Undefined 0 (2012) 1–0
IOS Press

Structural Reranking Models for Named Entity Recognition

Truc-Vien T. NGUYEN [a] and Alessandro MOSCHITTI [b]
[a] *CIMeC - University of Trento*
Corso Bettini 31, 38068 Rovereto (TN), Italy
[b] *DISI - University of Trento*
Via Sommarive 5, 38123 Povo (TN), Italy

Abstract. We present a method for incorporating global features in named entity recognizers using reranking techniques and the combination of two state-of-the-art NER learning algorithms: conditional random fields (CRFs) and support vector machines (SVMs). The reranker employs two kinds of features: flat and structured features. The former are generated by a polynomial kernel encoding entity features whereas tree kernels are used to model dependencies amongst tagged candidate examples. The experiments on two standard corpora in two languages, i.e. the Italian EVALITA 2009 and the English CoNLL 2003 datasets, show a large improvement on CRFs in F-measure, i.e., from 80.34% to 84.33% and from 84.86% to 87.99%, respectively. Our analysis reveals that (i) both kernels provide a comparable improvement over the CRFs baseline; and (ii) their combination improves CRFs much more than the sum of the individual contributions, suggesting an interesting synergy.

Keywords: named entity recognition, reranking, kernel methods, conditional random fields

1. Introduction

Research in statistical natural language processing has shown that reranking is a promising approach for enhancing system accuracy. This method first employs a probabilistic model to generate a list of *top-N* candidates and then reranks such list with additional features (see for example, (Moschitti et al., 2006; Moschitti et al., 2008; Dinarelli et al., 2009; Nguyen et al., 2010; Johansson and Moschitti, 2010; Moschitti et al., 2012; Severyn and Moschitti, 2012). The approach is appealing as it can flexibly incorporate arbitrary features into a learning model. These features help in discriminating good from bad hypotheses and consequently can be used to select the best one. Various algorithms have been applied for reranking in NLP applications, including parsing (Collins, 2000; Collins and Duffy, 2002; Charniak and Johnson, 2005; Huang, 2008), name tagging (Collins, 2002; Collins and Duffy, 2002), machine translation (Shen et al., 2004) and opinion detection (Johansson and Moschitti, 2010). Such work has exploited the discriminative property as the main criterion of the reranking algorithm.

Recent research (Moschitti et al., 2012; Severyn and Moschitti, 2012) has shown that reranking models improve when coupled with kernel methods (Collins and Duffy, 2001; Moschitti, 2004; Moschitti, 2006). The main contribution of the latter is the possibility to extract a huge amount of features along with their dependencies from the ranking hypotheses. Indeed, while feature-based learning algorithms only involve the dot product between feature vectors, kernel methods allow for a higher generalization by replacing the dot product with a function between pairs of linguistic objects. Such functions are a kind of similarity measure satisfying certain properties. For example, tree kernels (Collins and Duffy, 2001) encode grammatical information in learning algorithms by computing the number *subtrees* shared by two syntactic trees (associated with the classification instances). Similarly, sequence kernels (Lodhi et al., 2002) count the number of common *subsequences* shared by two input strings.

The use of reranking kernels for NER is promising as previous algorithms, i.e., described in (Collins, 2002; Collins and Duffy, 2002), only targeted the entity detection (and not entity classification) task. Besides, kernel methods offer a natural way to exploit linguistic properties. Finally, the NER hypotheses can be generated from an entire sentence such that it can be possible to capture important global semantic features.

In this paper, we describe the use of kernel methods for performing effective reranking of NER hypotheses provided by basic systems for both Italian and English languages over well-known corpora. The key aspect of our method regards the use of structured and flat features for discriminating between candidate hypotheses. For this purpose, we apply tree kernels to a tree structure encoding sentence annotation, which, in turn, refers to a tagged sequences. We also combine such model with a polynomial kernel applied to vectors of innovative global features.

Our main contribution is to show that tree kernels can be used to define general and effective features modeling syntactic but also semantic information for NER reranking. Our study demonstrates that the composite kernel is very effective for reranking named-entity sequences. Without the need of producing and heuristically combining learning models like previous work on NER, our composite kernel not only captures most of the flat features but also efficiently exploits structured features. More interestingly, this kernel yields significant improvement when applied to two corpora of two different languages. The evaluation on the Italian corpus shows that our approach outperforms the best-reported methods whereas on the English data it reaches the state-of-the-art.

In the remainder of this paper, Section 2 reports on the related work, Section 3 introduces the tree kernel theory, Section 4 describes the datasets and the baseline performance associated with them, Section 5 describes our reranking models, Section 6 illustrates out experiments and finally Section 7 derives the conclusions.

2. Related Work

Named-entities (NEs) are objects that can be referred by names (Chinchor and Robinson, 1998), such as people, organizations, and locations. NEs are essential for defining the semantics of a document. The research on NER has been promoted by the Message Understanding Conferences (MUCs, 1987-1998), the shared task of the Conference on Natural Language Learning (CoNLL, 2002-2003), and the Automatic Content Extraction program (ACE, 2002-2005). Figure 1 shows a text from the CoNLL 2003 corpus, where all named entities are in bold.

The MUC and ACE programs were held in order to exchange research findings on techniques for extracting a variety of levels of information from official data. However, most early works were based on hand-crafted rules. The use of machine learning for NER was firstly proposed in (Bikel et al., 1997; Leek, 1997) and consisted in a simple Hidden Markov Model (HMM). Since then, various IE learning algorithms have been developed. In the last decade, most effort has been focused on developing machine learning algorithms to reduce effort and time required by domain experts for engineering rules.

Existing approaches for NER using machine learning fall into two types. Simpler NER systems employing one learning algorithm only. These include, e.g., Maximum Entropy (Bender et al., 2003; Chieu and Ng, 2003; Curran and Clark, 2003), Hidden Markov Models (Zhou and Su, 2002), Perceptron (Carreras et al., 2003a), Adaboost (Carreras et al., 2003b), Conditional Random Fields (McCallum and Li, 2003) and Support Vector Machines (Mayfield et al., 2003).

The second type uses multiple learning algorithms used in classifier committees. For example, (Florian et al., 2003) present a classifier combination for NER in which four diverse models are combined under different conditions.

2.1. Reranking in Statistical Natural Language Processing

As mentioned in the introduction, discriminative rerankers have been used for many NLP applications, including parsing (Collins, 2000; Collins and Duffy, 2002; Charniak and Johnson, 2005; Huang, 2008), name tagging (Collins, 2002; Collins and Duffy, 2002), machine translation (Shen et al., 2004) and opinion detection (Johansson and Moschitti, 2010; Johansson and Moschitti, 2013). Their algorithm is very simple: first a basic classifier is used to generate a set of automatic annotation hypotheses of limited size, and then a second classifier, i.e., the reranker, is used to select the best hypothesis. The reranker is a simple binary classifier of hypothesis pairs but can encode complex dependencies, e.g., thanks to structural kernels (e.g., see (Moschitti, 2008)). In particular, tree, sequence and linear kernels can be applied to struc-

Owen Finegan has recovered from the knocks he took in last weekend's test against **Wales** and retains his place in the back-row ahead of **Daniel Manu**. The **Wallabies** have their sights set on a 13th successive victory to end their **European** tour with a 100 percent record but also want to turn on the style and provide **David Campese** with a fitting send-off in his final match in **Australian** colours.

