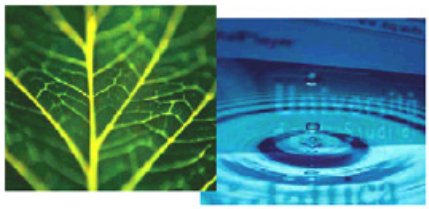
PhD Dissertation



International Doctorate School in Information and Communication Technologies

DISI - University of Trento

End-to-End Relation Extraction via Syntactic Structures and Semantic Resources

Truc-Vien T. Nguyen

Advisor:

Professor Alessandro Moschitti

Università degli Studi di Trento

June 2011

Abstract

Information Extraction (IE) aims at mapping texts into fixed structure representing the key information. A typical IE system will try to answer the questions like who are present in the text, what events happen and when these events happen. The task is making possible significant advances in applications that require deep understanding capabilities such as question-answering engines, dialogue systems, or the semantic web. Due to the huge effort and time consumation of developping extraction systems by domain experts, our approach focuses on machine learning methods that can accurately infer an extraction model by training on a dataset. The goal of this research is to design and implement models with improved performance by learning the combination of different algorithms or by inventing novel structures that are able to exploit kinds of evidence that have not been explored in the literature.

A basic component of an IE system is named entity recognition (NER) whose purpose is to locate objects that can be referred by names, belonging to a predefined set of categories. We approach this task by proposing a novel reranking framework that employs two learning phases to pick the

best candidate. The task is considered as sequence labelling with Conditional Random Fields (CRFs) is selected as the baseline algorithm. Our research employs novel kernels based on structured and unstructured features for reranking the N-best hypotheses from the CRFs baseline. The former features are generated by a polynomial kernel encoding entity features whereas tree kernels are used to model dependencies amongst tagged candidate examples.

Relation Extraction (RE) is concerned with finding relationships between pairs of entities in texts. State-of-the-art relation extraction model is based on convolution kernel over the constituent parse tree. In our research, we employ dependency parses from dependency parsing in addition to phrase-structure parses from constituent parsing. We define several variations of dependency parses to inject additional information into the trees. Additionally, we provide an extensive ablation over various types of kernels by combining the tree, sequence, and polynomial kernels. These novel kernels are able to exploit learned correlations between phrase-structure parses and grammatical relations.

A large amounts of wide-coverage semantic knowledge today exists in large repositories of unstructured or semi-structured text documents. The increased availability of online collaborative resources has attracted the attention of much work in the Artificial Intelligence (AI) community. Nevertheless, the ability to extract it using statistical machine learning techniques is hindered by well-known problems such as heavy supervision and scalability. These drawbacks can be alleviated by applying a form of weakly supervision, specifically named distant supervision (DS), to automatically derive explicit facts from the semi-structured part of Wikipedia.

To learn relational facts from Wikipedia without any labeled example or hand-crafted pattern, we employ DS where the relation providers are external repositories, e.g., YAGO (a huge semantic knowledge base), and the

training instances are gathered from Freebase (a huge semantic database). These allow for potentially obtaining larger training data and many more relations, defined in different sources. We apply state-of-the-art models for ACE RE, that are sentence level RE (SRLE), to Wikipedia. Based on a mapping table of relations from YAGO to ACE (according to their semantic definitions), we design a joint RE model of DS/ACE and tested it on ACE annotations (thus according to expert linguistic annotators). Moreover, we experiment with end-to-end systems for real-world RE applications. Consequently, our RE system is applicable to any document/sentence, i.e. another major improvement on previous work, which, to our knowledge, does not show experiments on end-to-end SLRE.

Keywords

[relation extraction, named entity recognition, kernel methods, tree kernel, reranking, distant supervision]

Acknowledgements

First and foremost, I want to thank my advisor Alessandro Moschitti, whose nice guidance and support have shaped my development to a researcher. His advices at any needed time have been leaded to lucid reasoning and clear explanation. His words of encouragement have also helped me overcome less fruitful periods in my research, and have shown me the value of being patient and proceeding the directed way.

I would like to thank Massimo Poesio for his encouragement, scientific advices, and insightful feedback in various stages of my PhD process. I thank Marco Baroni for prompt answers of almost every questions I have made. That has significantly helped to keep constant inspired ideas with enthusiasm. I am grateful to Fausto Giunchiglia for his active support, thanks to that my PhD defense has been hold smoothly.

I would like to thank the members of my thesis committee - Michael Strube, Razvan C. Bunescu, and Roberto Basili for their helpful comments during my defense, especially thank Michael Strube for his insightful feedbacks for the completion of this dissertation. I thank useful discussion and friendship from Richard Johansson. I thank friendship from Sucheta Ghosh

and thank Sebastian Varges for useful advices before he left the group.

I am grateful to Cogito S.p.a. for sponsoring me during three years of the Ph.D. and the LiveMemories project for a such useful academic environment and financial support for missions.

Finally, I would like to thank my family for believing in me and for encouraging me to pursue my career.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Natural Language Processing

In order to facilitate the understanding of human language, one needs knowledge from both linguistics and computer science perspectives. Natural Language Processing (NLP) lies as the interdisciplinary field between these two, dealing with statistical and/or rule-based modeling. In general, NLP strongly involves cognitive science, artificial intelligence, psycholinguistics, and neuroscience, among others. However, up to the 80s, most early NLP works were dominated by hand-built models, requiring a huge amount of human effort and time consuming. As a consequence, the past two decades have seen NLP algorithms mainly grounded in machine learning (ML). In this method, a corpus consisting of annotations made by human experts is used to infer a model. ML has made modern NLP algorithms require the understanding of a number of disparate fields, including linguistics, computer science, and statistics.

Most of ML-based approaches employ features that form the basic to infer a model. However, natural language tasks often have to deal with complex structures to be classified or clustered based on similarity. Thus, it is desirable to have a method of constructing kernels on sets whose elements are structures like strings, trees and graphs. To fulfill this need,

(Haussler, 1999) proposed convolution kernels that define a function between two objects based on similarities of their parts. These kernels allow a generalized convolution. However, linguistic objects prohibit this computation due to complex representations and thus, lead to huge number of sub-structures, (Collins and Duffy, 2001) showed how the algorithms can be efficiently applied to exponential sized representations of parse trees and tagged sequences.

1.2 Information Extraction

The term "Information Extraction" (IE) refers to a technology that is oriented towards the user's point of view in the current information-driven world. Rather than letting the user consider which documents need to be read, it extracts "relevant" pieces of information that answer the user's need. The kinds of information to be extracted vary in detail and tasks required. For example, a very classical task is extraction of named entities such as persons and organizations. However, it does not provide as much detail as attributes of the entities, relationships between them, or specific occurrences of events. Much has happened since the 90s with more demanding IE tasks and more steady evaluation methodology.

The research direction dates back in the 80s with the FRUMP system (DeJong, 1982), which seeks for event descriptions of newswire texts. In the early 1990, the MUC evaluations (DARPA, 1987 1995) were funding the development of metrics and statistical algorithms to support IE technologies. The results of these evaluations were reported at conferences that were called "Message Understanding Conferences (MUC)". The MUC proceedings were held in order to exchange research findings on the effectiveness of techniques used to extract a variety of levels of information from the formal test material. However, most early works were dominated by

hand-built models. The first use of machine learning for IE was only seen from (Bikel et al., 1997; Leek, 1997) with Hidden Markov Models (HMMs). Since then, various IE learning algorithms have been developed.

The Automatic Content Extraction programme (Doddington, 1999 2008) is a successor to MUC that has been running since a pilot study in 1999 and which has continued the competitive quantitative evaluation cycles of its predecessor. ACE differs from MUC in three significant ways: i) several MUC tasks are unified in ACE, for example the named entity recognition and co-reference resolution are conflated into one task "Entity Detection and Tracking (EDT)"; ii) the ACE program defines more complex tasks with more fine-grained taxonomy of entities and relations; iii) ACE results are restricted to participants and not public as MUC results.

Although the programs MUC and ACE provide several advantages to the IE community, the datasets and tasks developed are restricted in several ways. The data relies on pre-defined categories and, typically, these categories are given to an IE system as input to learn. Shifting to a new entity/relation requires a person to manually annotate new training examples with new labels. This expensive labor scales linearly with the number of categories required. Also, the dataset is annotated on a particular corpus, it tends to be biased towards the text domain it is built upon. These drawbacks can be alleviated by applying a form of weakly supervision, specifically named distant supervision (DS), using a huge amount of data (Etzioni et al., 2008; Mintz et al., 2009; Hoffmann et al., 2010), e.g. Wikipedia.

1.3 Convolution Kernels

Natural language tasks often has to deal with complex structures to be classified or clustered based on similarity. Thus, it is desirable to have a

method of constructing kernels on sets whose elements are structures like strings, trees and graphs. To fulfill this need, (Haussler, 1999) proposed convolution kernels that define a function between two objects based on the similarities of their parts. These kernels allow a generalized convolution. However, linguistic objects prohibit this computation due to complex representations and thus, lead to huge number of sub-structures. (Collins and Duffy, 2001) showed how the algorithms can be efficiently applied to exponential sized representations of parse trees and tagged sequences.

The syntactic (or subset) tree kernel (SST) (Collins and Duffy, 2001; Collins and Duffy, 2002) defines kernel funtion between parse trees which encode grammatical derivations. Tree kernel counts the number of subtrees shared by two input parse trees. Similarly, sequence kernel (Lodhi et al., 2002) tackles problems with objects that are strings of characters and the kernel function computes the number of common subsequences of characters in the two strings. Such substrings are then weighted according to a decay factor penalizing longer ones. Later, (Cancedda et al., 2003) proposed the use of this technique with sequences of words rather than characters. This approach shows efficient computationally since it ties in closely with standard linguistic pre-processing techniques that are based on words.

Since then, convolution kernels have been seen in numerous NLP tasks, including event extraction (Agarwal and Rambow, 2010), opinion analysis (Johansson and Moschitti, 2010; Wiegand and Klakow, 2010), question answering (Moschitti et al., 2007), semantic role labelling (Moschitti et al., 2008), relation extraction (Zhang et al., 2006; Nguyen et al., 2009b), named entity extraction and classification (Collins, 2002; Nguyen et al., 2010). These tasks either i) employ convolution kernels as key features to learn the model or ii) use them as evidence in discriminating between candidate tagged sequences.

1.4 Research Directions

Researchers have been noticed that NLP has been relied on ML. Statistical methods now dominate NLP, and have moved the field positively. Several ML models have been developed in the past two decades. How could we benefit from ML to purpose more robust IE systems? We follow three research directions:

1. From a linguistics point of view, semantics contrasts with syntax, the study of the combinatorics of units of a language (without reference to their meaning). However, semantics and syntax form two sides of human language, one may benefit from another to contribute for an overall effect. We propose an integrated model of syntactic and semantic computing for relation extraction. The proposed model makes use of structures derived from both the constituent and dependency parsing paradigms, integrated with semantic features derived from entities and relations.

2. From a computer science point of view, single learning algorithm does not lead to good enough accuracy. Different learning algorithms such as CRFs, SVMs employ their own encoding schemes, generalization techniques and optimization strategies. It is desirable to design a model that can combine multiple learning phases, exploiting various learning strategies with the target being to improve the basic model. Reranking is one of such models, providing the ability of using the output of one learning algorithm as the input of another.

3. From an artificial intelligence point of view, to cope with the need of labeled data in supervised learning setting, we employ distant supervision. However, the DS paradigm is based on the asumption that the repository of relation instances should be consistent with the textual

source (e.g. Freebase is a large repository of relation instances that is consistent with Wikipedia texts from Freebase Wikipedia Extraction). We relax this assumption by employing an external source that is YAGO. Although YAGO also uses Wikipedia as one source to build its relation instances. We show that, our method is generic and can be applied to any external source of relation instances and any Wikipedia document.

