	+	+
NumberAgree	M_i and M_j agree in number	+		+
NumberAgreeDual	-*-, supports dual number		+	
AnimacyAgree	M_i and M_j agree in animacy	+		+
Compatible	M_i and M_j don't disagree or overlap		+	
Alias	heuristical NE-matching	+		
DistanceSentence	distance in sentences between M_i and M_j	+		+
Appositive	M_i and M_j are in an apposition	+		
First_Mention	M_i is the first mention in its sentence		+	
DistanceMarkable	distance in mentions between M_i and M_j		+	
FirstSecondPerson	$M_{i/j}$ is a pronoun of the 1st/second person		+	
NonProSalience	for non-pro M_i, # preceding same-head mentions		+	
SpeakerAlias	heuristics for 1/2 pers. pro, use "speaker" layer			+

Table 4: Features used for Coreference Resolution in Arabic and Chinese: each feature describes a pair of mentions $\{M_i, M_j\},\ i < j$, where M_i is a candidate antecedent and M_j is a candidate anaphor

kernel-based MD (TKMD), we compare its performance against two baselines. The lower bound, "all-NP", considers all the NP-nodes in a parse tree to be candidate mentions. The upper bound, "gold-NP" only considers gold NP-nodes to be candidate mentions. Note that the upper bound does not include mentions that do not correspond to NP-nodes at all (around 12% of all the mentions in the development data, cf. Table 1 above).

Tables 7 and 8 also show the performance level of BART's rule-based MD module that was developed for English. Although this heuristic has proved reliable on the English data, for example, at the CoNLL 2011 and 2012 shared tasks, it is not robust enough to be ported as-is to other languages: indeed, the performance of the heuristic MD on Arabic and Chinese is lower than the all-NP baseline. This highlights the importance of a learning-based approach: while rule-based MD shows good results for English, we cannot expect spending ten more years on designing similar systems for other languages.

	R	P	F
Arabic	90.67	31.07	46.28
Chinese	98.37	38.27	55.1

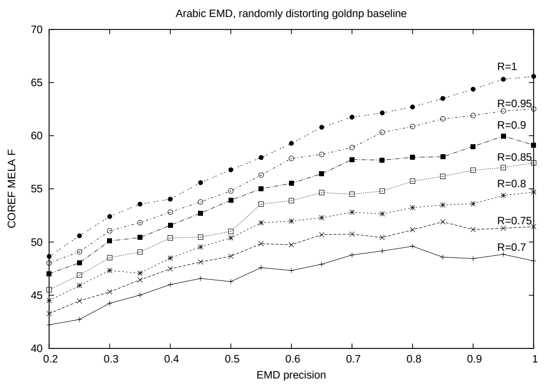
Table 6: Performance of the recall-oriented MD classifier on the CoNLL development set.

For both languages, the performance goes up drastically when one shifts from a realistic evaluation (the "all-NP" baseline) to gold NP mentions. Kernel-based MD is able to recover part of this difference, providing significant improvements over the baseline (t-test on individual documents, $p < 0.05$).

	Soon et al. (2001) features	Table 4 features
Arabic		
all-NP	46.15	46.32
English MD	43.46	43.49
TK-MD	48.13†	50.02†
gold-NP	63.27†	64.55†
Chinese		
all-NP	51.04	51.40
English MD	46.77	46.77
TK-MD	53.40†	53.86†
gold-NP	57.30†	57.98†

Table 7: Evaluating the impact of MD and linguistic knowledge: MELA F-score on the development set, significant improvement over the corresponding all-NP baseline shown with †.

Another important point is the difference between our basic feature set and more specific features (cf, Table 4). The contribution of extra features is relatively small and not significant, which is not surprising given the fact that all of them are very naïve and do not address any coreference-



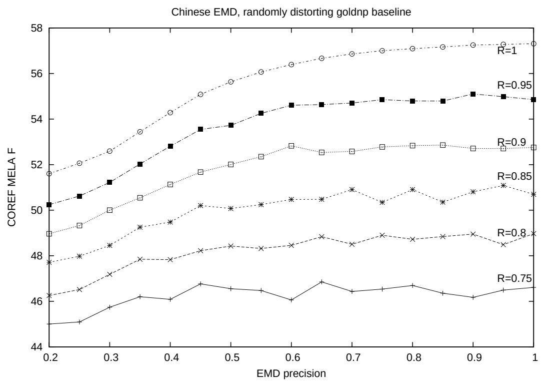


Figure 3: Performance of an end-to-end coreference resolution system for different values of MD Recall and Precision in a simulation experiment: MELA F-score on the Arabic and Chinese development data.

