Kernel-based Reranking for Named-Entity Extraction

Truc-Vien T. Nguyen and **Alessandro Moschitti** and **Giuseppe Riccardi**
Department of Information Engineering and Computer Science
University of Trento
nguyenthi,moschitti,riccardi@disi.unitn.it

Abstract

We present novel kernels based on structured and unstructured features for reranking the *N*-best hypotheses of conditional random fields (CRFs) applied to entity extraction. The former features 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 88.16%, respectively. Our analysis reveals that both kernels provide a comparable improvement over the CRFs baseline. Additionally, their combination improves CRFs much more than the sum of the individual contributions, suggesting an interesting kernel synergy.

1 Introduction

Reranking is a promising computational framework, which has drawn special attention in the Natural Language Processing (NLP) community. Basically, this method first employs a probabilistic model to generate a list of top-*n* candidates and then reranks this *n*-best list with additional features. One appeal of this approach is its flexibility of incorporating arbitrary features into a model. These features help in discriminating good from bad hypotheses and consequently their automatic learning. Various algorithms have been applied for reranking in NLP applications (Huang, 2008; Shen et al., 2004; Collins, 2002b; Collins and Koo, 2000), including parsing, name tagging and machine translation. This work has exploited the disciminative property as one of the key criterion of the reranking algorithm.

Reranking appears extremely interesting if coupled with kernel methods (Dinarelli et al., 2009; Moschitti, 2004; Collins and Duffy, 2001), as the latter allow for extracting from the ranking hypotheses a huge amount of features along with their dependencies. Indeed, while feature-based learning algorithms involve only 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. An example is the tree kernel (Collins and Duffy, 2001), where the objects are syntactic trees that encode grammatical derivations and the kernel function computes the number of common *subtrees*. Similarly, sequence kernels (Lodhi et al., 2002) count the number of common *subsequences* shared by two input strings.

Named-entities (NEs) are essential for defining the semantics of a document. NEs are objects that can be referred by names (Chinchor and Robinson, 1998), such as people, organizations, and locations. 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). In the literature, there exist various learning approaches to extract named-entities from text. A NER sys-

Coling 2010: Poster Volume, pages 901–909,
Beijing, August 2010

tem often builds some generative/discriminative model, then, either uses only one classifier (Carreras et al., 2002) or combines many classifiers using some heuristics (Florian et al., 2003).

To the best of our knowledge, reranking has not been applied to NER except for the reranking algorithms defined in (Collins, 2002b; Collins, 2002a), which only targeted the entity detection (and not entity classification) task. Besides, since kernel methods offer a natural way to exploit linguistic properties, applying kernels for NE reranking is worthwhile.

In this paper, we describe how kernel methods can be applied for reranking, i.e. detection and classification of named-entities, in standard corpora for Italian and English. The key aspect of our reranking approach is how structured and flat features can be employed in discriminating candidate tagged sequences. For this purpose, we apply tree kernels to a tree structure encoding NE tags of a sentence and combined them with a polynomial kernel, which efficiently exploits global features.

Our main contribution is to show that (a) tree kernels can be used to define general features (not merely syntactic) and (b) using appropriate algorithms and features, reranking can be very effective for named-entity recognition. 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, the 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 in the Italian corpus shows that our method outperforms the best reported methods whereas on the English data it reaches the state-of-the-art.

2 Background

2.1 The data

Different languages exhibit different linguistic phenomena and challenges. A robust NER system is expected to be well-adapted to multiple domains and languages. Therefore, we experimented with two datasets: the EVALITA 2009 Italian corpus and the well-known CoNLL 2003 English shared task corpus.

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, consist of completely new data, taken from the same newspaper and consists of 180 news stories.

The CoNLL 2003 English dataset is created within the shared task of CoNLL-2003 (Sang and 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.

Italian	GPE	LOC	ORG	PER
Train	2813	362	3658	4577
	24.65%	3.17%	32.06%	40.11%
Test	1143	156	1289	2378
	23.02%	3.14%	25.96%	47.89%

English	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 Italian EVALITA 2009 and English CoNLL 2003 corpora.