1.5 Thesis Contributions

The goal of this thesis is to design and develop computational models for information extraction systems with improved accuracy, but, at the same time, allow the most generalization.

1.5.1 Named Entity Recognition

Previous approaches for named entity recognition often take the first hypothesis ranked by the baseline for each sentence. However, in many cases, the first hypothesis is not the most accurate, i.e., its *F-measure* is not the highest among the N hypotheses. This means that reranking could improve the NER performance. Intuitively, arbitrary features different from those used by the base model should be employed by the reranker to exploit discriminative aspects of candidate tagged sequences. That has been seen in various reranking algorithms applied to parsing (Collins, 2000; Collins, 2002; Charniak and Johnson, 2005; Huang, 2008), machine translation (Shen et al., 2004), and opinion mining (Johansson and Moschitti, 2010).

Let $H_, \ldots, H_n$ be hypotheses generated over one sentence (figure 1.1), where each hypothesis H_i is represented by a list of named entity (NE) tags (figure 1.2). In order to capture correlations between NE tags of

Campese will be up against a familiar foe in the shape of **Barbarians** captain **Rob Andrew** , the man who kicked **Australia** to defeat with a last-ditch drop-goal in the **World Cup** quarter-final in **Cape Town**.

Figure 1.1: A sentence with the best hypothesis.

[Campese] [Barbarians] [Rob Andrew] [Australia] [World Cup] [Cape Town]
H_1: [O] [MISC] [PER] [LOC] [MISC] [LOC]
H_2: [O] [ORG] [PER] [LOC] [MISC] [LOC]
H_3: [PER] [MISC] [PER] [LOC] [MISC] [LOC]
H_4: [MISC] [MISC] [PER] [LOC] [MISC] [LOC]
H_5: [O] [LOC] [PER] [LOC] [MISC] [LOC]
H_6: [ORG] [MISC] [PER] [LOC] [MISC] [LOC]
H_7: [PER] [ORG] [PER] [LOC] [MISC] [LOC] (*the right hypothesis*)
H_8: [O] [O] [PER] [LOC] [MISC] [LOC]
H_9: [MISC] [ORG] [PER] [LOC] [MISC] [LOC]
H_{10}: [PER] [LOC] [PER] [LOC] [MISC] [LOC]

Figure 1.2: *10*-best hypotheses of one sentence.

various types of candidate tagged sequences, we propose using a framework that does linear combination of structural kernel and polynomial kernel. The former encodes the tree-like structure with entity types anchored as nodes and words as leaves. The latter is used to encode arbitrary features motivated from entity types with their *n-grams* of the preceding/following words.

The strength of these correlations is captured by a tree kernel and a polynomial kernel. Given a sentence with its N-best list, all pairs of hypotheses are generated. Then a binary classifier based on SVMs and kernel

methods can be trained 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). Experimental results on two standard datasets in two different languages show that a significant increase in performance is obtained when a linear combination of the structural and polonomial kernels is used.

1.5.2 Relation Extraction

We take the pairwise approach for the task of relation extraction (RE). Pair of entity mentions in the same sentence are generated as potential relations and then are classified as to whether they belong to a predefined set of relationships. We present a set of approaches to learn relational extractors that focus on one of two issues related to the nature of the task: 1) difference in the syntactic parsing structure and 2) difference in the type of supervision. We also propose a joint learning framework trying to combine two types of supervision and to improve the basic models.

Combining Parsing Paradigms

State-of-the-art relation extraction model is based on convolution kernel (Zhang et al., 2006) over the constituent parse tree. However, dependency structures offer some unique advantages, which should be exploited by an appropriate kernel. In our research, we employ syntactic parsing as the key features for our learning framework. In addition to phrase-structure parses from constituent parsing that are popular in NLP applications, we employ dependency parses from dependency parsing. We define several variations of dependency parses to inject additional information into the trees. Additionally, we provide an extensive ablation over various types of kernels by combining the tree, sequence, and polynomial kernels. These novel kernels are able to exploit learned correlations be-

tween phrase-structure parses and grammatical relations.

Distant Supervised Relation Extraction Using Kernel Methods

Previous work has shown that selecting the sentences containing the entities targeted by a given relation is enough accurate (Etzioni et al., 2008; Mintz et al., 2009) to provide reliable training data. However, only (Hoffmann et al., 2010) used DS to define extractors that are supposed to detect all the relation instances from a given input text. This is a harder test for the applicability of DS but, at the same time, the resulting extractor is very valuable: it can find rare relation instances that might be expressed in only one document. For example, the relation *President(Barrack Obama, United States*) can be extracted from thousands of documents thus there is a large chance of acquiring it. In contrast, *President(Eneko Agirre, SIGLEX)* is probably expressed in very few documents, increasing the complexity for obtaining it.

We extend DS by (i) considering relations from semantic repositories different from Wikipedia, i.e. YAGO, and (2) using training instances derived from any Wikipedia document. This allows for (i) potentially obtaining training data for many more relation types, defined in different sources; (ii) meaningfully enlarging the size of the DS data since the relation examples can be extracted from any Wikipedia document [1].

Additionally, by following previous work, we define state-of-the-art RE models based on kernel methods (KM) applied to syntactic/semantic structures. We use tree and sequence kernels that can exploit structural information and interdependencies among labels. Experiments show that our models are flexible and robust to Web documents as we achieve the interesting F1 of 74.29% on 52 YAGO relations. This is even more appreciable if we approximately compare with the previous result on RE using DS, i.e.

[1]Previous work assumes the page related to the *Infobox* as the only source for the training data.

61% (Hoffmann et al., 2010). Although the experiment setting is different from ours, the improvement of about 13 absolute percent points demonstrates the quality of our model.

Combining Learning Paradigms

In the second approach, we conduct a study on DS for the automatic acquisition of labeled data: the aim is to build sufficiently accurate RE applicable to general domains. Our approach is based on support vector machines (SVMs) and kernel methods applied to syntactic trees, where the latter are obtained with different parsing paradigms. We explore domain adaptation by defining a joint model between Web data derived with DS and manually annotated data from ACE. The results derived on both ACE and DS data demonstrate that our models improve previous state-of-the-art. Additionally, the learning ability of our approach allowed us to generalize the DS hypothesis to any relation using any Wikipedia document. This allows for enlarging the size of automatically produced RE datasets of several order of magnitude.

1.6 Thesis Outline

Below is a summary of the remaining chapters in this thesis, with references to the relevant publications:

- **Chapter 3:** This chapter presents a reranking framework that employs two learning phases to pick the best candidate for named entity recognition (Nguyen et al., 2010; Nguyen et al., 2009a).

- **Chapter 4:** We propose a novel convolution kernel that combines the two parsing structures - constituent and dependency parsing for relation extraction (Nguyen et al., 2009b).

- **Chapter 5:** We conduct intensive experiments to investigate the effects of relation order in the RE task using convolution kernels. The relation order is established using a number of entity properties.
- **Chapter 6:** An end-to-end distant supervised learning framework for extracting relational facts from Wikipedia articles with a relevant 67% F-measure (Nguyen and Moschitti, 2011). We also propose a joint model of distant and direct supervision.

Chapter 2

Background

In this chapter, we describe the background of the tasks this thesis is dealing with and basic algorithms that we employ. The tasks are concerned with automatic extraction of semantic aspects of text. Semantics (Liddell et al., 1891) is the study of meaning. It focuses on the relation between signifiers, such as words, phrases, signs and symbols, and what they stand for, their denotata. In computer science, the term refers to the meaning of languages, as opposed to their form (syntax). Additionally, it is applied to certain types of data structures specifically designed and used for representing information content. In this context, the term "semantics" is used to characterize concepts or entities in the world, and the relationships between them.

Following the literature, we define entity as one object in the world, and relation as binary relationship between pair of entities. Our tasks are framed as relation extraction (RE) and named entity recognition (NER). We explore the use of kernel methods based on syntactic and semantic structures for the targets RE and NER tasks. Syntax is derived from constituent and dependency parse trees whereas semantics concerns to entity types when used as flat features and as integrated in the syntactic parse tree as structured features. We investigate the effectiveness of such rep-

resentations in the automated RE from texts and in the incorporation of syntatic/semantic features into a reranking framework for NER.

2.1 Named Entity Recognition

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 2.1 shows a text from the CoNLL 2003 corpus, where all named entities are in bold.

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.

Figure 2.1: CoNLL text with all entities in bold.

The named entity recognition task involves either rule-based modeling or machine learning. Whereas the semantic web community mainly focused on rule based algorithms (Nguyen and Cao, 2007; Popov et al., 2003), natural language processing community mainly develops machine learning algorithms to reduce efforts and time required by domain experts. Existing approaches for NER using machine learning fall into two types.

A NER system may employ one learning algorithm to build a model. The used algorithms include maximum entropy (Bender et al., 2003; Chieu and Ng, 2003; Curran and Clark, 2003), hidden markov model (Zhou and Su, 2002), perceptron (Carreras et al., 2003a), adaboost (Carreras et al., 2003b), conditional random fields (McCallum and Li, 2003), support vector machines (Mayfield et al., 2003). However, single learning algorithm has limited performance, while possesses disparate properties and techniques, (Florian et al., 2003) presents a classifier combination experimental framework for NER in which four diverse classifiers are combined under different conditions.

2.2 Relation Extraction

Relation Extraction (RE) is defined as the task of finding relevant semantic relations between pairs of entities in texts. Figure 2.2 shows part of a document from the ACE 2004 corpus, a collection of news articles. In the text, the relation between *president* and *NBC's entertainment division* describes the relationship between the first entity (person) and the second (organization) where the person holds a managerial position.

Jeff Zucker, the longtime executive producer of NBC's "Today" program, will be named Friday as the new **president** of **NBC's entertainment division**, replacing Garth Ancier, NBC executives said.

Figure 2.2: An ACE 2004 relation with all entity mentions in bold.

To identify semantic relations using machine learning, three learning settings have mainly been applied, namely supervised methods (Miller et

al., 2000; Zelenko et al., 2002; Culotta and Sorensen, 2004; Kambhatla, 2004; Zhou et al., 2005), semi supervised methods (Brin, 1998; Agichtein and Gravano, 2000), and unsupervised method (Hasegawa et al., 2004). In a supervised learning setting, representative related work can be classified into generative models (Miller et al., 2000), feature-based (Roth and tau Yih, 2002; Kambhatla, 2004; Zhao and Grishman, 2005; Zhou et al., 2005) or kernel-based methods (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Zhang et al., 2005; Zhang et al., 2006).

The learning model employed in (Miller et al., 2000) used statistical parsing techniques to learn syntactic parse trees. It demonstrated that a lexicalized, probabilistic context-free parser with head rules can be used effectively for information extraction. Meanwhile, feature-based approaches often employ various kinds of linguistic, syntactic or contextual information and integrate into the feature space. (Roth and tau Yih, 2002) applied a probabilistic approach to solve the problems of named entity and relation extraction with the incorporation of various features such as word, part-of-speech, and semantic information from WordNet. (Kambhatla, 2004) employed maximum entropy models with diverse features including words, entity and mention types and the number of words (if any) separating the two entities.

Recent work on Relation Extraction has mostly employed kernel-based approaches over syntactic parse trees. Kernels on parse trees were pioneered by (Collins and Duffy, 2001). This kernel function counts the number of common subtrees, weighted appropriately, as the measure of similarity between two parse trees. (Culotta and Sorensen, 2004) extended this work to calculate kernels between augmented dependency trees. (Zelenko et al., 2002) proposed extracting relations by computing kernel functions between parse trees. (Bunescu and Mooney, 2005a) proposed a shortest path dependency kernel by stipulating that the information to model a

relationship between two entities can be captured by the shortest path between them in the dependency graph.