	Soon et al. (2001) features	Table 4 features
Arabic		
all-NP	46.79	47.36
English MD	43.77	43.65
TK-MD	48.38†	51.54†
gold-NP	63.07†	65.57†
Chinese		
all-NP	53.26	53.24
English MD	48.99	48.99
TK-MD	58.11†	58.15†
gold-NP	59.97†	60.04†

Table 8: Evaluating the impact of MD and linguistic knowledge: MELA F-score on the official CoNLL-2012 test set, significant improvement over the corresponding all-NP baseline shown with †.

related phenomena specific for Arabic and Chinese. However, the extra features help more when the MD improves. This suggests that a robust MD module is an essential prerequisite for further work on coreference in new languages: a more accurate set of mentions provides a better testbed for manually engineered language-specific features or constraints.

6 Conclusion and Future Work

In this paper we have investigated possibilities for language-independent mention detection based on syntactic tree kernels. We have shown that a kernel-based approach can provide a robust preprocessing system that is a vital prerequisite for fast and efficient development of end-to-end multilingual coreference resolvers.

We have evaluated different tree and vector representations, showing that the best performance is achieved by applying radial pruning to parse trees and augmenting the resulting representation with feature vectors, encoding very basic and shallow properties of candidate NPs.

We have investigated possibilities of incorporating our MD module to an end-to-end coreference resolution system. Our evaluation results show significant improvement over the system relying on the "all-NP" baseline for both Arabic and Chinese. It should be stressed that no other baseline is available without using deep linguistic expertise.

In the future, we plan to follow two directions to further improve our algorithm. First, we want to consider more global models of MD, providing joint inference over sets of NP nodes, and, possibly, incorporating CR predictions as well. Several studies (Daume III and Marcu, 2005; Denis and Baldridge, 2009) followed this direction recently, showing promising results for joint MD and CR modeling.

Second, we want to combine our learning-based MD with more traditional heuristic systems. While our approach provides a fast reliable testbed and allows CR researchers to specifically focus on coreference, rule-based MD modules have been created for a variety of languages, especially for European ones, in the past decade. We believe that by combining such systems with our kernel-based algorithm, we can build MD modules that show a high performance level and, at the same time, are more robust and portable to different domains and corpora.

Acknowledgments

The research described in this paper has been partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under the grant #288024: LIMOSINE – Linguistically Motivated Semantic aggregation engiNes.

References

Shane Bergsma and David Yarowsky. 2011. NADA: A robust system for non-referential pronoun detection. In *Proc. DAARC*, pages 12–23, Faro, Portugal, October.

Adriane Boyd, Whitney Gegg-Harrison, and Donna Byron. 2005. Identifying nonreferential *it*: A machine learning approach incorporating linguistically motivated patterns. In *Proceedingd of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing,*, pages 40–47.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Pascal Denis and Jason Baldridge. 2009. Global joint models for coreference resolution and named entity classification. In *Procesamiento del Lenguaje Natural 42, Barcelona: SEPLN*.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassell, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program–tasks, data, and evaluation. In *Proceedings of the Language Resources and Evaluation Conference*.

Richard Evans. 2001. Applying machine learning toward an automatic classification of *it*. *Literary and Linguistic Computing*, 16(1):45–57.

Radu Florian, Hany Hassan, Abraham Ittycheriah, Hongyan Jing, Xiaoqiang Luo, Nicolas Nicolov, and Salim Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1–8.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of HLT/NAACL*.

Hal Daume III and Daniel Marcu. 2005. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing*.

Abraham Ittycheriah, Lucian Vlad Lita, Nanda Kambhatla, Nicolas Nicolov, Salim Roukos, and Margo Stys. 2003. Identifying and tracking entity mentions in a maximum entropy framework. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press.

Lauri Karttunen. 1976. Discourse referents. In J. McKawley, editor, *Sytax and Semantics*, volume 7, pages 361–385. Academic Press.

Jonathan K Kummerfeld, Mohit Bansal, David Burkett, and Dan Klein. 2011. Mention detection: Heuristics for the OntoNotes annotations. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 102–106, Portland, Oregon, USA, June. Association for Computational Linguistics.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of European Conference on Machine Learning*, pages 318–329.

Alessandro Moschitti. 2008. Kernel methods, syntax and semantics for relational text categorization. In *Proceeding of the International Conference on Information and Knowledge Management*, NY, USA.

Vincent Ng. 2008. Unsupervised models for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 640–649.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Sixteenth Conference on Computational Natural Language Learning (CoNLL 2012)*, Jeju, Korea.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistic*, 27(4):521–544.

Yannick Versley, Simone Paolo Ponzetto, Massimo Poesio, Vladimir Eidelman, Alan Jern, Jason Smith, Xiaofeng Yang, and Alessandro Moschitti. 2008. BART: a modular toolkit for coreference resolution. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, pages 9–12.

Imed Zitouni and Radu Florian. 2008. Mention detection crossing the language barrier. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.