2.2 The baseline algorithm

We selected Conditional Random Fields (Lafferty et al., 2001) as the baseline model. Conditional

random fields (CRFs) 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 of an entity, or not part of an entity at all. For example, consider the sentence with their corresponding NE tags, each word is labeled with a tag indicating its appropriate named-entity, resulting in annotated text, such as:

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 at k words centered in the target word *w* (i.e. the one we want to classify with the B, O, I tags). More in detail such features are:

- The **word itself**, its **prefixes**, **suffixes**, and **part-of-speech**
- **Orthographic/Word features**. These are binary and mutually exclusive features that test whether a word contains *all upper-cased, initial letter upper-cased, all lower-cased, roman-number, dots, hyphens, acronym, lonely initial, punctuation mark, single-char, and functional-word*.
- **Gazetteer features**. Class (geographical, first name, surname, organization prefix, location prefix) of words in the window.
- **Left Predictions**. The predicted tags on the left of the word in the current classification.

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

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], The Alexandria Digital Library (ADL) gazetteer[3]. The company data is included with all the publicly traded companies listed in Google directory[4], the European business directory[5]. For Italian, the generic proper nouns are extracted from Wikipedia and various Italian sites.

2.3 Support Vector Machines (SVMs)

Support Vector Machines refer to a supervised machine learning technique based on the latest results of the statistical learning theory. Given a vector space and a set of training points, i.e. positive and negative examples, SVMs find a separating hyperplane $H(\vec{x}) = \vec{\omega} \times \vec{x} + b = 0$ where $\omega \in R^n$ and $b \in R$ are learned by applying the Structural Risk Minimization principle (Vapnik, 1998). SVMs are a binary classifier, but they can be easily extended to multi-class classifier, e.g. by means of the *one-vs-all* method (Rifkin and Poggio, 2002).

One strong point of SVMs is the possibility to apply kernel methods to implicitly map data in a new space where the examples are *more easily* separable as described in the next section.

2.4 Kernel methods

Kernel methods (Schölkopf and Smola, 2001) are an attractive alternative to feature-based methods since the applied learning algorithm only needs to compute a product between a pair of objects (by means of kernel functions), avoiding the explicit feature representation. A kernel function is a scalar product in a possibly unknown feature space. More precisely, The object *o* is mapped in $\vec{x}$ with a feature function $\phi : \mathcal{O} \rightarrow \Re^n$, where $\mathcal{O}$ is the set of the objects.

The kernel trick allows us to rewrite the decision hyperplane as:

$$H(\vec{x}) = \Big(\sum_{i=1..l} y_i \alpha_i \vec{x}_i \Big) \cdot \vec{x} + b =$$

[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

$$\sum_{i=1..l} y_i \alpha_i \vec{x}_i \cdot \vec{x} + b = \sum_{i=1..l} y_i \alpha_i \phi(o_i) \cdot \phi(o) + b,$$

where y_i is equal to 1 for positive and -1 for negative examples, $\alpha_i \in \Re$ with $\alpha_i \geq 0$, o_i $\forall i \in \{1, .., l\}$ are the training instances and the product $K(o_i, o) = \langle \phi(o_i) \cdot \phi(o) \rangle$ is the kernel function associated with the mapping ϕ.

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.

Regarding NLP applications, kernel methods have attracted much interest due to the ability of implicitly exploring huge amounts of structural features. The parse tree kernel (Collins and Duffy, 2001) and string kernel (Lodhi et al., 2002) are examples of the well-known convolution kernels used in various NLP tasks.

2.5 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 (ST), the SubSet Trees (SST) and the Partial Trees (PT). For sake of space, we do not report the mathematical description of them, which is available in (Vishwanathan and Smola, 2002), (Collins and Duffy, 2001) and (Moschitti, 2006), respectively. In contrast, we report some descriptions in terms of feature space that may be useful to understand the new engineered kernels.

In principle, a SubTree (ST) is defined by taking any node along with its descendants. A SubSet Tree (SST) is a more general structure which does not necessarily include all the descendants. The distinction is that an SST must be generated by applying the same grammatical rule set which generated the original tree, as pointed out in (Collins and Duffy, 2001). A Partial Tree (PT) is a more general form of sub-structures obtained by relaxing constraints over the SSTs. Figure 1 shows the overall fragment set of the ST, SST and PT kernels for the syntactic parse tree of the sentence fragment: *gives a talk* .