Although approaches in RE have been dominated by kernel-based methods, until now, most of research in this line has used the kernel as some similarity measures over diverse features (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Zhang et al., 2005). These are not convolution kernels and produce a much lower number of substructures than the PT kernel. A recent approach successfully employs a convolution tree kernel (of type SST) over constituent syntactic parse tree (Zhang et al., 2006; Zhou et al., 2007), but it does not capture grammatical relations in dependency structure.

2.3 Learning Machines

Machine learning allows to develop algorithms which learn from environment and improve automatically with experience. The input is often provided with a set of instances (so-called training data), from those a target function is learnt and applied to the output (i.e. test data). Usually the examples are in the form of attribute vectors, so that the input space is a subset of R^n. Once the attribute vectors are available, a number of sets of hypotheses could be chosen for the problem. Among these, linear functions are the best understood and simplest to apply.

Traditional statistics and the classical neural networks literature have developed many methods for discriminating between two classes of instances using linear functions, as well as methods for interpolation using linear functions. These techniques, which include both efficient iterative procedures and theoretical analysis of their generalisation properties, provide the framework within which the construction of more complex systems.

Binary classification is frequently performed by using a real-valued function $f : X \subseteq R^n \longrightarrow R$ in the way that $x = (x_1, \ldots, x_n)'$ is assigned to the positive class if $f(X) \geq 0$, and otherwise to the negative class. We consider the case where $f(X)$ is a linear function of $x \in X$ so that it can be written as:

$$\begin{aligned} f(x) &= \langle w \cdot x \rangle \\ &= \textstyle\sum_{i=1}^{n} w_i x_i + b \end{aligned}$$

where $(w, b) \in R^n \times R$ are the parameters that control the function and the decision rule is given by $sgn(f(x))$, where we will use the convention that $sgn(0) = 1$. The learning methodology implies that these parameters must be learned from the data.

2.4 Kernel Machines

The limited computational power of linear learning machines was highlighted in the 1960s by Minsky and Papert. Complex real-world applications require more expressive hypothesis spaces than linear functions. Kernel representations offer an alternative solution by projecting the data into a high dimensional feature space to increase the computational power of the linear learning machines. The use of linear machines in the dual representation makes it possible to perform this step implicitly.

By replacing the inner product with an appropriately chosen "kernel" function, one can implicitly perform a non-linear mapping to a high dimensional feature space without increasing the number of tunable parameters, provided the kernel computes the inner product of the feature vectors corresponding to the two inputs. A kernel function is a scalar product that does implicit mapping feature vectors from $\Re^d$ to a new space $\Re^n$. Kernels provide an efficient way to carry out these calculations when n is large or even infinite.

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

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

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

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.

However, in many NLP applications, the input data cannot fit to feature vectors in $\Re^d$ as the objects being modeled are strings, trees, graphs. (Collins and Duffy, 2001) proposed convolution kernels for various NLP structures. From then, kernel methods have attracted much interest due to their ability of implicitly exploring huge amounts of structural features automatically extracted from the original object representation. The kernels for structured natural language data, such as parse tree kernel (Collins and Duffy, 2001), string kernel (Lodhi et al., 2002), or word sequence kernel (Cancedda et al., 2003) are examples of the well-known convolution kernels used in many NLP applications.

2.5 Support Vector Machines (SVMs)

Support Vector Machines refer to a supervised machine learning technique based on the latest results of the statistical learning theory (Vapnik, 1998). SVMs provide efficiently training the linear machines in the kernel-induced feature spaces, while respecting the insights provided by the generalisation theory, and exploiting the optimisation theory. Thanks to the Karush-Kuhn-Tucker theorem, they can produce "sparse" dual representation of the hypothesis, resulting in extremely efficient algorithms. Moreover, due

to Mercer's conditions on the kernels, the corresponding optimization problems are convex and hence no local minima.

An important feature of these systems is that, while enforcing the learning biases suggested by the generalisation theory, they also produce "sparse" dual representations of the hypothesis, resulting in extremely efficient algorithms. This is due to the Karush-Kuhn-Tucker conditions, which hold for the solution and play a crucial role in the practical implementation and analysis of these machines. Another important feature of the Support Vector approach is that due to Mercer's conditions on the kernels the corresponding optimisation problems are convex and hence have no local minima. This fact, and the reduced number of non-zero parameters, mark a clear distinction between these system and other pattern recognition algorithms, such as neural networks.

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, 1995). SVMs is a binary classifier, but it can be easily extended to multi-class classifier, e.g. by means of the *one vs. rest* method (Rifkin and Poggio, 2002). One strong point of SVMs is the possibility to apply kernel methods (Robert Müller et al., 2001) to implicitly map data in a new space where the examples are *more easily* separable.

2.6 Tree Kernels

Tree kernels represent trees in terms of their sub-structures (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 SubSet Trees (SSTs) and the Partial Trees (PTs).

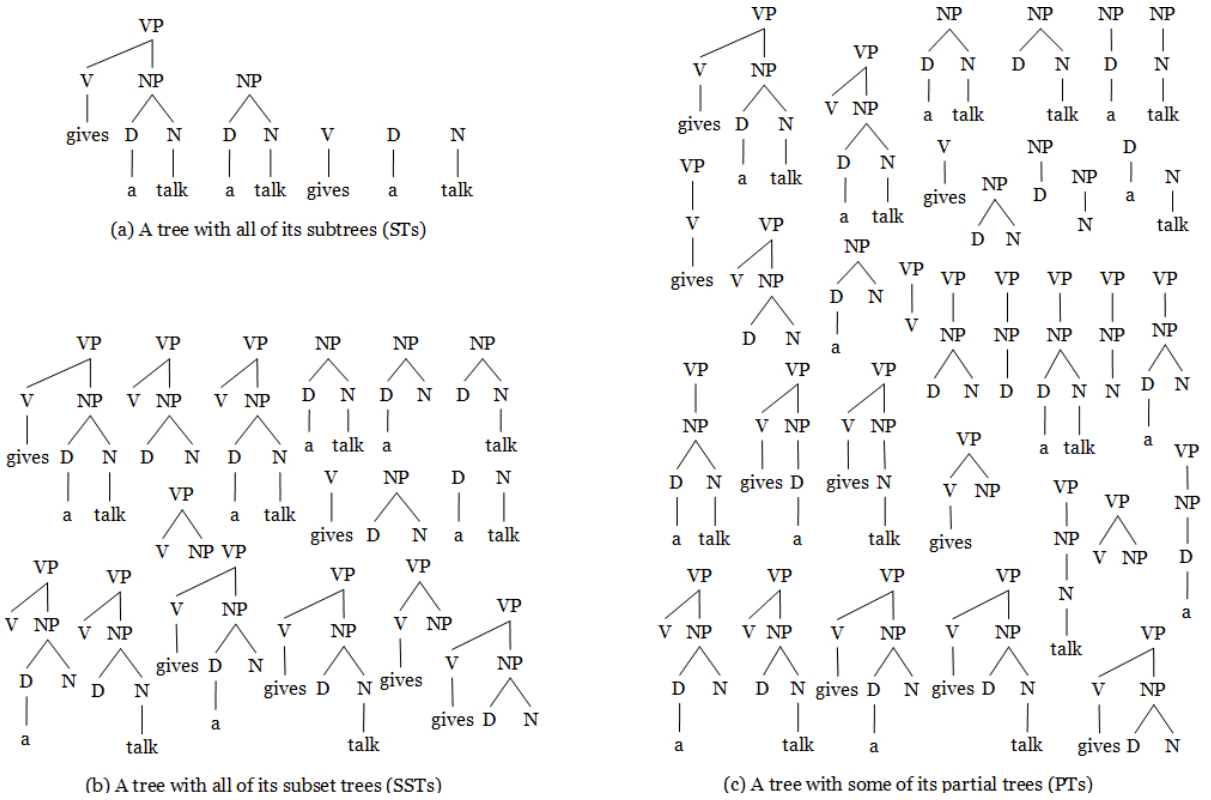


Figure 2.3: Three kinds of tree kernels.

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 2.3 shows the overall fragment set of the ST, SST and PT kernels for the syntactic parse tree of the sentence fragment: *gives a talk*. In general, the tree kernel results depend on the specific application but also on the study that suggest which tree kernel

type should be applied for the target task.

To see how to carry out efficient computation of this high dimensional representation, we follow the algorithm (Collins and Duffy, 2001). We enumerate all tree fragments that occur in the training data 1,...,n. Each tree is represented by an n dimensional vector where the i'th component count the number of occurences of the i'th tree fragment. Let us define the function $h_i(T)$ to be the number of occurrences of the i'th tree fragment in tree T, so that T is now represented as $h(T) = (h_1(T), h_2(T), ..., h_n(T))$.

We then examine the inner product between two trees T_1 and T_2 under this representation $K(T_1, T_2) = h(T_1) \cdot h(T_2)$. To compute K we first define the set of nodes in trees T_1 and T_2 as N_1 and N_2 respectively. We define the indicator function $I_i(n)$ to be 1 if sub-tree is seen rooted at node n and 0 otherwise. It follows that $h_i(T_1) = \sum_(n_1 \in N_1) I_i(n_1)$ and $h_i(T_2) = \sum_(n_2 \in N_2) I_i(n_2)$. With some simple algebra we have:

$$\begin{aligned} & h(T_1) \cdot h(T_2) = \textstyle\sum_i h_i(T_1) h_i(T_2) \\ & = \textstyle\sum_(n_1 \in N_1) \sum_(n_2 \in N_2) \sum_i I_i(n_1) I_i(n_2) \\ & = \textstyle\sum_(n_1 \in N_1) \sum_(n_2 \in N_2) C(n_1, n_2) \end{aligned}$$

where we define $C(n_1, n_2) = \sum_i I_i(n_1) I_i(n_2)$. Note that $C(n_1, n_2)$ can be computed in polynomial time, due to the following recursive definition:

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

$$C(n_1, n_2) = \prod^{n} c(n_1)_(j = 1)(1 + C(ch(n_1, j), ch(n_2, j))),$$

where $nc(n_1)$ is the number of children of n_1 in the tree; because the productions at n_1/n_2 are the same, we have $nc(n_1) = nc(n_2)$. The i'th child node of n_1 is $ch(n_1, i)$.

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

2.6.1 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 $\phi' = \phi \circ \phi_M$.

Chapter 3

Reranking Model for NER

In this chapter, we present a method of incorporating global features for named entity recognition based on reranking technique, combining the 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 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 effect. Lastly, the global features, when integrated in the baseline, yields much less improvement wrt. when integrated by reranking, proving the necessity of those features in the discrimination of hypotheses.

3.1 Introduction

Research in statistical natural language processing has shown the promise of reranking approach in enhancing the accuracy. 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. The approach is appealing in 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, 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). This work has exploited the disciminative property as one of the key criterion of the reranking algorithm.

Recent works have shown substantial improvement of the reranker in coupling with kernel methods (Collins and Duffy, 2001; Moschitti, 2004), 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.

3.2 Motivation

The reranking algorithms described in (Collins, 2002; Collins and Duffy, 2002) 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 the context of named entity recognition, as a hypothesis generated over one sentence provide a natural way to integrate semantic features, we can flexibly integrate those features into the post-process. We employ a tree kernel encoding NE tags of a sentence and combine them with a polynomial kernel, which efficiently exploits global features.

In this chapter, 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.

3.3 Datasets and Baseline

Statistical natural language learning always has to deal with the problem that is the possibility to crash on any new dataset. This happens with any approach of any NLP task. A robust NER system is expected to be well-adapted to multiple 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 are shown in tables 3.1 and 3.2.

3.3.1 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.

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 Loca-

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

Table 3.1: Statistics on the CoNLL English dataset

tion (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.

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

Table 3.2: Statistics on the EVALITA Italian dataset

3.3.2 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. 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.