Fig. 1. CoNLL text with all entities in bold

tural and feature-vector representations of the hypothesis pairs to train the reranker. For example, we have used them for reranking the output of (i) a multi-label hierarchical text classifiers (Moschitti et al., 2012); (ii) a question answering system (Severyn and Moschitti, 2012); (iii) a concept labeling and segmentation model for speech (Dinarelli et al., 2009); and (iv) a semantic role labeling system (Moschitti et al., 2006; Moschitti et al., 2008).

A typical approach to discriminative reranking using kernel methods is the preference reranker. The task concerns the selection of the correct hypothesis candidate H_i from a candidate set $\mathscr{H} = \{H_1, H_2, \ldots, H_k\}$. This task is reduced to a binary classification problem by training a model on hypothesis *pairs*. The member order of the latter determines the positive or negative label of the data. For example, if H_1 is better than H_2, we can choose to have the following order $\langle H_1, H_2 \rangle$, consequently, $\langle H_2, H_1 \rangle$ will be a negative example. This training set can then be used to train a binary classifier. At classification time, pairs are not formed (since the correct candidate is not known); instead, the standard one-versus-all binarization method is applied.

The kernels are then engineered to implicitly represent the *differences* between the objects in the pairs. If we have a valid kernel K over $\mathscr{H}$, we can construct a preference kernel $P_K : \mathscr{H} \times \mathscr{H} \rightarrow \Re$ as follows:

$$P_K(\langle H_1, H_2 \rangle, \langle H_3, H_4 \rangle) = K(H_1, H_3) + K(H_2, H_4) - K(H_1, H_4) - K(H_2, H_3) \quad (1)$$

It is easy to show that P_K is also a valid Mercer's kernel (Shawe-Taylor and Cristianini, 2004). This makes it possible to use kernel methods to train the reranker. The next section describes tree kernels in detail to demonstrate how they can generate effective features to represent trees. The latter are used to represent NER hypotheses in our reranker.

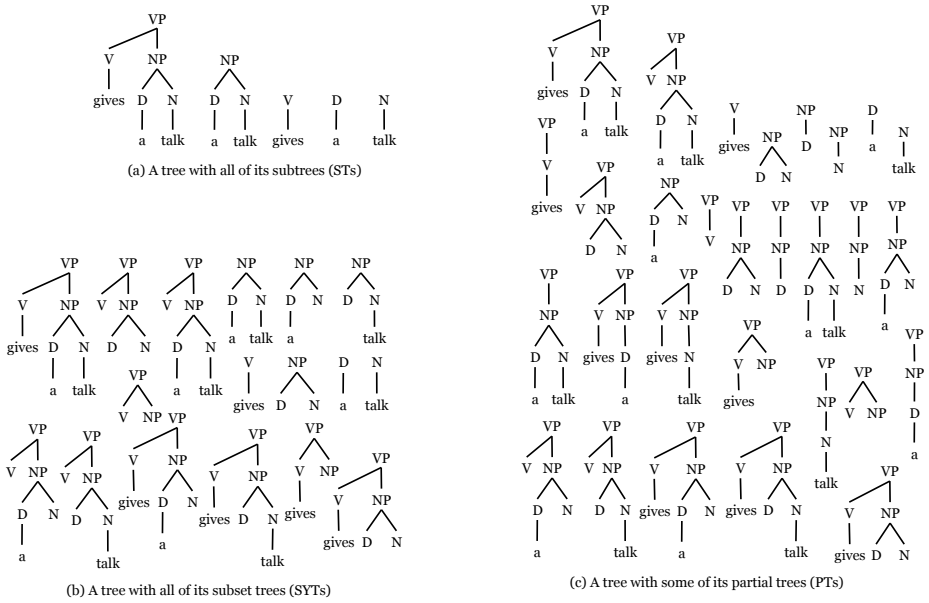
3. Tree Kernels

Tree kernels represent trees in terms of their substructures (called tree fragments). Such fragments form a feature space, which, in turn, is mapped into a vector space. Tree kernels measure the similarity between pair of trees by counting the number of fragments in common. There are three important characterizations of fragment type: the SubTrees (STs), the Syntactic Trees (SYTs) and the Partial Trees (PTs).

3.1. Tree Fragment Types

An ST is defined by taking any node along with its descendants. An SYT is a more general structure, which does not necessarily include all the descendants. One strong constraint is that an SYT (and also an ST) must be generated by applying the same grammatical rule set that generated the original tree, as pointed out in (Collins and Duffy, 2001). The rules impose grammatical constraints on the children of a node, which cannot be separated (otherwise the grammar rule generating an SYT with split children may be constituted by a new rule and thus would be different from the initial set of rules). A Partial Tree (PT) is a more general form of substructure obtained by relaxing constraints over the SYTs, i.e., children can be separated to generate PTs. Figure 2 shows the overall fragment set of the ST, SYT and PT kernels for the syntactic parse tree of the sentence fragment: *gives a talk*. In general, the tree kernel impact depends on the specific application, which should suggest the appropriate kernel type.

To carry out efficient computation of this high dimensional representation, we follow the algorithm introduced in (Collins and Duffy, 2001). We enumerate all tree fragments that occur in the training data from 1 to n. Thus, each tree is represented by an n-dimensional vector where the i'th component counts the number of occurrences of the i'th tree fragment. Let us define the function $c_i(T)$ to be the number of



(a) A tree with all of its subtrees (STs)

(b) A tree with all of its subset trees (SYTs)

(c) A tree with some of its partial trees (PTs)

Fig. 2. Three kinds of tree fragment sets: ST, SYT (also known as SST) and PT: they all originate from the first tree of each subfigure, i.e., the tree associated with "gives a talk" (which also corresponds to the largest fragment).

occurrences of the i'th tree fragment in tree T, so that T is now represented as $\vec{c}(T) = (c_1(T), c_2(T), ..., c_n(T))$.

We then examine the inner product between two trees T_1 and T_2 under this representation, i.e., $K(T_1, T_2) = \vec{c}(T_1) \cdot \vec{c}(T_2)$. To compute K, we define (i) the set of nodes in trees T_1 and T_2 as N_1 and N_2 respectively; and (ii) the indicator function $I_i(n)$ to be 1 if subtree is seen rooted at node n and 0 otherwise. It follows that $c_i(T_1) = \sum_{n_1 \in N_1} I_i(n_1)$ and $c_i(T_2) = \sum_{n_2 \in N_2} I_i(n_2)$. With some simple algebra we have:

$$\vec{c}(T_1) \cdot \vec{c}(T_2) = \sum_i c_i(T_1) c_i(T_2) = \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1) I_i(n_2) \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \Delta(n_1, n_2)$$

where we define $\Delta(n_1, n_2) = \sum_i I_i(n_1) I_i(n_2)$. Note that $\Delta(n_1, n_2)$ can be computed in polynomial time and determines the kernel type.