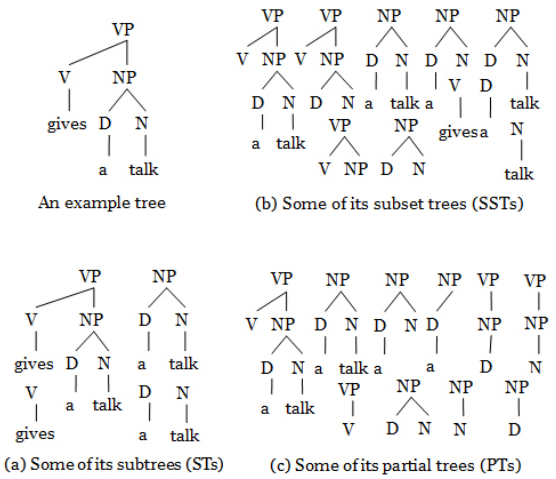

Figure 1: Three kinds of tree kernels.

In the next section, we will define new structures for tagged sequences of NEs which along with the application of the PT kernel produce innovative tagging kernels for reranking.

3 Reranking Method

3.1 Reranking Strategy

As a baseline we trained the CRFs model to generate *10*-best candidates per sentence, along with their probabilities. Each candidate was then represented by a semantic tree together with a feature vector. We consider our reranking task as a binary classification problem where examples are pairs of hypotheses $< H_i, H_j >$.

Given a sentence "**South African Breweries Ltd bought stakes in the Lech and Tychy brewers**" and three of its candidate tagged sequences:

H_1 *B-ORG I-ORG I-ORG I-ORG O O O O B-ORG O B-ORG O* (the correct sequence)

H_2 *B-MISC I-MISC B-ORG I-ORG O O O O B-ORG I-ORG I-ORG O*

H_3 *B-ORG I-ORG I-ORG I-ORG O O O O B-ORG O B-LOC O*

where B-ORG, I-ORG, B-LOC, O are the generated NE tags according to IOB notation as described in Section 3.2.

With the above data (an original sentence together with a list of candidate tagged sequences), the following pairs of hypotheses will be gener-

ated $< H_1, H_2 >$, $< H_1, H_3 >$,$< H_2, H_1 >$ and $< H_3, H_1 >$, where the first two pairs are positive and the latter pairs are negative instances. Then a binary classifier based on SVMs and kernel methods can be trained to discriminate between the best hypothesis, i.e. $< H_1 >$ and the others. At testing time the hypothesis receiving the highest score is selected (Collins and Duffy, 2001).

3.2 Representation of Tagged Sequences in Semantic Trees

We now consider the representation that exploits the most discriminative aspects of candidate structures. 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 the tree structure. See figure 2 as an example of candidate tagged sequence and its semantic tree.

With the sentence "**South African Breweries Ltd bought stakes in the Lech and Tychy brewers**" and three of its candidate tagged sequences in the previous section, the training algorithm considers to construct a tree for each sequence, with the named-entity tags as pre-terminals and the words as leaves. See figure 2 for an example of the semantic tree for the first tagged sequence.

With this tree representation, for a word w_i, the target NE tag would be set at parent and the features for this word are at child nodes. This allows us to best exploit the inner product between competing candidates. Indeed, in the kernel space, the inner product counts the number of common subtrees thus sequences with similar NE tags are likely to have higher score. For example, the similarity between H_1 and H_3 will be higher than the similarity of the previous hypotheses with H_2; this is reasonable since these two also have higher F_1.

It is worth noting that another useful modification is the flexibility of incorporate diverse, arbitrary features into this tree structure by adding children to the parent node that contains entity tag. These characteristics can be exploited efficiently with the PT kernel, which relaxes constraints of production rules. The inner product can implicitly include these features and deal better with sparse data.

3.3 Global features

Mixed *n-grams* features

In previous works, some global features have been used (Collins, 2002b; Collins, 2002a) 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 *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 (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 the target un-annotated 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 word w of the test set, we can rely on the most probable NE category as feature. 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 unnanotated corpus.