3.3.3 Baseline Results

We trained the NER classifier on the two datasets. The Italian system participated in the EVALITA 2007 NER task (Nguyen et al., 2009a). In addition to the base CRF classifier we trained another classifier where we employed Support Vector Machines (SVMs). Although the second model performed worse than the base model, we reported the results for completeness.

The CoNLL English dataset

The EVALITA Italian dataset

Table 3.5 and 3.6 shows the final results on the Italian test set with CRFs and SVMs. We found that, with the same set of features, the accuracy of the NE classifiers trained with two models are rather competitive. Moreover, the NE classes GPE and PER reach quite good F1 values, while

[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	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 3.3: 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 3.4: SVMs results on the CoNLL Test set

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 3.5: 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 3.6: SVMs results on the EVALITA Test set

the recognition of ORG and LOC seems problematic. This is in line with previous results in which ORG seems to be the most difficult to learn. Lack of resource (the gazetteer for LOC is the least) may stand for this low accuracy of LOC class.

3.4 Reranking Method

3.4.1 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 where the first is the correct sequence:

H_1 *B-ORG I-ORG I-ORG I-ORG O O O O B-ORG O B-ORG O*

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 generated $< H_1, H_2 >$, $< H_1, H_3 >$,$< H_2, H_1 >$ 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).

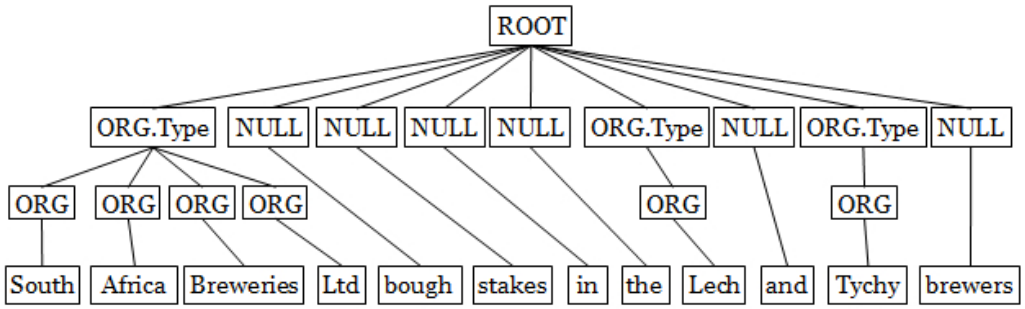
3.4.2 Representation of Tagged Sequences in Semantic Trees

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 3.1 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 3.1 for an example of the semantic tree for the first tagged sequence.

	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

Figure 3.1: Semantic structure of a candidate 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.

3.4.3 Global features

Mixed *n-grams* features

In previous works, some global features have been used (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 *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 un-nanotated 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 (Surdeanu et al., 2003; Bunescu and Mooney, 2005a), it is included 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

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

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 3.7 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-grams* patterns to learn and explicitly push the score of the correct sequence above the score of competing sequences.

Feature	Description
w_s-w_{s+1}-$\cdots$-w_e	Entity string
g_s-g_{s+1}-$\cdots$-g_e	The gazetteer feature within the entity
f_s-f_{s+1}-$\cdots$-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

Feature	Description
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;	Mixed trigrams of the words/frequent-tag features before/after the end of the entity

Feature	Description
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	

Table 3.7: Global features with the polynomial kernel for reranking.

3.4.4 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 2.3.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:

- $K(x, y) = L(x) \cdot L(y)$ 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.

3.5 Experiments

3.5.1 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. To the base CRF classifier presented in section 3.3.3, we enrich the feature set with the most frequent tag added in each token. The results with enriched features, trained with the baseline algorithm CRFs.

3.5.2 Reranking Results

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**	**88.25**	**88.16**
(Ratinov and Roth, 2009)	*N/A*	*N/A*	*90.57*

Table 3.8: Reranking results on the English test set.

Tables 3.8 and 3.9 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

Italian Test	**Pr**	**Re**	$\mathbf{F_1}$
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**
(Zanoli et al., 2009)	*84.07*	*80.02*	*82.00*

Table 3.9: Reranking results on the Italian test set.

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, 2009) 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.

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

Chapter 4

Combining Parsing Paradigms

In this chapter, we explore the use of innovative kernels based on syntactic and semantic structures for a target relation extraction task. Syntax is derived from constituent and dependency parse trees whereas semantics concerns to entity types and lexical sequences. We investigate the effectiveness of such representations in the automated relation extraction from texts. We process the above data by means of Support Vector Machines along with the syntactic tree, the partial tree and the word sequence kernels. Our study on the ACE 2004 corpus illustrates that the combination of the above kernels achieves high effectiveness and significantly improves the current state-of-the-art (Nguyen et al., 2009b).

4.1 Introduction

We study and invent diverse convolution and sequence kernels by providing several kernel combinations. To fully exploit the potential of dependency trees, in addition to the SST kernel, we applied the partial tree (PT) kernel proposed in (Moschitti, 2006), which is a general convolution tree kernel adaptable for dependency structures. We also investigate various sequence kernels (e.g. the word sequence kernel (WSK) (Cancedda et al., 2003)) by incorporating dependency structures into word sequences. These are

also enriched by including information from constituent parse trees. These form an integrated syntactic/semantic model of relation classification.

We conduct experiments on the standard ACE 2004 newswire and broadcast news domain. The results show that although some kernels are less effective than others, they exhibit properties that are complementary to each other. In particular, we found that relation extraction can benefit from increasing the feature space by combining kernels (with a simple summation) exploiting the two different parsing paradigms. Our experiments on RE show that the current composite kernel, which is constituent-based is more effective than those based on dependency trees and individual sequence kernel but at the same time their combinations, i.e. dependency plus constituent trees, improve the state-of-the-art in RE. More interestingly, also the combinations of various sequence kernels gain significant better performance than the current state-of-the-art (Zhang et al., 2005).

Overall, these results are interesting for the computational linguistics research since they show that the above two parsing paradigms provide different and important information for a semantic task such as RE. Regarding sequence-based kernels, the WSK gains better performance than previous sequence and dependency models for RE. A review of previous work on RE is described in section 2.2, our specific kernels for RE are described is section 4.3. The experiments and results then are presented in section 4.4, respectively.

4.2 Motivation

Several approaches have been proposed for automatically learning semantic relations from texts. Among others, there has been increased interest in the application of kernel methods (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Bunescu and Mooney, 2005b; Zhang et

al., 2005). Their main property is the ability of exploiting a huge amount of features without an explicit feature representation. This can be done by computing a kernel function between a pair of linguistic objects, as desribed in section 1.3.

Previous work on the use of kernels for RE has exploited some similarity measures over diverse features (Culotta and Sorensen, 2004; Zhang et al., 2005). However, the use of kernels over dependency trees (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a) showed much lower accuracy than feature-based methods (Zhao and Grishman, 2005). That may be due to one problem of these dependency kernels is that they do not exploit the overall structural aspects of dependency trees. A more effective solution is the application of convolution kernels to constituent parse trees (Zhang et al., 2006) but this is not satisfactory from a general perspective since dependency structures offer some unique advantages, which should be exploited by an appropriate kernel.

Therefore, studying convolution tree kernels for dependency trees is worthwhile also considering that, to the best of our knowledge, these models have not been previously used for relation extraction[1] task. Additionally, sequence kernels should be included in such global study since some of their forms have not been applied to RE.

4.3 Kernels for RE

In this section we describe the previous kernels based on constituent trees as well as new kernels based on diverse types of trees and sequences for relation extraction. As mentioned in the previous section, we can engineer kernels by combining tree and sequence kernels. Thus we focus on the problem

[1]The function defined on (Culotta and Sorensen, 2004), although on dependency trees, is not a convolution tree kernel.

to define structure embedding the desired syntactic relational information between two named entities (NEs).

4.3.1 Constituent and Dependency Structures

Syntactic parsing (or syntactic analysis) aims at identifying grammatical structures in a text. A parser thus captures the hidden hierarchy of the input text and processes it into a form suitable for further processing. There are two main paradigms for representing syntactic information: constituent and dependency parsing, which produces two different tree structures.

Constituent tree encodes structural properties of a sentence. The parse tree contains constituents, such as noun phrases (NP) and verb phrases (VP), as well as terminals/part-of-speech tags, such as determiners (DT) or nouns (NN). Figure 4.1.a shows the constituent tree of the sentence: *In Washington, U.S. officials are working overtime.*

Dependency tree encodes grammatical relations between words in a sentence with the words as nodes and dependency types as edges. An edge from a word to another represents a grammatical relation between these two. Every word in a dependency tree has exactly one parent except the root. Figure 4.1.b shows and example of the dependency tree of the previous sentence.

Given two NEs, such as *Washington* and *officials*, both the above trees can encode the syntactic dependencies between them. However, since each parse tree corresponds to a sentence, there may be more than two NEs and many relations expressed in a sentence. Thus, the use of the entire parse tree of the whole sentence holds two major drawbacks: first, it may be too computationally expensive for kernel calculation since the size of a complete parse tree may be very large (up to 300 nodes in the Penn Treebank (Marcus et al., 1993)); second, there is ambiguity on the target pairs of NEs, i.e. different NEs associated with different relations are described

by the same parse tree. Therefore, it is necessary to identify the portion of the parse tree that best represent the useful syntactic information.

Let e_1 and e_2 be two entity mentions in the same sentence such that they are in a relationship R. For the constituent parse tree, we used the path-enclosed tree (PET), which was firstly proposed in (Moschitti, 2004) for Semantic Role Labeling and then adapted by (Zhang et al., 2005) for relation extraction. It is the smallest common sub-tree including the two entities of a relation. The dashed frame in Figure 4.1.a surrounds PET associated with the two mentions, *officials* and *Washington*. Moreover, to improve the representation, two extra nodes T1-PER, denoting the type PERSON, and T2-LOC, denoting the type LOCATION, are added to the parse tree, above the two target NEs, respectively. In this example, the above PET is designed to capture the relation *Located-in* between the entities "officials" and "Washington" from the ACE corpus. Note that, a third NE, *U.S.*, is characterized by the node GPE (GeoPolitical Entity), where the absence of the prefix T1 or T2 before the NE type (i.e. GPE), denotes that the NE does not take part in the target relation.

In previous work, some dependency trees have been used (Bunescu and Mooney, 2005a) but the employed kernel just exploited the syntactic information concentrated in the path between e_1 and e_2. In contrast, we defined and studied three different dependency structures whose potential can be fully exploited by our convolution partial tree kernel:

- Dependency Words (DW) tree is similar to PET adapted for dependency tree constituted by simple words. We select the minimal subtree which includes e_1 and e_2, and we insert an extra node as father of the NEs, labeled with the NE category. For example, given the tree in Figure 4.1.b, we design the tree in Figure 4.1.c surrounded by the dashed frames, where T1-PER, T2-LOC and GPE are the extra nodes inserted as fathers of *Washington*, *soldier* and *U.S.*.