3.2. Syntactic Tree Kernel

To compute STK, we need to define $\Delta(n_1, n_2)$ as follows:

- If the productions at n_1 and n_2 are different $\Delta(n_1, n_2) = 0$.
- If the productions at n_1 and n_2 are the same, and n_1 and n_2 are pre-terminals, then $\Delta(n_1, n_2) = 1$.
- Else if the productions at n_1 and n_2 are the same and n_1 and n_2 are not pre-terminals,

$$\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + \Delta(ch(n_1, j), ch(n_2, j))), \quad (2)$$

where $nc(n_1)$ is the number of children of n_1 in the tree ($nc(n_1) = nc(n_2)$ since the productions at n_1/n_2 are the same) and $ch(n_1, j)$ is the j'th child of n_1.

Note that $\Delta(n_1, n_2)$ counts the number of *common subtrees* rooted at both n_1 and n_2. From the identity $\vec{c}(T_1) \cdot \vec{c}(T_2) = \sum_{n_1, n_2} \Delta(n_1, n_2)$, and the recursive definition of $\Delta(n_1, n_2)$, it follows that STK can be calculated in $O(|N_1||N_2|)$ time.

3.3. Partial Tree Kernel (PTK)

The computation of PTFs is carried out by the Δ function defined as follows:

1. if the node labels of n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;
2. else:

$$\Delta(n_1, n_2) = 1 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1) = l(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta(ch(n_1, \vec{I}_{1j}), ch(n_2, \vec{I}_{2j}))$$

where $\vec{I}_1 = \langle k_1, k_2, k_3, .. \rangle$ and $\vec{I}_2 = \langle k'_1, k'_2, k'_3, .. \rangle$ are index sequences associated with the ordered child sequences $ch(n_1, \cdot)$ of n_1 and $ch(n_2, \cdot)$ of n_2, respectively. $\vec{I}_{1j}$ and $\vec{I}_{2j}$ point to the j-th child in the corresponding sequence, and $l(\cdot)$ returns the sequence length, i.e., the number of children in the sequence.

Furthermore, we add two decay factors: μ for the depth of the tree and λ for the length of the child subsequences with respect to the original sequence, which accounts for gaps. Hence, the PTK expression for $\Delta(n_1, n_2) =$

$$\mu\Big(\lambda^2 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1)=l(\vec{I}_2)} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta(ch(n_1, \vec{I}_{1j}), ch(n_2, \vec{I}_{2j}))\Big) \quad (3)$$

where $d(\vec{I}_1) = \vec{I}_{1l(\vec{I}_1)} - \vec{I}_{11} + 1$ and $d(\vec{I}_2) = \vec{I}_{2l(\vec{I}_2)} - \vec{I}_{21} + 1$. This way, we penalize both larger trees and child subsequences with gaps. Eq. 3 is more general than Eq. 2. Indeed, if we only consider shared subsequences that contain all the node children (no child of a node is skipped), we actually obtain the STK kernel. The computational complexity of PTK is $O(p\rho^2 |N_{T_1}| \times |N_{T_2}|)$ (Moschitti, 2006), where p is the largest subsequent of children that we want to consider and ρ is the maximal outdegree observed in the two trees. However, as shown in (Moschitti, 2006), the average running time again tends to be linear in the number of nodes for natural language syntactic trees.

3.4. The Polynomial Kernel

The polynomial kernel between two candidate tagged sequences is defined as:

$$K(x, y) = (1 + \vec{x}_1 \cdot \vec{x}_2)^2,$$

where $\vec{x}_1$ and $\vec{x}_2$ are two feature vectors extracted from the two sequences with the global feature template.

3.5. Kernel Engineering

Kernel engineering can be carried out by combining basic kernels with additive or multiplicative operators or by designing specific data objects (vectors, sequences and tree structures) for the target tasks.

It is worth noting that well-known kernels applied to new structures produce completely new kernels as shown hereafter. Let $\mathrm{K}(t_1, t_2) = \phi(t_1) \cdot \phi(t_2)$ be a basic kernel, where t_1 and t_2 are two trees. If we map t_1 and t_2 into two new structures s_1 and s_2 with a mapping $\phi_M(\cdot)$, we obtain: $\mathrm{K}(s_1, s_2) = \phi(s_1) \cdot \phi(s_2) = \phi(\phi_M(t_1)) \cdot \phi(\phi_M(t_2)) = \phi'(t_1) \cdot \phi'(t_2) = \mathrm{K}'(t_1, t_2)$, which is a noticeably different kernel induced by the mapping $\phi' = \phi \circ \phi_M$.

3.6. The role of tree kernels in Reranking

Eq. 1 allow us to use any kernel K for defining the preference reranker. Therefore, we can use tree kernels, such as PTK or STK, to describe the similarity between two hypotheses. In Section 5.2, we will describe a tree representation for hypotheses, which can be used as input of tree kernels. The main idea is that tree kernels can measure the similarity between such hypotheses in terms of their subtrees. Such similarity can learn the order of hypotheses according to their accuracy. For example, Eq. 1 imposes that if H_1 is similar to H_3 and H_2 is similar to H_4, then H_1 more accurate than H_2 will suggest that H_3 is more accurate than H_4.

In the next section we present the first step of reranking, i.e., the basic NER used for building the hypotheses along with the baseline results.

4. Datasets and Baseline for Named Entity Recognition

A robust NER system is expected to be adaptable to different domains and languages. Therefore, we experimented with two datasets: i) the well-known CoNLL 2003 English shared task corpus; and ii) the EVALITA 2009 Italian corpus. Statistics about such datasets are shown in tables 1 and 2. In the following sections we describe the datasets and the typical accuracy that baseline NERs attain on them.

4.1. Datasets

The CoNLL English dataset. The CoNLL 2003 English dataset is created within the shared task of CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003). It is a collection of news wire articles from the Reuters Corpus, annotated with four entity types: Person (PER), Location (LOC), Organization (ORG) and Miscellaneous name (MISC). The training and the development datasets are news feeds from August 1996, while the test set contains news feeds from December 1996. Accordingly, the named entities in the test dataset are considerably different from those that appear in the training or the development set.