The unlabeled corpus for Italian was collected from La Repubblica [6] and it contains over 20 millions words. Whereas the unlabeled corpus 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 (Bunescu and Mooney, 2005a; Surdeanu et al., 2003), it is in-

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

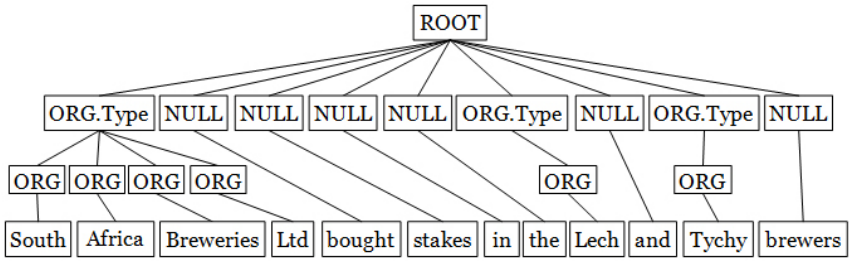

Figure 2: Semantic structure of the first sequence

cluded in the global set together with its orthographic feature. We now describe some primitives for our global feature framework.

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 gazetteers/frequent-tag before/after the start/end of an entity. In addition to the normal *n-grams* solely based on words, we mixed words with gazetteers/frequent-tag seen from a large corpus and create mixed *n-grams* features.

Table 2 shows the full set of global features in our reranking framework. Features are anchored to each entity instance and adapted to entity types. This helps to discriminate different entities with the same surface forms. Moreover, they can be combined with *n-grams* patterns to learn and explicitly push the score of the correct sequence above the score of competing sequences.

3.4 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 tree and entity kernels. Thus we focus on the problem to define structure embedding the desired relational information among tagged sequences.

The Partial Tree Kernel

Let $F = f_1, f_2, \ldots, f_{|F|}$ be a tree fragment space of type PTs and let the indicator function $I_i(n)$ be equal to 1 if the target f_1 is rooted at node n and 0 otherwise, we define the PT kernel as:

$$K(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$$

where N_{T_1} and N_{T_2} are the set of nodes in T_1 and T_2 respectively and $\Delta(n_1, n_2) = \sum_{i=1}^{|F|} I_i(n_1) I_i(n_2)$, i.e. the number of common fragments rooted at the n_1 and n_2 nodes of the type shown in Figure 1.c.

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.

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:

Feature	Description
$w_s\_w_{s+1}\_\ldots\_w_e$	Entity string
$g_s\_g_{s+1}\_\ldots\_g_e$	The gazetteer feature within the entity
$f_s\_f_{s+1}\_\ldots\_f_e$	The most frequent NE tag feature (seen from a large corpus) within the entity
hw	The head word of the entity
lhw	Indicates whether the head word is lower-cased
$w_{s-1}\_w_s$; $w_{s-1}\_g_s$; $g_{s-1}\_w_s$; $g_{s-1}\_g_s$	Mixed bigrams of the words/gazetteer features before/after the start of the entity
$w_e\_w_{e+1}$; $w_e\_g_{e+1}$; $g_e\_w_{e+1}$; $g_e\_g_{e+1}$	Mixed bigrams of the words/gazetteer features before/after the end of the entity
$w_{s-1}\_w_s$; $w_{s-1}\_f_s$; $f_{s-1}\_w_s$; $f_{s-1}\_f_s$	Mixed bigrams of the words/frequent-tag features before/after the start of the entity
$w_e\_w_{e+1}$; $w_e\_f_{e+1}$; $f_e\_w_{e+1}$; $f_e\_f_{e+1}$	Mixed bigrams of the words/frequent-tag features before/after the end of the entity
$w_{s-2}\_w_{s-1}\_w_s$; $w_{s-1}\_w_s\_w_{s+1}$; $w_{e-1}\_w_e\_w_{e+1}$; $w_{e-2}\_w_{e-1}\_w_e$	Trigram features of the words before/after the start/end of the entity
$w_{s-2}\_w_{s-1}\_g_s$; $w_{s-2}\_g_{s-1}\_w_s$; $w_{s-2}\_g_{s-1}\_g_s$; $g_{s-2}\_w_{s-1}\_w_s$; $g_{s-2}\_w_{s-1}\_g_s$; $g_{s-2}\_g_{s-1}\_w_s$; $g_{s-2}\_g_{s-1}\_g_s$; $w_{s-1}\_w_s\_g_{s+1}$; $w_{s-1}\_g_s\_w_{s+1}$; $w_{s-1}\_g_s\_g_{s+1}$; $g_{s-1}\_w_s\_w_{s+1}$; $g_{s-1}\_w_s\_g_{s+1}$; $g_{s-1}\_g_s\_w_{s+1}$; $g_{s-1}\_g_s\_g_{s+1}$	Mixed trigrams of the words/gazetteer features before/after the start of the entity
$w_{e-1}\_w_e\_g_{e+1}$; $w_{e-1}\_g_e\_w_{e+1}$; $w_{e-1}\_g_e\_g_{e+1}$; $g_{e-1}\_w_e\_w_{e+1}$; $g_{e-1}\_w_e\_g_{e+1}$; $g_{e-1}\_g_e\_w_{e+1}$; $g_{e-1}\_g_e\_g_{e+1}$; $w_{e-2}\_w_{e-1}\_g_e$; $w_{e-2}\_g_{e-1}\_w_e$; $w_{e-2}\_g_{e-1}\_g_e$; $g_{e-2}\_w_{e-1}\_w_e$; $g_{e-2}\_w_{e-1}\_g_e$; $g_{e-2}\_g_{e-1}\_w_e$; $g_{e-2}\_g_{e-1}\_g_e$	Mixed trigrams of the words/gazetteer features before/after the end of the entity
$w_{s-2}\_w_{s-1}\_f_s$; $w_{s-2}\_f_{s-1}\_w_s$; $w_{s-2}\_f_{s-1}\_f_s$; $f_{s-2}\_w_{s-1}\_w_s$; $f_{s-2}\_w_{s-1}\_f_s$; $f_{s-2}\_f_{s-1}\_w_s$; $f_{s-2}\_f_{s-1}\_f_s$; $w_{s-1}\_w_s\_f_{s+1}$; $w_{s-1}\_f_s\_w_{s+1}$; $w_{s-1}\_f_s\_f_{s+1}$; $f_{s-1}\_w_s\_w_{s+1}$; $f_{s-1}\_w_s\_f_{s+1}$; $f_{s-1}\_f_s\_w_{s+1}$; $f_{s-1}\_f_s\_f_{s+1}$	Mixed trigrams of the words/frequent-tag features before/after the start of the entity
$w_{e-1}\_w_e\_f_{e+1}$; $w_{e-1}\_f_e\_w_{e+1}$; $w_{e-1}\_f_e\_f_{e+1}$; $f_{e-1}\_w_e\_w_{e+1}$; $f_{e-1}\_w_e\_f_{e+1}$; $f_{e-1}\_f_e\_w_{e+1}$; $f_{e-1}\_f_e\_f_{e+1}$; $w_{e-2}\_w_{e-1}\_f_e$; $w_{e-2}\_f_{e-1}\_w_e$; $w_{e-2}\_f_{e-1}\_f_e$; $f_{e-2}\_w_{e-1}\_w_e$; $f_{e-2}\_w_{e-1}\_f_e$; $f_{e-2}\_f_{e-1}\_w_e$; $f_{e-2}\_f_{e-1}\_f_e$	Mixed trigrams of the words/frequent-tag features before/after the end of the entity

Table 2: Global features in the entity kernel for reranking. 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".

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

The tagging kernels between two tagged sequences are defined in the following combinations:

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.

4 Experimens and Results

4.1 Experimental Setup

As a baseline we trained the CRFs classifier on the full training portion (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 recovered 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 2.1. The three kernels were trained on the training portion.

Italian Test	P	R	F
$CRFs$	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**
(Zanoli et al., 2009)	*84.07*	*80.02*	*82.00*

English Test	P	R	F
$CRFs$	85.37	84.35	84.86
CTK	87.19	84.79	85.97
CFK	86.53	86.75	86.64
CTFK	**88.07**	**88.25**	**88.16**
(Ratinov and Roth,)	*N/A*	*N/A*	*90.57*

Table 3: Reranking results of the three tagging kernels on the Italian and English testset.

4.2 Discussion

Table 3 presents 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 give improvement over 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 data. This suggests that these two kernels, corresponding to two kinds of feature, complement each other.