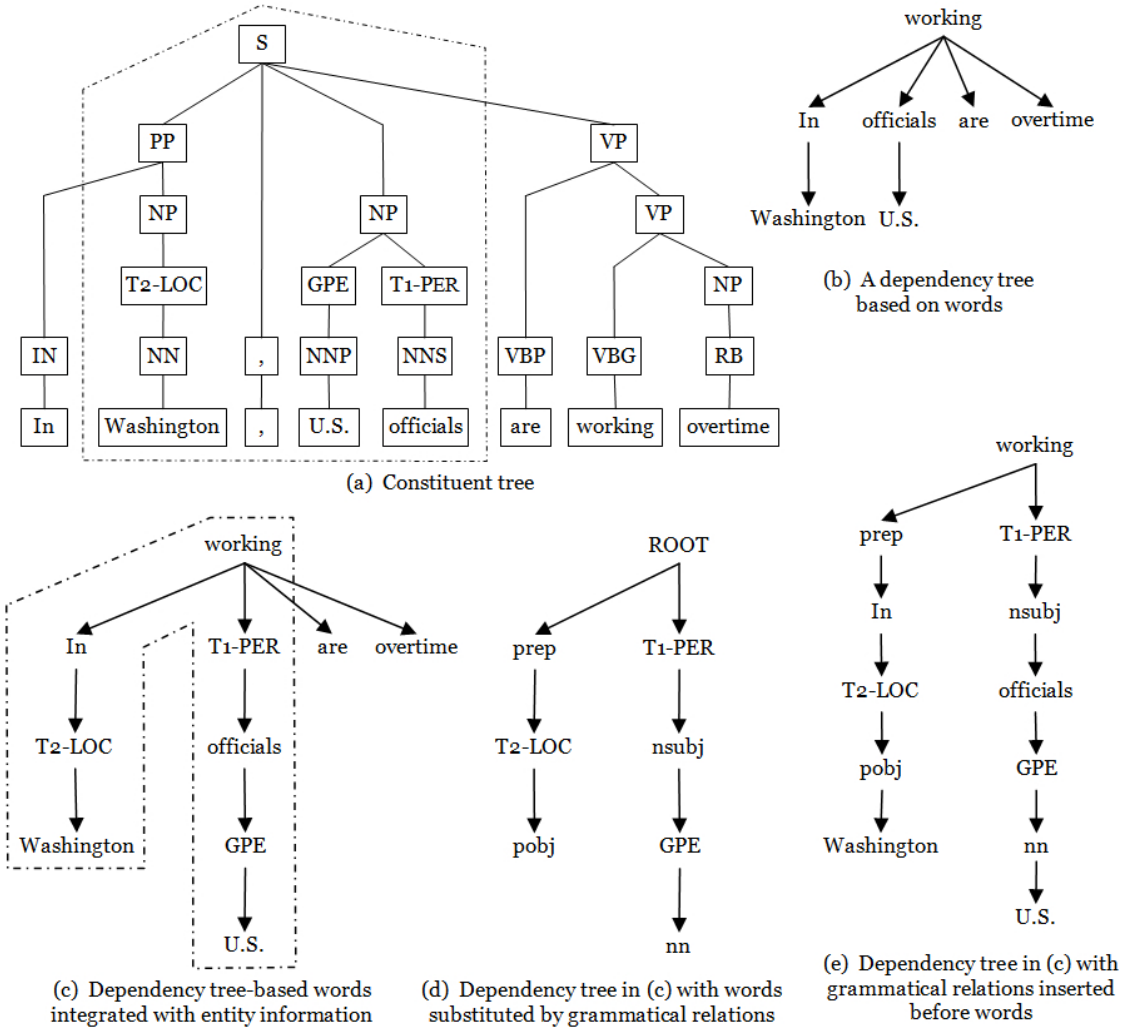


Figure 4.1: The constituent and dependency parse trees integrated with entity information

- Grammatical Relation (GR) tree, i.e. the DW tree in which words are replaced by their grammatical functions, e.g. *prep*, *pobj* and *nsubj*. For example, Figure 4.1.d, shows the GR tree for the previous relation: *In* is replaced by *prep* , *U.S.* by *nsubj* and so on.

- Grammatical Relation and Words (GRW) tree, words and grammatical functions are both used in the tree, where the latter are inserted as a father node of the former. For example, Figure 4.1.e, shows such tree for the previous relation.

4.3.2 Sequential Structures

Some sequence kernels have been used on dependency structures (Bunescu and Mooney, 2005b). These kernels just used lexical words with some syntactic information. To fully exploit syntactic and semantic information, we defined and studied six different sequences, which include features from constituent and dependency parse trees and NEs:

1. Sequence of terminals (lexical words) in the PET (SK_1), e.g.:
   T2-LOC Washington , U.S. T1-PER officials.

2. Sequence of part-of-speech (POS) tags in the PET (SK_2), i.e. the SK_1 in which words are replaced by their POS tags, e.g.:
   T2-LOC NN , NNP T1-PER NNS.

3. Sequence of grammatical relations in the PET (SK_3), i.e. the SK_1 in which words are replaced by their grammatical functions, e.g.:
   T2-LOC pobj , nn T1-PER nsubj.

4. Sequence of words in the DW (SK_4), e.g.:
   Washington T2-LOC In working T1-PER officials GPE U.S..

5. Sequence of grammatical relations in the GR (SK_5), i.e. the SK_4 in which words are replaced by their grammatical functions, e.g.:
   pobj T2-LOC prep ROOT T1-PER nsubj GPE nn.

6. Sequence of POS tags in the DW (SK_6), i.e. the SK_4 in which words are replaced by their POS tags, e.g.:
   NN T2-LOC IN VBP T1-PER NNS GPE NNP.

It is worth noting that the potential information contained in such sequences can be fully exploited by the word sequence kernel.

4.3.3 Combining Kernels

Given that syntactic information from different parse trees may have different impact on relation extraction (RE), the viable approach to study the role of dependency and constituent parsing is to experiment with different syntactic models and measuring the impact in terms of RE accuracy. For this purpose we compared the composite kernel described in (Zhang et al., 2006) with the partial tree kernels applied to DW, GR, and GRW and sequence kernels based on six sequences described above. The composite kernels include polynomial kernel applied to entity-related feature vector. The word sequence kernel (WSK) is always applied to sequential structures. The used kernels are described in more detail below.

Polynomial Kernel

The basic kernel between two named entities of the ACE documents is defined as:

$$K_P(R_1, R_2) = \sum_{i=1,2} K_E(R_1.E_i, R_2.E_i),$$

where R_1 and R_2 are two relation instances, E_i is the i^{th} entity of a relation instance. $K_E(\cdot, \cdot)$ is a kernel over entity features, i.e.:

$$K_E(E_1, E_2) = \sum_i C(E_1 \cdot f_i, E_2 \cdot f_i),$$

where f_i represents the i^{th} entity feature extracted from the two NEs.

For the ACE 2004, the features used include: entity headword, entity type, entity subtype, mention type, and LDC[2] mention type. The last four attributes are taken from the ACE corpus 2004. In ACE, each mention has a head annotation and an extent annotation.

Kernel Combinations

[2]Linguistic Data Consortium (LDC): http://www.ldc.upenn.edu/Projects/ACE/

1. Polynomial kernel plus a tree kernel:

$$CK_1 = \alpha \cdot K_P + (1 - \alpha) \cdot K_x,$$

where α is a coefficient to give more impact to K_P and K_x is either the partial tree kernel applied to one the possible dependency structures, DW, GR or GRW or the SST kernel applied to PET, described in the previous section.

2. Polynomial kernel plus constituent plus dependency tree kernels:

$$CK_2 = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{SST} + K_{PT})$$

where K_{SST} is the SST kernel and K_{PT} is the partial tree kernel (applied to the related structures as in point 1).

3. Constituent tree plus square of polynomial kernel and dependency tree kernel:

$$CK_3 = \alpha \cdot K_{SST} + (1 - \alpha) \cdot (K_P + K_{PT})^2$$

4. Dependency word tree plus grammatical relation tree kernels:

$$CK_4 = K_{PT-DW} + K_{PT-GR}$$

where K_{PT-DW} and K_{PT-GR} are the partial tree kernels applied to dependency structures DW and GR.

5. Polynomial kernel plus dependency word plus grammatical relation tree kernels:

$$CK_5 = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{PT-DW} + K_{PT-GR})$$

Some preliminary experiments on a validation set showed that the second, the fourth and the fifth combinations yield the best performance with $\alpha = 0.4$ while the first and the third combinations yield the best performance with $\alpha = 0.23$.

Regarding WSK, the following combinations are applied:

1. $SK_3 + SK_4$

2. $SK_3 + SK_6$

3. $SSK = \sum_{i=1,..,6} SK_i$

4. $K_{SST} + SSK$

5. $CSK = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{SST} + SSK)$

Preliminary experiments showed that the last combination yields the best performance with $\alpha = 0.23$.

We used a polynomial expansion to explore the bi-gram features of i) the first and the second entity participating in the relation, ii) grammatical relations which replace words in the dependency tree. Since the kernel function set is closed under normalization, polynomial expansion and linear combination (Schölkopf and Smola, 2001), all the illustrated composite kernels are also proper kernels.

4.4 Experiments

Our experiments aim at investigating the effectiveness of convolution kernels adapted to syntactic parse trees and various sequence kernels for the RE task. For this purpose, we use the subset and partial tree kernel over different kinds of trees, namely constituent and dependency syntactic parse trees. Diverse sequences are applied individually and in combination together. We consider our task of relation extraction as a classification problem where categories are relation types. All pairs of entity mentions in the same sentence are taken to generate potential relations, which will be processed as positive and negative examples.

4.4.1 Experimental setup

We use the newswire and broadcast news domain in the English portion of the ACE 2004 corpus provided by LDC. This data portion includes 348 documents and 4400 relation instances. It defines seven entity types and seven relation types. Every relation is assigned one of the seven types: Physical, Person/Social, Employment/Membership/-Subsidiary, Agent-Artifact, PER/ORG Affiliation, GPE Affiliation, and Discourse. For sake of space, we do not explain these relationships here, nevertheless, they are explicitly described in the ACE document guidelines. There are 4400 positive and 38,696 negative examples when generating pairs of entity mentions as potential relations.

Documents are parsed using Stanford Parser (Klein and Manning, 2003) to produce parse trees. Potential relations are generated by iterating all pairs of entity mentions in the same sentence. Entity information, namely entity type, is integrated into parse trees. To train and test our binary relation classifier, we used SVMs. Here, relation detection is formulated as a multiclass classification problem. The *one vs. rest* strategy is employed by selecting the instance with largest margin as the final answer. For experimentation, we use 5-fold cross-validation with the Tree Kernel Tools (Moschitti, 2004) (available at http://disi.unitn.it/~moschitt/Tree-Kernel.htm).

4.4.2 Results

In this section, we report the results of different kernels setup over constituent (CT) and dependency (DP) parse trees and sequences taken from these parse trees. The tree kernel (TK), composite kernel (CK_1, CK_2, CK_3, CK_4, and CK_5 corresponding to five combination types in Section 4.3.3) were employed over these two syntactic trees. For the tree kernel, we

apply the SST kernel for the path-enclosed tree (PET) of the constituent tree and the PT kernel for three kinds of dependency tree DW, GR, and GRW, described in the previous section. The two composite kernels CK_2 and CK_3 are applied over both two parse trees. The word sequence kernels are applied over six sequences $SK_1, SK_2, SK_3, SK_4, SK_5$, and SK_6 (described in Section 4.3.3).

The results are shown in Table 4.1 and Table 4.2. In the first table, the first column indicates the structure used in the combination shown in the second column, e.g. PET associated with CK_1 means that the SST kernel is applied on PET (a portion of the constituent tree) and combined with the CK_1 schema whereas PET and GR associated with CK_5 means that SST kernel is applied to PET and PT kernel is applied to GR in CK_5. The remaining three columns report Precision, Recall and F1 measure. The interpretation of the second table is more immediate since the only tree kernel involved is the SST kernel applied to PET and combined by means of CK_1.

Parse Tree	Kernel	P	R	F
PET	**CK_1**	**69.5**	**68.3**	**68.9**
DW	CK_1	53.2	59.7	56.3
GR	CK_1	58.8	61.7	60.2
GRW	CK_1	56.1	61.2	58.5
DW and GR	CK_5	59.7	64.1	61.8
PET and GR	**CK_2**	**70.7**	**69.0**	**69.8**
	CK_3	**70.8**	**70.2**	**70.5**

Table 4.1: Results on the ACE 2004 with six structures.