CoNLL	LOC	MISC	ORG	PER
Train	7140	3438	6321	6600
	30.38%	14.63%	26.90%	28.09%
Dev	1837	922	1341	1842
	30.92%	15.52%	22.57%	31.00%
Test	1668	702	1661	1617
	29.53%	12.43%	29.41%	28.63%

Table 1
Statistics on the CoNLL English dataset

EVALITA	GPE	LOC	ORG	PER
Train	2813	362	3658	4577
	24.65%	3.17%	32.06%	40.11%
Test	1143	156	1289	2378
	29.53%	12.43%	29.41%	28.63%

Table 2
Statistics on the EVALITA Italian dataset

The EVALITA Italian dataset. The EVALITA 2009 Italian dataset is based on I-CAB, the Italian Content Annotation Bank (Magnini et al., 2006), annotated with four entity types: Person (PER), Organization (ORG), Geo-Political Entity (GPE) and Location (LOC). The training data, taken from the local newspaper "L'Adige", consists of 525 news stories, which belong to five categories: News Stories, Cultural News, Economic News, Sports News and Local News. Test data, on the other hand, consists of completely new texts, taken from the same newspaper and consists of 180 news stories.

4.2. The baseline algorithm

We selected Conditional Random Fields (CRFs) (Lafferty et al., 2001) as the baseline model. These are a probabilistic framework for labeling and segmenting sequence data. They present several advantages over other purely generative models such as Hidden Markov models (HMMs) by relaxing the independence assumptions required by HMMs. Besides, HMMs and other discriminative Markov models are prone to the label bias problem, which is effectively solved by CRFs.

The named-entity recognition (NER) task is framed as assigning label sequences to a set of observation sequences. We follow the IOB notation where the NE tags have the format B-TYPE, I-TYPE or O, which mean that the word is a beginning, a continuation, or not part of an entity at all. For example, the following sentence is labeled with the tags above:

Il/O **presidente**/O **della**/O **Fifa**/B-ORG **Sepp**/B-PER **Blatter**/I-PER **affermando**/O **che**/O **il**/O **torneo**/O **era**/O **stato**/O **ottimo**/O (FIFA president Sepp Blatter says that the tournament was excellent)

For our experiments, we used CRF++ [1] to build our recognizer, which is a model trained discriminatively with the unigram and bigram features. These are extracted from a window of k words centered in the target word w (i.e. the one we want to classify with the B, O, I tags). The features used for each token are described below.

Basic features. The basic features are small, primitive units, consisting of the word itself with some features directly derived from it:

1. w_i for $i = 1 \ldots n$ is the i'th word
2. l_i is the word in lower-case
3. $p1_i$, $p2_i$, $p3_i$, $p4_i$ are four prefixes of w_i of size 1, 2, 3 or 4.
4. $s1_i$, $s2_i$, $s3_i$, $s4_i$ are four suffixes of w_i as before.
5. f_i is the part-of-speech of w_i
6. g_i is the orthographic feature that tests whether a word contains *all upper case, initial upper case, all lower-case letters*.

[1] http://crfpp.sourceforge.net

Category	Pr	Re	F_1
All	85.37	84.35	84.86
LOC	90.25	88.61	89.42
MISC	79.81	74.51	77.07
ORG	80.02	77.85	78.92
PER	87.94	90.92	89.41

Table 3
CRFs results on the CoNLL test set

Category	Pr	Re	F_1
All	84.76	84.18	84.47
LOC	87.99	88.6	88.29
MISC	79.22	75.76	77.45
ORG	80.96	76.81	78.83
PER	87.3	90.85	89.04

Table 4
SVMs results on the CoNLL test set

7. k_i is a feature that tests whether a token is a word, a number, a symbol, a punctuation mark.
8. o_i is the gazetteer feature. We simply look up w_i in our knowledge base.

Combined features. A bi-gram feature is derived from the combination of two consecutive units. To allow the highest generalization, the consecutive units can be lexical words or any of their basic features as described in the previous section. For example, the text "presidential palace" can have one feature as "presidential/palace" and another feature as "JJ/NN" where JJ and NN are the part-of-speech of presidential and palace.

1. The combinations w_{i-2}/w_{i-1}, w_{i-1}/w_i, and w_i/w_{i+1}
2. Similar to the first, but use l_i instead of w_i
3. Similar to the first, but use $p1_i$, $p2_i$, $p3_i$, $p4_i$ instead of w_i
4. Similar to the first, but use $s1_i$, $s2_i$, $s3_i$, $s4_i$ instead of w_i
5. Similar to the first, but use f_i instead of w_i
6. Similar to the first, but use o_i instead of w_i
7. The combinations w_{i-2}/f_{i-2}, w_{i-1}/f_{i-1}, w_i/f_i, and w_{i+1}/f_{i+1}
8. The combinations f_{i-2}/o_{i-2}, f_{i-1}/o_{i-1}, f_i/o_i, and f_{i+1}/o_{i+1}
9. The combinations w_{i-2}/o_{i-2}, w_{i-1}/o_{i-1}, w_i/o_i, and w_{i+1}/o_{i+1}

The gazetteer lists are built with names imported from different sources. For English, the geographic features are imported from NIMA's GEOnet Names Server (GNS)[2] and The Alexandria Digital Library (ADL) gazetteer[3]. The company data is included with all the publicly traded companies listed in Google directory[4] and the European business directory[5]. For Italian, the generic proper nouns are extracted from Wikipedia and various Italian sites. Moreover, the gazetteer lists for Italian are extracted from La Repubblica (Baroni et al., 2004), a large corpus of Italian newspaper text by using rule-based approach with patterns tuned specifically for each NE class.

4.3. Baseline Results

We trained the NER classifier on the two datasets. The Italian system participated in the EVALITA 2009 NER task (Nguyen et al., 2009). In addition to the base CRF classifier we trained another classifier with SVMs. Although this performed worse than the base model, we reported its results for completeness.

Table 5 and 6 show the final performance on the Italian test set with CRFs and SVMs. We found that,

[2]http://www.nima.mil/gns/html
[3]http://www.alexandria.ucsb.edu
[4]http://directory.google.com/Top/Business
[5]http://www.europages.net

Category	Pr	Re	F_1
All	83.43	77.48	80.34
GPE	83.83	84.6	84.22
LOC	76.99	45.74	57.38
ORG	72.74	60.42	66.01
PER	90.6	89.14	89.86

Table 5
CRFs results on the EVALITA test set

Category	Pr	Re	F_1
All	82.84	77.8	80.24
GPE	82.72	85.07	83.88
LOC	77.67	48.52	59.73
ORG	71.66	61.56	66.23
PER	90.92	88.51	89.7

Table 6
SVMs results on the EVALITA test set

with the same set of features, the accuracy of the NE classifiers trained with two models is rather competitive. Moreover, the NE classes, GPE and PER, achieve a rather good F1, while the recognition of ORG and LOC seems more problematic. This is in line with previous work according to which ORG seems to be the most difficult category to be learned. The lack of resources (the gazetteer for LOC is the smallest) may be another important reason for such low accuracy.

5. Reranking Models for Named Entity Recognition

In this section we show our most important contribution, i.e., the definition of a reranker for the outcome of a NER applied to an entire sentence. One important characteristic of such model is the use of structural kernels (i.e., tree kernels) to learn the reranking classifier. We also combine them with a polynomial kernel applied to traditional and innovative NER features.