To better collocate our results with previous work, we report the best NER outcome on the Italian (Zanoli et al., 2009) and the English (Ratinov and Roth,) datasets, in the last row (in italic) of each table. This shows 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[9]. Also note that we are very close to the F1 achieved by the best system of CoNLL 2003, i.e. 88.8.

5 Conclusion

We analyzed the impact of kernel-based approaches for modeling dependencies between tagged sequences for NER. Our study illustrates that each individual kernel, either with structured or with flat features clearly gives improvement to the base model. Most interestingly, as we showed, these contributions are independent and, the approaches can be used together to yield better results. The composite kernel, which combines both kinds of features, can outperform the state-of-the-art.

In the future, it will be very interesting to use syntactic/semantic kernels, as for example in (Basili et al., 2005; Bloehdorn and Moschitti, 2007a; Bloehdorn and Moschitti, 2007b). Another promising direction is the use of syntactic trees, feature sequences and pairs of instances, e.g. (Nguyen et al., 2009; Moschitti, 2008).

Acknowledgments

We would like to thank Roberto Zanoli and Marco Dinarelli for helpful explanation about their work. This work has been partially funded by the LiveMemories project (http://www.livememories.org/) and Expert System (http://www.expertsystem.net/) research grant.

[9] In the future we will be able to integrate them with the authors collaboration.

References

Basili, Roberto, Marco Cammisa, and Alessandro Moschitti. 2005. Effective use of WordNet semantics via kernel-based learning. In *CoNLL*.

Bloehdorn, Stephan and Alessandro Moschitti. 2007a. Combined syntactic and semantic kernels for text classification. In *ECIR*.

Bloehdorn, Stephan and Alessandro Moschitti. 2007b. Structure and semantics for expressive text kernels. In *CIKM*.

Bunescu, Razvan C. and Raymond J. Mooney. 2005a. A shortest path dependency kernel for relation extraction. In *EMNLP*.

Carreras, Xavier, Lluís Màrques, and Llus Padró. 2002. Named entity extraction using Adaboost. In *CoNLL*.

Chieu, Hai Leong and Hwee Tou Ng. 2002. Named entity recognition: A maximum entropy approach using global information. In *COLING*.

Chieu, Hai Leong and Hwee Tou Ng. 2003. Named entity recognition with a maximum entropy approach. In *CoNLL*.

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

Collins, Michael and Nigel Duffy. 2001. Convolution kernels for natural language. In *NIPS*.

Collins, Michael and Terry Koo. 2000. Discriminative reranking for natural language parsing. In *ICML*.

Collins, Michael. 2002a. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *ACL*.

Collins, Michael. 2002b. Ranking algorithms for named-entity extraction boosting and the voted perceptron. In *ACL*.

Dinarelli, Marco, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Re-ranking models based on small training data for spoken language understanding. In *EMNLP*.

Florian, Radu, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *CoNLL*.

Huang, Liang. 2008. Forest reranking: Discriminative parsing with non-local features. In *ACL-HLT*.

Lafferty, John, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

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

Magnini, Bernardo, 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 *LREC*.

Moschitti, Alessandro. 2004. A study on convolution kernels for shallow semantic parsing. In *ACL*.

Moschitti, Alessandro. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *ICML*.

Moschitti, Alessandro. 2008. Kernel methods, syntax and semantics for relational text categorization. In *CIKM*.

Nguyen, Truc-Vien T., Alessandro Moschitti, and Giuseppe Riccardi. 2009. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *EMNLP*.

Ratinov, Lev and Dan Roth. Design challenges and misconceptions in named entity recognition. In *CoNLL*.

Rifkin, Ryan Michael and Tomaso Poggio. 2002. *Everything old is new again: a fresh look at historical approaches in machine learning*. PhD thesis, MIT.

Sang, Erik F. Tjong Kim and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CoNLL*.

Schölkopf, Bernhard and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Shen, Libin, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *HLT-NAACL*, Boston, Massachusetts, USA.

Surdeanu, Mihai, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *ACL*.

Vapnik, Vladimir N. 1998. *Statistical Learning Theory*. John Wiley and Sons, New York.

Vishwanathan, S.V.N. and Alexander J. Smola. 2002. Fast kernels on strings and trees. In *NIPS*.

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