We note that: first, the dependency kernels, i.e. the results on the rows from 3 to 6 are below the composite kernel CK_1, i.e. 68.9. This is the state-of-the-art in RE, designed by (Zhang et al., 2006), where our implementation provides a slightly smaller result than the original version

Kernel	P	R	F
$\mathbf{CK_1}$	**69.5**	**68.3**	**68.9**
SK_1	72.0	52.8	61.0
SK_2	61.7	60.0	60.8
SK_3	62.6	60.7	61.6
SK_4	73.1	50.3	59.7
SK_5	59.0	60.7	59.8
SK_6	57.7	61.8	59.7
$\mathbf{SK_3 + SK_4}$	**75.0**	**63.4**	**68.8**
$SK_3 + SK_6$	66.8	65.1	65.9
$\mathbf{SSK = \sum_i SK_i}$	**73.8**	**66.2**	**69.8**
CSK	**75.6**	**66.6**	**70.8**
$\mathbf{CK_1 + SSK}$	**76.6**	**67.0**	**71.5**
(Zhou et al., 2007) (CK_1 with Heuristics)	*82.2*	*70.2*	75.8

Table 4.2: Results on the ACE 2004 with different kernel setups.

(i.e. an F1 of about 72 using a different syntactic parser).

Second, CK_1 improves to 70.5, when the contribution of PT kernel applied to GR (dependency tree built using grammatical relations) is added. This suggests that dependency structures are effectively exploited by PT kernel and that such information is somewhat complementary to constituent trees.

Third, in the second table, the model $CK_1 + SSK$, which adds to CK_1 the contribution of diverse sequence kernels, outperforms the state-of-the-art by 2.6%. This suggests that the sequential information encoded by several sequence kernels can better represents the dependency information.

Finally, we also report in the last row (in italic) the superior RE result by (Zhou et al., 2007). However, to achieve this outcome the authors used the composite kernel CK_1 with several heuristics to define an effective portion of constituent trees. Such heuristics expand the tree and remove unnecessary information allowing a higher improvement on RE. They are

tuned on the target RE task so although the result is impressive, we cannot use it to compare with pure automatic learning approaches, such us our models.

Chapter 5

Relation Ordering Strategies

In this chapter, we study the variation in performance of RE systems with the use of entity features at coarse and fine-grained levels. In the literature, state-of-the-art RE models based on ACE often employ entity attributes as features in the learning machine. Such attributes include entity types (Person, Organization, Location), entity subtypes, mention types/-subtypes, and headword as a guide to drive relation extraction. However, we show that, the use of such features depends on the order of the two entities participating in a relation. More importantly, the deeper categories of the entities we use, the higher the performance of RE.

5.1 Motivation

5.1.1 Coarse and Fine-grained Features

In the ACE 2004 program, entities are limited to the 7 types and 42 subtypes, whereas relations are assigned to the 7 syntactic relation class types. A mention is a textual references to an entity. Each mention has some attributes like type, LDCtype, role, or reference. Table 5.1 shows the categories in the ACE 2004 corpus with their quantities, 5.2 and 5.3 describe entity/relation types with their descriptions and examples.

Category name	Number
News Categories	8
Entity Types	7
Entity Subtypes	42
Entity Classes	5
Relation Types	7
Relation Subtypes	22
Relation LDCLexicalConditions	6
Mention Types	4
Mention LDC Types	28
Mention Roles	5
Mention References	2

Table 5.1: Detailed statistics on the ACE 2004 corpus.

Entity type	Entity subtype
Person (PER)	
Organization (ORG)	Government, Commercial, Educational, Non-Profit, Other
Facility (FAC)	Building, Subarea-Building, Bounded-Area, Conduit, Path, Barrier, Plant, Other
Location (LOC)	Address, Boundary, Celestial, Water-Body, Land-Region -Natural, Region-Local, Region-Subnational, Region-National, Region-International
GPE (Geo-political)	Continent, Nation, State-or-Province, County-or-District, Population-Center, Other
Vehicle (VEH)	Land, Air, Water, Subarea-Vehicle, Other
Weapon (WEA)	Blunt, Exploding, Sharp, Chemical, Biological, Shooting, Projectile, Nuclear, Other

Table 5.2: Entity types and subtypes defined in ACE 2004.

5.1.2 Statistics

In this section, we report the statistical distribution of possible combinations of entity/mention types/subtypes. The statistics are shown in

Relation type	Example
Physical (PHYS)	*a military base in Germany* [PHYS ("a military base", "Germany")]
Person/Social (PER-SOC)	*his lawyer* [PER-SOC ("his", "his lawyer")]
Employment/Membership /Subsidiary (EMP-ORG)	*George Bush, the US president* [EMP-ORG ("the US president", "US")]
Agent-Artifact (ART)	*My house is in West Philadelphia* [ART ("my", "my house")]
PER/ORG Affiliation (Other-AFF)	*Cuban-American people* [OTHER-AFF ("people", "Cuban-American")]
GPE Affiliation (GPE-AFF)	*U.S. businessman Edmond Pope* [GPE-AFF ("U.S. businessman", "U.S")]
Discourse (DISC)	*Many of these people* [DISC ("Many of these people", "these people")]

Table 5.3: Relation types and their description as defined in ACE 2004.

table 5.4.

Combination type	**Number**	**Average**
Entity type	28	3.6071
Entity type/subtype	486	1.7058
Entity type/subtype, Mention type	2246	1.3313
Entity type/subtype, Mention type/LDCtype	3514	1.2555

Table 5.4: Statistics on the average number of relation types for each combination of categories/subcategories of entities and mentions. The deeper level of the category, the lower number of relation types, corresponding to the higher results in relation classification.

5.1.3 Relation Ordering Strategy

Kernel approaches for RE (Zhou et al., 2005; Bunescu and Mooney, 2005a; Zhang et al., 2005; Zhang et al., 2006; Nguyen et al., 2009b) have employed

most of the entity and mention features from ACE 2004, including entity type/subtype, mention type/LDCtype/role/reference/headword. However, each relation takes two different entity arguments. Therefore, it takes into account the relation order, which is established by some discrete algorithm based on the two entities participating in a relation.

We note that, the relation order does participate in all kernel settings. As shown in the figure 5.1, the order is expressed by "T1" and "T2" in combination with entity types. The tree kernel CT based on SST will find all the *subtrees* that match either "T1-PER" and "T2-LOC". Thus, the relation R_1 with "T1-PER" and "T2-LOC" differs from the relation R_2 with "T2-PER" and "T1-LOC". Similarly, the entity kernel ENK employs the function $K_E(E_1, E_2) = \sum_i C(E_1 \cdot f_i, E_2 \cdot f_i)$ that counts the number of *feature values* in common between entity E_i of two relations. Considering three relations: $R_1 = (E_1, E_2)$, $R_2 = (E_2, E_3)$, $R_3 = (E_3, E_2)$, obviously R_1 is expected to have more values shared with R_3 than with R_2, since they have at least entity E_2 in common.

5.2 Experiments

5.2.1 Kernel Setting

We use the four kernels in the section 4.3.3: the entity kernel ENK, the tree kernel CT, the composite kernel CK_1, and the hybrid kernel CSK.

The entity kernel

The basic kernel between two named entities of the ACE documents is defined as:

$$ENK = K_P(R_1, R_2) = \sum_{i=1,2} K_E(R_1.E_i, R_2.E_i),$$

where R_1 and R_2 are two relation instances, E_i is the i^{th} entity of a relation

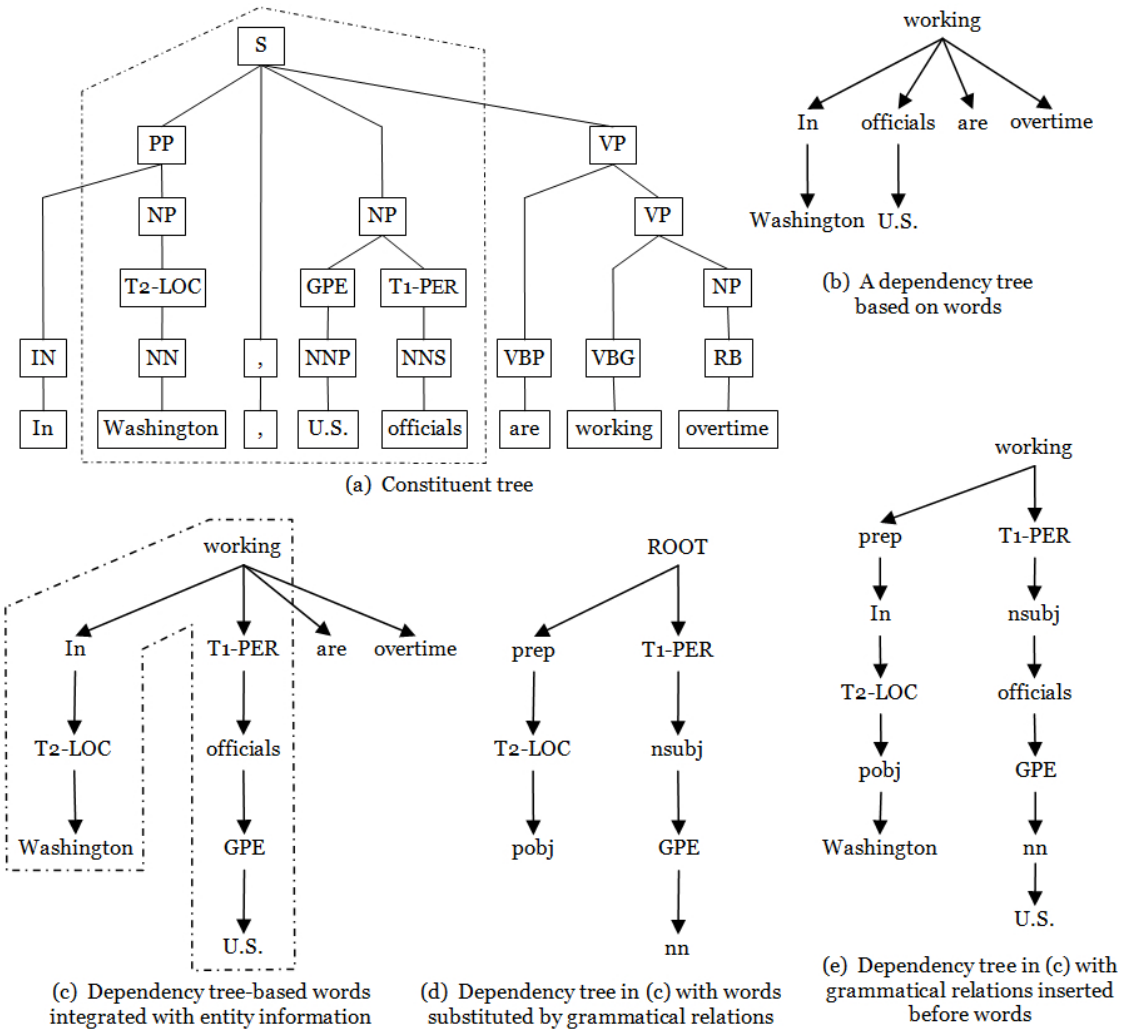


Figure 5.1: The constituent and dependency parse trees integrated with entity information

instance. $K_E(\cdot,\cdot)$ is a kernel over entity features, i.e.:

$$K_E(E_1, E_2) = \sum_i C(E_1 \cdot f_i, E_2 \cdot f_i),$$

where f_i represents the i^{th} entity feature extracted from the two NEs.

For the ACE 2004, the features used include: entity headword, entity type, entity subtype, mention type, and LDC[1] mention type. The last four attributes are taken from the ACE corpus 2004. In ACE, each mention

[1]Linguistic Data Consortium (LDC): http://www.ldc.upenn.edu/Projects/ACE/

has a head annotation and an extent annotation.

The composite kernel

The composite kernel is formulated by the tree kernel plus the entity kernel:

$$CK_1 = \alpha \cdot CT + (1 - \alpha) \cdot ENK,$$

where CT is the SST tree kernel applied to PET and α is a coefficient to give more impact to CT or ENK, as described in the section 4.3.1.