5.1. Reranking Strategy

We first train a CRFs model to generate *10*-best candidates per sentence, along with their probabilities. Each candidate is then represented by a semantic tree together with a feature vector. We set our reranking task as a binary classification problem where examples are pairs of hypotheses $< H_i, H_j >$. As an example, let us consider the sentence:

> "**South African Breweries Ltd bought stakes in the Lech and Tychy brewers**"

tagged by CRFs in three different ways:

H_i : *B-ORG I-ORG I-ORG I-ORG O O O O B-ORG O B-ORG O*
H_j : *B-MISC I-MISC B-ORG I-ORG O O O O B-ORG I-ORG I-ORG O*
H_k : *B-ORG I-ORG I-ORG I-ORG O O O O B-ORG O B-LOC O*,

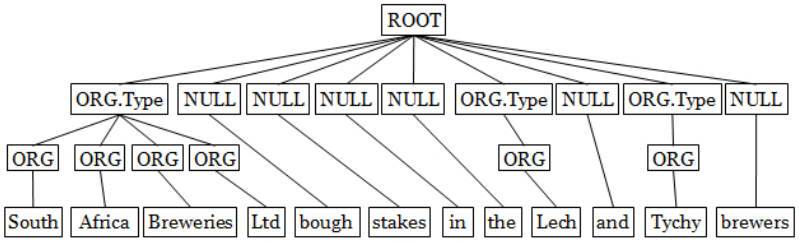
where the first candidate is the correct sequence whereas B-ORG, I-ORG, B-LOC, O are the NE tags generated according to the IOB notation (as described in Section 4.2). Using such tagged sequences, we build the following pairs of hypotheses:

$+1 < H_i, H_j >$, $+1 < H_i, H_k >$, $-1 < H_j, H_i >$, and $-1 < H_k, H_i >$,

where we assign the positive label to the first two pairs and negative one to the others as we assume that positive instances must have the correct hypothesis H_i, in the first position. Then a binary classifier based on SVMs and kernel methods can be trained on such pairs to discriminate between the best hypothesis, i.e. H_i, and the others. At testing time the hypothesis receiving the highest score is selected (Collins and Duffy, 2001).

	South	African	Breweries	Ltd	bought	stakes	in	the	unlisted	Lech	and	Tychy	brewers
H_i	B-ORG	I-ORG	I-ORG	I-ORG	O	O	O	O	O	B-ORG	O	B-ORG	O
H_j	B-MISC	I-MISC	B-ORG	I-ORG	O	O	O	O	O	B-ORG	I-ORG	I-ORG	O
H_k	B-ORG	I-ORG	I-ORG	I-ORG	O	O	O	O	O	B-ORG	O	B-LOC	O

(a) Candidate tagged sequences



(b) Semantic tree of the first sequence

Fig. 3. Semantic structure of a candidate sequence

5.2. *Representation of Tagged Sequences in Semantic Trees*

We provide a representation for hypotheses based on structures and global features. As in the case of NER, an input candidate is a sequence of word/tag pairs $x = \{w_1/t_1 ... w_n/t_n\}$, where w_i is the i'th word and t_i is the i'th NE tag for that word. The first representation we consider is a tree structure. For example, Figure 3 shows the semantic tree for H_i. In such tree representation: (a) the words are the leaf nodes; (b) their BIO tags are the father nodes and (c) the NER category is the father of BIO tags. A final fake root node connects all the NEs of the sentence. If a word is not an NE, it is simply connected to the *null* node.

This structure allows for best exploiting tree kernels between competing candidates. Indeed, in such kernel space, the inner product counts the number of common subtrees, i.e., the kernel computed on pairs of sentences containing similar sequences of NE tags are likely to have higher score. For example, the similarity between H_i and H_k will be higher than the similarity between H_j and H_k. This is reasonable since H_i and H_k have the highest F_1.

It is worth noting that another useful modification is the flexibility of incorporating diverse and arbitrary features into this tree structure by adding them as children of the entity tag nodes. These features can be exploited efficiently with PTK, which relaxes the constraints of production rules.

5.3. *Global features*

In addition to the structural representation, we use traditional feature vectors. Their advantage is that they can specify some global and unstructured information about the NEs of the entire sentence. In particular, we use novel features described hereafter.

Mixed n-grams features. In previous work, some global features have been used, e.g., (Collins, 2002; Collins and Duffy, 2002) but the employed algorithm just exploited arbitrary information regarding word types and linguistic patterns. In contrast, we define and study diverse features by also considering the *n-grams* patterns preceding and following the target entity.

Complementary context. In supervised learning, NER systems often suffer from low recall, which is caused by lack of both resource and context. For example, a word like "Arkansas" may not appear in the training set and, in the test set, there may not be enough context to infer its NE tag. In such cases, neither global features defined in (Chieu and Ng, 2002) nor aggregated contexts (Chieu and Ng, 2003) can help.

To overcome this deficiency, we employed the following unsupervised procedure: first, the baseline NER is applied to a target unlabeled corpus. Second, we associate each word of the corpus with the most frequent NE category assigned in the previous step. Finally, the above tags are used as features during the training of the improved NER and also for building the feature representation for a new classification instance. This way, for any unknown (i.e., never seen in the training set) word w of the test set, we can rely on

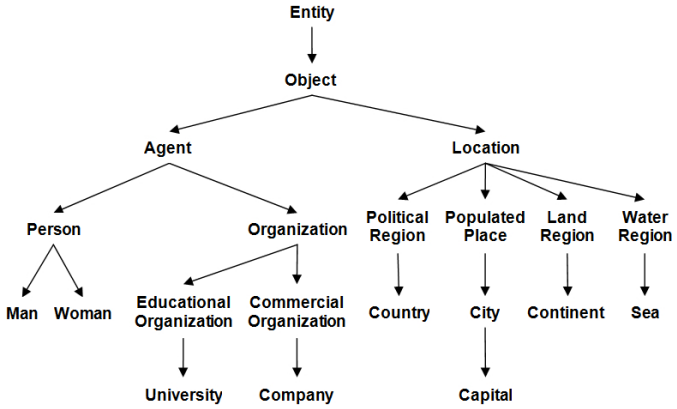


Fig. 4. An excerpt from the ontology

the most probable NE category as feature. That is, we use the most frequent category assigned to w by a basic NER over a reference corpus. The advantage is that we derived it by using the average over many possible contexts of w, which are in the different instances of the unlabeled corpus.

The unlabeled corpus for Italian was collected from La Repubblica [6] and it contains over 20 millions words, whereas the one for English was collected mainly from The New York Times [7] and BBC news stories [8] with more than 35 millions words.

Head word. As the head word of an entity plays an important role in information extraction (Surdeanu et al., 2003; Bunescu and Mooney, 2005), it is included in the global set together with its orthographic representation. More in detail, our global features are the following:

1. w_i for $i = 1 \ldots n$ is the i'th word
2. t_i is the NE tag of w_i
3. g_i is the gazetteer feature of the word w_i
4. f_i is the most frequent NE tag seen in a large corpus of w_i
5. h_i is the head word of the entity. We normally set the head word of an entity as its last word. However, when a preposition exists in the entity string, its head word is set as the last word before the preposition. For example, the head word of the entity "University of Pennsylvania" is "University".
6. Mixed *n-grams* features of the words and their gazetteer/frequent tag before/after the start/end of an entity. In addition to the normal *n-grams* solely based on words, we mix words with gazetteer/frequent tag seen from a large corpus and create mixed *n-grams* features.

Table 7 shows the full set of global features in our reranking framework. These features are anchored for each entity instance and adapted to entity categories. For example, the entity string (first feature) of the entity "United Nations" with entity type "ORG" is "ORG United Nations". This helps to discriminate different entities with the same surface forms. Moreover, they can be combined with *n-gram* patterns to learn and explicitly push the score of the correct sequence above the score of competing sequences.

An example of feature extraction. Given the sentence *"Eric Furda, dean of admissions at the University of Pennsylvania, is taking reader questions."* with the target entity *"University of Pennsylvania"*, Table 8 describes each feature with an example in context using the following primitives:

- $w_s\_w_{s+1}\_\ldots\_w_e$ = University of Pennsylvania (the entity string)
- g_{s-1} = det (the gazetteer feature of the word *the* is a determiner)
- g_s = ORG_PREFIX (the gazetteer feature of the word *University* is an organization prefix)
- g_{s+1} = prep (the gazetteer feature of the word *of* is a preposition)
- f_{s-1} = O (the most frequent NE tag of the word *the* is "O", i.e., the *null* tag)
- f_s = ORG (the most frequent NE tag of the word *University* is an organization)

[6] http://www.repubblica.it/
[7] http://www.nytimes.com/
[8] http://news.bbc.co.uk/

F_i	Notation	Description
f_1	w_s-w_{s+1}-$\cdots$-w_e	Entity string, from s (start) to e (end)
f_2	g_s-g_{s+1}-$\cdots$-g_e	The gazetteer features of words of the entity
f_3	f_s-f_{s+1}-$\cdots$-f_e	The most frequent NE tag feature (seen from a large corpus) of words of the entity
f_4	*hw*	The head word of the entity (see Section 5.3)
f_5	*lhw*	Indicates whether the head word is lowercased
f_6	w_{s-1}-w_s; w_{s-1}-g_s; g_{s-1}-w_s; g_{s-1}-g_s	Mixed bigrams of the word/gazetteer feature associated with the word before the start, followed by the word/gazetteer feature of the start of the entity
f_7	w_e-w_{e+1}; w_e-g_{e+1}; g_e-w_{e+1}; g_e-g_{e+1}	Mixed bigrams of the word/gazetteer feature associated with the end of the entity, followed by the next word/gazetteer feature
f_8	w_{s-1}-w_s; w_{s-1}-f_s; f_{s-1}-w_s; f_{s-1}-f_s	Similar to f_6, but using the frequent-tag instead of the gazetteer feature
f_9	w_e-w_{e+1}; w_e-f_{e+1}; f_e-w_{e+1}; f_e-f_{e+1}	Similar to f_7, but using the frequent-tag instead of the gazetteer feature
f_{10}	w_{s-2}-w_{s-1}-w_s; w_{s-1}-w_s-w_{s+1}; w_{e-1}-w_e-w_{e+1}; w_{e-2}-w_{e-1}-w_e	Trigram features of the words before/after the start/end of the entity
f_{11}	w_{s-2}-w_{s-1}-g_s; w_{s-2}-g_{s-1}-w_s; w_{s-2}-g_{s-1}-g_s; g_{s-2}-w_{s-1}-w_s; g_{s-2}-w_{s-1}-g_s; g_{s-2}-g_{s-1}-w_s; g_{s-2}-g_{s-1}-g_s; w_{s-1}-w_s-g_{s+1}; w_{s-1}-g_s-w_{s+1}; w_{s-1}-g_s-g_{s+1}; g_{s-1}-w_s-w_{s+1}; g_{s-1}-w_s-g_{s+1}; g_{s-1}-g_s-w_{s+1}; g_{s-1}-g_s-g_{s+1}	Mixed trigrams of the words/gazetteer features before the start of the entity
f_{12}	w_{e-1}-w_e-g_{e+1}; w_{e-1}-g_e-w_{e+1}; w_{e-1}-g_e-g_{e+1}; g_{e-1}-w_e-w_{e+1}; g_{e-1}-w_e-g_{e+1}; g_{e-1}-g_e-w_{e+1}; g_{e-1}-g_e-g_{e+1}; w_{e-2}-w_{e-1}-g_e; w_{e-2}-g_{e-1}-w_e; w_{e-2}-g_{e-1}-g_e; g_{e-2}-w_{e-1}-w_e; g_{e-2}-w_{e-1}-g_e; g_{e-2}-g_{e-1}-w_e; g_{e-2}-g_{e-1}-g_e	Mixed trigrams of the words/gazetteer features after the end of the entity
f_{13}	w_{s-2}-w_{s-1}-f_s; w_{s-2}-f_{s-1}-w_s; w_{s-2}-f_{s-1}-f_s; f_{s-2}-w_{s-1}-w_s; f_{s-2}-w_{s-1}-f_s; f_{s-2}-f_{s-1}-w_s; f_{s-2}-f_{s-1}-f_s; w_{s-1}-w_s-f_{s+1}; w_{s-1}-f_s-w_{s+1}; w_{s-1}-f_s-f_{s+1}; f_{s-1}-w_s-w_{s+1}; f_{s-1}-w_s-f_{s+1}; f_{s-1}-f_s-w_{s+1}; f_{s-1}-f_s-f_{s+1}	Mixed trigrams of the words/frequent-tag features before the start of the entity
f_{14}	w_{e-1}-w_e-f_{e+1}; w_{e-1}-f_e-w_{e+1}; w_{e-1}-f_e-f_{e+1}; f_{e-1}-w_e-w_{e+1}; f_{e-1}-w_e-f_{e+1}; f_{e-1}-f_e-w_{e+1}; f_{e-1}-f_e-f_{e+1}; w_{e-2}-w_{e-1}-f_e; w_{e-2}-f_{e-1}-w_e; w_{e-2}-f_{e-1}-f_e; f_{e-2}-w_{e-1}-w_e; f_{e-2}-w_{e-1}-f_e; f_{e-2}-f_{e-1}-w_e; f_{e-2}-f_{e-1}-f_e	Mixed trigrams of the words/frequent-tag features after the end of the entity

Table 7
Global features with the polynomial kernel for reranking

– f_{s+1} = O (the most frequent NE tag of the word *of*)