Some preliminary experiments on a validation set showed that this combination yields the best performance with $\alpha = 0.23$.

The hybrid kernel

The hybrid kernel is formulated by the tree kernel plus the six sequence kernels that are derived from a combination of constituent/dependency parses:

$$CSK = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{SST} + SSK)$$

Preliminary experiments showed that this combination yields the best performance with $\alpha = 0.23$.

5.2.2 Ordering Strategy

In our approach, the identification of relation order is performed using a number of entity/mention attributes, which comprise seven attributes in total. Based on a subset of those attributes, we performed experiments with five algorithms from 5.2.1 to 5.2.5, to define the relation order. We also present our previous results (Nguyen et al., 2009b) where the order is establlished using three features (5.2.6).

Algorithm 5.2.1: DEFINE_RELATION_ORDER_1()

```
order ← true
E1, E2 : The first and second entity
M1, M2 : Mention of entity E1 and E2
type1 ← mention headword of M1
type2 ← mention headword of M2
if type1 < type2
  then order ← true
  else order ← false
return (order)
```

Algorithm 5.2.2: DEFINE_RELATION_ORDER_2()

```
order ← true
E1, E2 : The first and second entity
M1, M2 : Mention of entity E1 and E2
type1 ← entity type of E1
type2 ← entity type of E2
if type1 < type2
  then order ← true
  else
    do { if type1 > type2
         then order ← false
         else
           do { type1 ← mention headword of M1
                type2 ← mention headword of M2
                if type1 < type2
                  then order ← true
                  else order ← false
return (order)
```

5.2.3 Experimental Setup

We use the same dataset and softwares as described in section 4.4.1. However, for experimentation, due to the expensiveness of various settings and algorithms, we cannot do 5-fold cross-validation, instead, in the whole ACE

Algorithm 5.2.3: DEFINE_RELATION_ORDER_3()

```
order ← true
E1, E2 : The first and second entity
M1, M2 : Mention of entity E1 and E2
type1 ← entity type of E1
type2 ← entity type of E2
if type1 < type2
  then order ← true
  else
    do { if type1 > type2
         then order ← false
         else
           do { type1 ← entity subtype of E1
                type2 ← entity subtype of E2
                if type1 < type2
                  then order ← true
                  else
                    do { if type1 > type2
                         then order ← false
                         else
                           do { type1 ← mention headword of M1
                                type2 ← mention headword of M2
                                do { if type1 < type2
                                     then order ← true
                                     else order ← false
return (order)
```

2004 corpus that contains 348 documents, we split the data in a training set that contains 280 documents and a test set that contains the remaining 68 documents.

5.2.4 Results

In this section, we report the results of different kernels setupon the ACE 2004 corpus. The results are shown in tables from 5.6 to 5.13. Our previous results (Nguyen et al., 2009b) is also shown in table 5.15 where we used

Algorithm 5.2.4: DEFINE_RELATION_ORDER_4()

```
order ← true
E1, E2 : The first and second entity
M1, M2 : Mention of entity E1 and E2
type1 ← entity type of E1
type2 ← entity type of E2
if type1 < type2
  then order ← true
  else
    do { if type1 > type2
         then order ← false
         else
           do { type1 ← entity subtype of E1
                type2 ← entity subtype of E2
                if type1 < type2
                  then order ← true
                  else
                    do { if type1 > type2
                         then order ← false
                         else
                           do { type1 ← mention type of M1
                                type2 ← mention type of M2
                                if type1 < type2
                                  then order ← true
                                  else
                                    do { if type1 > type2
                                         then order ← false
                                         else
                                           do { type1 ← mention headword of M1
                                                type2 ← mention headword of M2
                                                do { if type1 < type2
                                                     then order ← true
                                                     else order ← false
return (order)
```

Algorithm 5.2.5: DEFINE_RELATION_ORDER_5()

```
order ← true
E1, E2 : The first and second entity
M1, M2 : Mention of entity E1 and E2
type1 ← entity type of E1
type2 ← entity type of E2
if type1 < type2
  then order ← true
  else
    do { if type1 > type2
         then order ← false
         else
           do { type1 ← entity subtype of E1
                type2 ← entity subtype of E2
                if type1 < type2
                  then order ← true
                  else
                    do { if type1 > type2
                         then order ← false
                         else
                           do { type1 ← mention type of M1
                                type2 ← mention type of M2
                                if type1 < type2
                                  then order ← true
                                  else
                                    do { if type1 > type2
                                         then order ← false
                                         else
                                           do { type1 ← mention subtype of M1
                                                type2 ← mention subtype of M2
                                                if type1 < type2
                                                  then order ← true
                                                  else
                                                    do { if type1 > type2
                                                         then order ← false
                                                         else
                                                           do { type1 ← mention headword of M1
                                                                type2 ← mention headword of M2
                                                                if type1 < type2
                                                                  then order ← true
                                                                  else order ← false }
                                                }
                                         }
                                }
                         }
                }
         }
return (order)
```

Algorithm 5.2.6: DEFINE_RELATION_ORDER_OLD()

```
order ← true
E1, E2 : The first and second entity
M1, M2 : Mention of entity E1 and E2
type1 ← entity type of E1
type2 ← entity type of E2
if type1 < type2
  then order ← true
  else
    do { if type1 > type2
           then order ← false
           else
             do { type1 ← mention type of M1
                  type2 ← mention type of M2
                  if type1 < type2
                    then order ← true
                    else
                      do { if type1 > type2
                             then order ← false
                             else
                               do { type1 ← mention headword of M1
                                    type2 ← mention headword of M2
                                    if type1 < type2
                                      then order ← true
                                      else order ← false
return (order)
```

seven features in the entity kernel but the relation order is established based on only three features. The corresponding between results and algorithms are shown in table 5.5.

Table results	**Relation order strategy**	**Features in the ordering algorithm**	**Features in the entity kernel**
Table 5.6	Algorithm 5.2.1	1	1
Table 5.7	Algorithm 5.2.2	2	2
Table 5.8	Algorithm 5.2.3	3	3
Table 5.9	Algorithm 5.2.4	4	4
Table 5.10	Algorithm 5.2.5	5	5
Table 5.11		5	6
Table 5.14		5	7
Table 5.12	Algorithm 5.2.1	1	7
Table 5.13	Algorithm 5.2.2	2	7
Table 5.15	Algorithm 5.2.6	3	5

Table 5.5: Correspondence between results and different relation order strategies.

Kernel	**Pr**	**Re**	**F_1**
ENK	36.02	30.05	32.76
CT	69.83	63.38	66.45
$CK1$	78.03	59.47	67.50
CSK	80.87	69.48	74.75

Table 5.6: Results with one feature: mention headword; the order is established by using one feature.

Kernel	**Pr**	**Re**	**F_1**
ENK	76.97	65.88	70.99
CT	71.21	66.98	69.03
$CK1$	89.17	77.31	82.82
CSK	93.22	79.66	85.91

Table 5.7: Results with two features: entity type and mention headword; the order is established by using two features.

The results show that:

First, the hybrid kernel CSK outperforms the composite kernel $CK1$ with less than four features, (tables 5.6, 5.7, and 5.8). It demonstrates that the sequence kernels derived from constituent/dependency parses are still

Kernel	Pr	Re	F_1
ENK	75.49	66.98	70.98
CT	71.21	66.98	69.03
$CK1$	90.57	79.66	84.76
CSK	93.32	80.91	86.67

Table 5.8: Results with three features: entity type/subtype, and mention headword; the order is established by using three features.

Kernel	Pr	Re	F_1
ENK	81.90	75.74	78.70
CT	71.99	66.35	69.06
$CK1$	96.03	87.01	91.30
CSK	96.36	82.79	89.06

Table 5.9: Results with four features: entity type/subtype, and mention type-/headword; the order is established by using four features.

Kernel	Pr	Re	F_1
ENK	81.43	76.84	79.07
CT	71.99	66.35	69.06
$CK1$	97.02	86.70	91.57
CSK	96.01	82.94	89.00

Table 5.10: Results with five features: entity type/subtype, and mention type-/LDCtype/headword; the order is established by using five features.

Kernel	Pr	Re	F_1
ENK	81.19	77.00	79.04
CT	71.99	66.35	69.06
$CK1$	96.86	86.85	91.58
CSK	96.01	82.79	88.91

Table 5.11: Results with six features: entity type/subtype, and mention type-/LDCtype/role/headword; the order is established by using five features: entity type/subtype, mention type/LDCtype/-headword.

robust as far as less refined features are used.

Second, when more than four features are used in both the feature space and relation order algorithm, as shown in table 5.9, 5.10, 5.11, and 5.14, the performance in all kernel settings are almost competitive. It proves that the four features: entity type/subtype, mention type/headword are the most relevant for the relation extraction task.

Finally, the last four tables 5.12, 5.13, 5.14, and 5.15 clearly demonstrate the effect of relation ordering technique. With the refinement from one to

Kernel	Pr	Re	F_1
ENK	40.44	59.94	48.30
CT	69.78	63.22	66.34
$CK1$	80.75	77.46	79.07
CSK	85.04	75.59	80.03

Table 5.12: Results with seven features: entity type/subtype, and mention type-/LDCtype/role/reference/headword; the order is established by using one feature: mention headword.

Kernel	Pr	Re	F_1
ENK	73.68	72.30	72.99
CT	71.21	66.98	69.03
$CK1$	92.62	82.47	87.25
CSK	94.16	80.75	86.94

Table 5.13: Results with seven features: entity type/subtype, and mention type-/LDCtype/role/reference/headword; the order is established by using two features: entity type and mention headword.

Kernel	Pr	Re	F_1
ENK	81.35	77.15	79.20
CT	71.99	66.35	69.06
$CK1$	96.35	86.85	91.36
CSK	96.20	83.10	89.17

Table 5.14: Results with seven features: entity type/subtype, and mention type-/LDCtype/role/reference/headword; the order is established by using five features: entity type/subtype, mention type/LDCtype/headword.

Kernel	Pr	Re	F_1
ENK	28.80	39.75	33.40
CT	71.86	66.35	69.00
CK1	71.63	69.95	70.78
CSK	76.30	71.05	73.58

Table 5.15: Previous results with five features: entity type/subtype, and mention type/LDCtype/headword; the order is established by using three features: entity type, mention type, and mention headword.

five features in the ordering algorithm, the results increase 12.29% absolute points in the composite kernel CSK (which gains the highest performance).

Chapter 6

Large Scale IE

In this chapter, we model distant supervision (DS) based on Wikipedia and YAGO for Relation Extraction (RE) at sentence level (i.e. as defined in ACE). More specifically, we use the relations defined in external repositories such as YAGO and extract training data from Freebase documents. From these, we also derive training data for our named entity recognizer, used to build end-to-end RE systems. These are made robust and flexible by the use of kernels applied to both dependency and constituency syntactic structures. The experiments show that DS data (i) produces a meaningful F1 of 74.29% on our Wikipedia test set and (ii) improves RE from ACE data (Nguyen and Moschitti, 2011). Additionally, our end-to-end experiments demonstrated that our extractors are generally applicable.

6.1 Motivation and Related Work

The extraction of relational data from text has drawn popularity for its potential application in a broad range of task. Especially with the paradigm of Wikipedia and Web search, an interesting idea would be automated extraction of relational facts, or world knowledge from the Web (Yates, 2009). To identify semantic relations using machine learning, three learning settings have mainly been applied, namely supervised methods (Zelenko et al.,

2002; Culotta and Sorensen, 2004; Kambhatla, 2004; Zhou et al., 2005), semi supervised methods (Brin, 1998; Agichtein and Gravano, 2000), and unsupervised method (Hasegawa et al., 2004; Etzioni et al., 2008).