We point out that we used the ontology and knowledge base from the KIM ontology (Kiryakov et al., 2004) as our gazetteer. The KIM proton ontology contains about 300 classes, 100 attributes and relations. KIM World Knowledge Base (KB) contains about 77,500 entities with more than 110,000 aliases. Figure 4 shows an excerpt from the KIM ontology: we take the deepest class subsuming a leaf that matches the target piece of text (multiwords) as tags. For example, if the text "New York" matches with LOCATION, STATE, CITY then CITY will be chosen since it is the deepest class in the ontology. If a text matches with many classes in different branches, then a more general class will be chosen. For example, if the text "Washington" matches with PERSON and CITY, which lie in two different branches in the ontology, then we choose

F_i	Example: "Eric Furda, dean of admissions at the University of Pennsylvania, is taking reader questions"
f_1	University of Pennsylvania
f_2	ORG_PREFIX prep STATE
f_3	ORG O ORG
f_4	University
f_5	yes
f_6	"the University"; "the ORG_PREFIX"; "det University"; "det ORG_PREFIX"
f_7	"Pennsylvania ,"; "Pennsylvania ,"; "STATE ,"; "STATE ,"
f_8	"the University"; "the ORG"; "O University"; "O ORG"
f_9	"Pennsylvania ,"; "Pennsylvania O"; "ORG ,"; "ORG O"
f_{10}	"at the University"; "the University of"; "of Pennsylvania ,"; "University of Pennsylvania"
f_{11}	"at the ORG_PREFIX"; "at det University"; "at det ORG_PREFIX"; "prep the University"
f_{12}	"of Pennsylvania ,"; "of STATE ,"; "of STATE ,"; "prep Pennsylvania ,"
f_{13}	"at the ORG"; "at O University"; "at O ORG"; "O the University"
f_{14}	"of Pennsylvania ,"; "of ORG ,"; "of ORG O"; "O Pennsylvania ,"

Table 8
Examples of global features in real context

the class ENTITY as the parent class for both PERSON and CITY.

5.4. Reranking with Composite Kernel

In this section we describe our novel tagging kernels based on diverse global features as well as semantic trees for reranking candidate tagged sequences. As mentioned in the previous section, we can engineer kernels by combining trees and entity kernels.

5.5. The Tagging Kernels

In our reranking framework, we incorporate the probability from the original model with the tree structure as well as the feature vectors. Let us consider the following notations:

- $K(x,y) = L(x) \cdot L(y)$ is the basic kernel where $L(x)$ is the log probability of a candidate tagged sequence x under the BIO-tag probability model.
- $TK(x,y) = t(x) \cdot t(y)$ is the partial tree kernel applied to our semantic tree.
- $FK(x,y) = f(x) \cdot f(y)$ is the polynomial kernel applied to the global features.

The tagging kernels between two tagged sequences are defined as follows:

1. $CTK = \alpha \cdot K + (1-\alpha) \cdot TK$
2. $CFK = \beta \cdot K + (1-\beta) \cdot FK$
3. $CTFK = \gamma \cdot K + (1-\gamma) \cdot (TK + FK)$,

where α, β, γ are parameters weighting the two participating terms. Experiments on the validation set showed that these combinations yield the best performance with $\alpha = 0.2$ for both languages, $\beta = 0.4$ for English and $\beta = 0.3$ for and Italian, $\gamma = 0.24$ for English and $\gamma = 0.2$ for Italian.

6. Experiments

In these experiments, we compare the baseline performance with the different reranking models presented in the previous section.

6.1. Experimental setup

As a baseline we trained the CRF classifier on the entire training set (11,227 sentences in the Italian and 14,987 sentences in the English corpus). In developing a reranking strategy for both English and Italian, the training data was split into 5 sections, and in each case the baseline classifier was trained on 4/5 of the data, then used to decode the remaining 1/5.

The top 10 hypotheses together with their log probabilities were computed for each training sentence. Similarly, a model trained on the whole training data was used to produce 10 hypotheses for each sentence in the development set. For the reranking experiments, we applied different kernel setups to the two corpora described in Section 3. We enriched the feature set of the base CRF classifier (presented in section 4.3) with the most frequent tag added to each token.

English Test	Pr	Re	F_1
CRFs	85.37	84.35	84.86
CTK	87.19	84.79	85.97
CFK	86.53	86.75	86.64
CTFK	**88.07**	**87.91**	**87.99**
(Finkel et al., 2005)	*N/A*	*N/A*	*86.86*
(Ratinov and Roth, 2009)	*N/A*	*N/A*	*90.57*

Table 9

Reranking results on the English test set and comparison with the state-of-the-art systems (last two rows). Note that the first row indicates the CRF baseline result obtained in Section 4.3

Italian Test	Pr	Re	F_1
CRFs	83.43	77.48	80.34
CTK	84.97	78.03	81.35
CFK	84.93	79.13	81.93
CTFK	**85.99**	**82.73**	**84.33**
(Gesmundo, 2009)	*86.06*	*77.33*	*81.46*
(Zanoli et al., 2009)	*84.07*	*80.02*	*82.00*

Table 10

Reranking results on the Italian test set and comparison with the state-of-the-art systems (last two rows). Note that the first row indicates the CRF baseline result obtained in Section 4.3

6.2. Reranking Results

Tables 9 and 10 present the reranking results on the test data of both corpora. The results show a 20.29% relative improvement in F-measure for Italian and 21.79% for English.

CFK based on unstructured features achieves higher accuracy than *CTK* based on structured features. However, the huge amount of subtrees generated by the PT kernel may limit the expressivity of some structural features, e.g. many fragments may only generate noise. This problem is less important with the polynomial kernel where global features are tailored for individual entities. In any case, the experiments demonstrate that both tagging kernels *CTK* and *CFK* improve on the CRFs baseline in both languages. This suggests that structured and unstructured features are effective in discriminating between competing NE annotations.

Furthermore, the combination of the two tagging kernels on both standard corpora shows a large improvement in F-measure from 80.34% to 84.33% for Italian and from 84.86% to 88.16% for English. This suggests that these two kernels, corresponding to two kinds of features, complement each other.

To better collocate our results with previous work, we report the state of the art for NER on the Italian (Zanoli et al., 2009; Gesmundo, 2009) and the English (Ratinov and Roth, 2009; Finkel et al., 2005) datasets, in the last two rows (in italic) of each table, where the very last row shows the highest results. This demonstrates that our model outperforms the best Italian NER system and it is close to the state-of-art model for English, which exploits many complex features. Also note that we are very close to the F1 achieved by the best system of CoNLL 2003, i.e. 88.8.

7. Conclusion

In this paper, we have studied the use of discriminative reranking for named entity recognition. The major contribution with respect to previous work is the use of structural kernels for representing hypotheses of the NE annotation over an entire sentence. In particular, we first use a basic NER, e.g., built with CRFs or SVMs, to generate global hypotheses of NE annotation of the sentence. Then, we apply a reranker, i.e., a classifier based on SVMs and tree kernels, to select the best hypothesis. Such reranker can better select the most accurate hypothesis since it can use features describing the global view of the sentence annotation and its label dependencies. In particular, tree kernels are

applied to a tree representation of the NE annotation. This way, all the possible tree fragments generated by the kernel correspond to features encoding global dependencies between the NE labels and words of the sentence.

Another interesting contribution of our paper is the combination of tree kernels with global features encoded in traditional vectors. This joint model is rather effective as it can capture two different important aspects of the sentence annotation: tree kernels provide structural dependencies between NEs, whereas the vectors can describe global properties without structure, which are less sparse. Additionally, we used innovative features, e.g., the most frequent tag associated with words in an external corpus and the KIM gazetteer.

The comparative results on two well-known benchmarks in Italian and English language suggest that the partial tree kernel applied to our structural representations is rather effective. Indeed, we could improve the state of the art of the Italian NER and achieve almost the same accuracy of very complex model for English. Again, this improvement is due to the richer/global features available to the reranker, which always improve the basic model. Nevertheless, (Ratinov and Roth, 2009), applying very accurate and complex *manual* feature engineering, provide a higher accuracy for English. This can be due to several factors, e.g., we did not re-implement their features. It is worth noting that their system can be used as basic input of our reranker, thus our model may improve it.

Finally, as our reranking kernels are rather efficient and effective, in the future, we would like to use them for other tasks, e.g., reranking sentence-level relation extraction or coreference resolution annotations.

Acknowledgement

This research has been supported by the EC framework FP7/2007-2013 under the grants #247758: ETERNALS – Trustworthy Eternal Systems via Evolving Software, Data and Knowledge, and #288024: LIMOSINE – Linguistically Motivated Semantic aggregation engiNes.

References

Marco Baroni, Silvia Bernardini, Federica Comastri, Lorenzo Piccioni, Alessandra Volpi, Ra Volpi, Guy Aston, and Marco Mazzoleni. 2004. Introducing the la repubblica corpus: A large, annotated, tei(xml)-compliant corpus of newspaper italian. In *In LREC 2004*, pages 1771–1774.

Oliver Bender, Franz Josef Och, and Hermann Ney. 2003. Maximum entropy models for named entity recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 148–151. Edmonton, Canada.

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, ANLC '97, pages 194–201, Stroudsburg, PA, USA.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Xavier Carreras, Lluís Màrquez, and Lluís Padró. 2003a. Learning a perceptron-based named entity chunker via online recognition feedback. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 156–159. Edmonton, Canada.

Xavier Carreras, Lluís Màrquez, and Lluís Padró. 2003b. A simple named entity extractor using adaboost. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 152–155. Edmonton, Canada.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Hai Leong Chieu and Hwee Tou Ng. 2002. Named entity recognition: A maximum entropy approach using global information. In *In Proceedings of COLING02*, pages 190–196.

Hai Leong Chieu and Hwee Tou Ng. 2003. Named entity recognition with a maximum entropy approach. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 160–163. Edmonton, Canada.

Nancy Chinchor and Patricia Robinson. 1998. Muc-7 named entity task definition. In *The MUC*.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proceedings of NIPS*, pages 625–632.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of ICML*, ICML '00, pages 175–182. Morgan Kaufmann Publishers Inc.

Michael Collins. 2002. Ranking algorithms for named entity extraction: Boosting and the votedperceptron. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 489–496, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

James R. Curran and Stephen Clark. 2003. Language independent ner using a maximum entropy tagger. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 164–167. Edmonton, Canada.

Marco Dinarelli, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Re-ranking models based-on small training data for spoken language understanding. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1076–1085, Singapore, August. Association for Computational Linguistics.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of CoNLL*, pages 168–171, Edmonton, Canada.

Andrea Gesmundo. 2009. Bidirectional Sequence Classification for Part of Speech Tagging. In *Proceedings of Evaluation of NLP and Speech Tools for Italian*, Reggio Emilia, Italy.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08: HLT*, pages 586–594, Columbus, Ohio, June. Association for Computational Linguistics.

Richard Johansson and Alessandro Moschitti. 2010. Reranking models in fine-grained opinion analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 519–527, Beijing, China, August. Coling 2010 Organizing Committee.

Richard Johansson and Alessandro Moschitti. 2013. Relational features in fine-grained opinion analysis. *Computational Linguistics*, 39(3).

Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, and Damyan Ognyanoff. 2004. Semantic annotation, indexing, and retrieval. *J. Web Sem.*, 2(1):49–79.

John D. Lafferty, Andrew McCallum, and Fernando C. N.Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Timothy Robert Leek. 1997. Information extraction using hidden markov models.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, , and Chris Watkins. 2002. Text classification using string kernels. *journal of Machine Learning Research*, pages 419–444.

Bernardo Magnini, Emmanuele Pianta, Christian Girardi, Matteo Negri, Lorenza Romano, Manuela Speranza, Valentina Bartalesi Lenzi, and Rachele Sprugnoli. 2006. I-CAB: the italian content annotation bank. In *Proceedings of LREC*.

James Mayfield, Paul McNamee, and Christine Piatko. 2003. Named entity recognition using hundreds of thousands of features. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 184–187. Edmonton, Canada.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 188–191. Edmonton, Canada.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2006. Semantic role labeling via tree kernel joint inference. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 61–68, New York City, June. Association for Computational Linguistics.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2008. Tree kernels for semantic role label-

ing. *Computational Linguistics*, 34:193–224, June.

Alessandro Moschitti, Qi Ju, and Richard Johansson. 2012. Modeling topic dependencies in hierarchical text categorization. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 759–767, Jeju Island, Korea, July. Association for Computational Linguistics.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 335–342, Barcelona, Spain, July.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of ECML*, pages 318–329, Berlin, Germany, September. Machine Learning: ECML 2006, 17th European Conference on Machine Learning.

Alessandro Moschitti. 2008. Kernel methods, syntax and semantics for relational text categorization. In *Proceedings of CIKM*, pages 253–262, New York, NY, USA. ACM.

Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Conditional random fields: Discriminative training over statistical features for named entity recognition. In *Proceedings of EVALITA 2009 workshop, the 11st International Conference of the Italian Association for Artificial Intelligence (AI*IA)*, Reggio Emilia, Italy, December.

Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2010. Kernel-based reranking for named-entity extraction. In *Coling 2010: Posters*, pages 901–909, Beijing, China, August. Coling 2010 Organizing Committee.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June. Association for Computational Linguistics.

Aliaksei Severyn and Alessandro Moschitti. 2012. Structural relationships for large-scale learning of answer re-ranking. In *SIGIR*, pages 741–750.

J. Shawe-Taylor and N. Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *Proceedings of HLT-NAACL*, pages 177–184, Boston, Massachusetts, USA, May 2 - May 7.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of ACL*, pages 8–15, Sapporo, Japan, July.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Roberto Zanoli, Emanuele Pianta, and Claudio Giuliano. 2009. Named entity recognition through redundancy driven classifiers. In *EVALITA*.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.