Early work on Relation Extraction has mostly employed kernel-based approaches (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Zhang et al., 2005). Kernels on parse trees were pioneered by (Collins and Duffy, 2001). (Culotta and Sorensen, 2004) extended this work to calculate kernels between augmented dependency trees. Recent literature has shown that an efficient and appropriate kernel can be used to solve the RE problem, exploiting the advantages of the underlying structures (Nguyen et al., 2009b; Zhang et al., 2006).

Traditional relation classifiers use only labeled data to train. However, labeled instances are expensive, or time consuming to obtain, as they require efforts of experienced human annotators. Meanwhile, unlabeled data may be relatively easy to collect, but there exist very few ways to use them. (Bunescu and Mooney, 2007) proposes a way of using a handful training set for RE. However, that works was applied to very few relation types that are impractical (two datasets for two relations). Distant supervised learning (Mintz et al., 2009) addresses this problem by using large amount of data to build classifiers. By using a large amount of unlabeled data and more relation instances, it can obviate problems with noisy features.

Although several approaches have been proposed to address the scarcity of labeled data (Bunescu and Mooney, 2007; Mintz et al., 2009; Riedel et al., 2010), only (Riedel et al., 2010) has tried to adapt those proposed learning algorithms to another text domain. However, none of them has ever tried to transfer the learning model another label set on another domain. It is widely known that the adaptation on new domains of statistical models would probably lead to a drop in performance. The drop would be even more to transfer the learning model to a different label set. Obvi-

ously, the most important challenge of an extraction system is that it must handle arbitrary domains and types of knowledge with little or no human involvement.

Perhaps most similar to our distant supervision algorithm is the effective method of (Hoffmann et al., 2010) who extract relations from a Wikipedia page by using supervision from the page's infobox. (Riedel et al., 2010) also tries to improve the distant supervision assumption with constraint-driven semi-supervision. In contrast to the former that only use the Infobox related page, our approach allows obtaining training data for relations defined in different sources. While the latter has targeted only three frequent relation types, our algorithm allow us to extract many more relations different documents and different domains.

6.2 Distant Supervision

Relation Extraction (RE) from text as defined in ACE (Doddington et al., 2004) concerns the extraction of relationships between two entities. This is typically carried out by applying supervised learning, e.g. (Zelenko et al., 2002), to hand-labeled corpora. Although, the resulting models are far more accurate than unsupervised approaches, they require labeled data and tend to be domain-dependent as different domains involve different relations.

The drawbacks above can be alleviated by applying a form of weakly supervision, specifically named distant supervision (DS), using Wikipedia data (Etzioni et al., 2008; Mintz et al., 2009; Hoffmann et al., 2010). The main idea is to exploit (i) relation repositories, e.g. the *Infobox*, x, of Wikipedia to define a set of relation types $RT(x)$ and (ii) the text of the page associated with x to produce the training sentences, which are supposed to express instances of $RT(x)$.

Previous work has applied DS to RE at *corpus level*, e.g., (Etzioni et al., 2008; Mintz et al., 2009): relation extractors are (i) learned using such not completely accurate data and (ii) applied to extract relation instances from the whole corpus. The multiple pieces of evidence for each relation instance are then exploited to recover from errors of the automatic extractors. Additionally, a recent approach, i.e., (Hoffmann et al., 2010), has shown that DS can be also applied at level of Wikipedia article: given a target *Infobox* template, all its attributes[1] can be extracted from a given document matching such template.

In contrast, sentence-level RE (SLRE) has been only modeled with the traditional supervised approach, e.g., using the data manually annotated in ACE (Culotta and Sorensen, 2004; Kambhatla, 2004; Zhou et al., 2005; Bunescu and Mooney, 2005a; Zhang et al., 2005; Zhang et al., 2006; Bunescu and Mooney, 2007; Nguyen et al., 2009b). The resulting extractors are very valuable as they find rare relation instances that might be expressed in only one document. For example, the relation *President(Barrack Obama, United States*) can be extracted from thousands of documents thus there is a large chance of acquiring it. In contrast, *President(Eneko Agirre, SIGLEX)* is probably expressed in very few documents (if not just one sentence), increasing the complexity for obtaining it.

We propose several enhancements of SLRE: first, the use of DS, where the relation providers are external repositories, e.g., YAGO (Suchanek et al., 2007), and the training instances are gathered from Freebase (Metaweb Technologies, 2010). These allow for potentially obtaining larger training data and many more relations, defined in different sources.

Second, we adapt state-of-the-art models for ACE RE, based on Support Vector Machines (SVMs) and kernel methods (KM), to Wikipedia. We used tree and sequence kernels that can exploit structural information and

[1]This is a simpler tasks as one of the two entity is fixed.

interdependencies among possible labels. The comparative experiments show that our models are flexible and robust to Web documents as we achieve the interesting F1 of 74.29% on 52 YAGO relations. To give a very rough idea of the importance of the results, the document-level attribute extraction based on DS showed an F1 of 61% (Hoffmann et al., 2010).

Third, to provide strong evidence of the quality of our SLRE, we manually mapped relations from YAGO to ACE based on their descriptions. Then, we designed a joint RE model combining DS and ACE data and tested it on ACE annotations (thus according to expert linguistic annotators). The improvement of 2.29 percent points (76.23%-73.94%) shows that our DS data is consistent and valuable.

Finally, since our aim is to produce RE for real-world applications, we experimented with end-to-end systems. For this purpose, we also exploit Freebase for creating training data for our robust Named Entity Recognizer (NER). Consequently, our RE system is applicable to any document/sentence, i.e. another major improvement on previous work, which, to our knowledge, does not show experiments on end-to-end SLRE. The satisfactory F1 of 67% for the 52 YAGO relations suggests that our technology can be applied to real scenarios. This convinced us to make available: (i) the training set relations (68,429 instances), (ii) the small manual validated set (2,601 instances) and (iii) the mapping between ACE and YAGO relations.

6.3 Methodology for Data Creation

The resources we used to implement DS are YAGO, a large knowledge base of entities and relations, and Freebase, a collection of Wikipedia news articles. Our procedure uses entities and facts from YAGO to provide relation instances. For each pair of entities that appears in some YAGO relations, we retrieve all the sentences of the Freebase documents that

Relation name	New instance
actedIn	Gary Sweet, Police Rescue
actedIn	Louise Fletcher, One Flew over the Cuckoo's Nest
directed	Akira Kurosawa, Kagemusha
directed	Tyler Perry, Daddy's Little Girls
isAffiliatedTo	John Hewson, Liberal party
isAffiliatedTo	Jay Nixon, Democratic party
locatedIn	Nagoya, Aichi Prefecture
locatedIn	Kathmandu, Nepal
produced	Boz Scaggs, Some Change
produced	Francis Ford Coppola, Apocalypse Now Redux
wrote	Carolyn Janice Cherry, Merchanter's Luck
wrote	Erich Maria Remarque, All Quiet on the Western Front

Table 6.1: Some of Relation instances extracted by our system that did not appear in YAGO.

contain such entities.

Additionally, as DS data is noisy, for accurately evaluating our extractors, we (i) manually annotated a small dataset and (ii) mapped some YAGO relations to ACE. This way we can measure on the ACE data the impact of Wikipedia training data.

6.3.1 ACE (Automatic Content Extraction)

The ACE effort (Doddington et al., 2004) aims at developing technology for automatically carrying out inference in natural language text. The data includes the entities being mentioned, the relations among these entities that are directly expressed, and the events in which these entities participate. Moreover, data includes various source types (image, audio, text) and languages (English, Arabic). We use the ACE 2004 corpus with seven relation types: Physical (PHYS), Person/Social (PER-SOC), Employment/Membership/Subsidiary (EMP-ORG), Agent-Artifact (ART), PER/ORG Affiliation (Other-AFF), GPE Affiliation (GPE-AFF), and Discourse (DISC).

These relationships are explicitly described in the ACE document guidelines.

RE, as defined in ACE, is the task of finding relevant semantic relations between pairs of entities in texts. For example, the following sentence from the ACE 2004 corpus expresses the employee/organization relation (EMP-ORG) between the first entity, i.e. *Tara Singh Hayer* (of type *person*) and the second entity, i.e. *The Indo-Canadian Times* (of type *organization*).

Tara Singh Hayer, editor of **The Indo-Canadian Times.**

Figure 6.1: A text that signifies a relation instance in ACE 2004 with all entity mentions in bold.

6.3.2 YAGO

This is a huge semantic knowledge base derived from WordNet and Wikipedia. It comprises about more than 2 million entities (like *persons*, *organizations*, *cities*, etc.) and 20 million facts connecting such entities. These include the taxonomic Is-A hierarchy as well as semantic relations between entities. The facts of YAGO have been extracted from the category system and the *Infoboxes* of Wikipedia and have been combined with taxonomic relations from Wordnet.

We use the YAGO ontology and the knowledge base, version *2008-w40-2*, whose validation has shown an accuracy of 95% for 99 relations. However, some of them are (a) rather trivial, e.g. *familyNameOf* or *givenNameOf*; (b) describe numerical attributes that change over time, e.g. *hasBudget*, *hasGDP* or *hasPopulation*; (c) symmetric, e.g. *hasPredecessor* and *hasSuccessor*; and (d) used for data management and not convey semantics, e.g. *describes* or *foundIn*. Therefore, we removed trivial relations, unstable relations, and those used for data management. We ob-

tained 1,489,156 instances of 52 relation types to be used with our distant supervised approach. Some examples are shown in Table 6.2.

Relation name	**Size**	**Example**
actedIn	28,836	George Clooney, Batman & Robin
bornIn	36,189	Alan Turing, London
bornOnDate	441,274	William Shakespeare, 26/04/1564
createdOnDate	12,377	A.S. Roma, 22/07/1927
created	95,248	Apple Inc., Dylan
dealsWith	98	Vietnam, France
describes	2,124,543	http://en.wikipedia.org/wiki/British_Columbia, British Columbia
diedIn	13,618	Leonhard Euler, Saint Petersburg
diedOnDate	205,469	Alfred Hitchcock, 29/04/1980
directed	23,723	Mel Gibson, Braveheart
discovered	87	Noam Chomsky, Chomsky hierarchy
domain	94	worksAt, wordnet_person
establishedOnDate	110,830	Grand Canyon National Park, 26/02/1919
exports	72	Ecuador, wordnet_banana
familyNameOf	569,410	Francisco Goya, Goya
givenNameOf	568,852	John Williams, John
graduatedFrom	4,968	Albert Einstein, University of Zurich
happenedIn	3,698	Battle of Waterloo, Waterloo, Belgium
hasAcademicAdvisor	1,599	Georg Cantor, Karl Weierstrass
hasArea	62,720	Rocky Mount, NorthCarolina, 92.7 km2
hasBudget	4,170	Gladiator (2000 film), $103 million
hasCallingCode	311	Hong Kong, 852
hasCapital	1,368	Canada, Ottawa
hasChild	4,454	Nero Claudius Drusus, Claudius
hasCurrency	367	British Virgin Islands, United States dollar
hasDuration	30,791	Quebec, French language
hasEconomicGrowth	43	Israel, 5.3%
hasExpenses	43	United Kingdom, $1,040 billions
hasExport	41	Finland, $92.6 billions
hasGDPPPP	273	Nova Scotia, $31,966 billions
hasProductionLanguage	40,738	The Sixth Sense, English language
hasProduct	997	Sony, PlayStation
hasSuccessor	55,535	Jimmy Carter, Ronald Reagan

Relation name	Size	Example
hasWonPrize	23,076	Albert Einstein, Nobel Prize in Physics
imports	53	Denmark, wordnet_machinery
influences	9,614	Aristotle, Nicolaus Copernicus
interestedIn	2,131	Nicolaus Copernicus, Heliocentrism
inLanguage	3,563,112	London, Londres
isAffiliatedTo	13,038	George W. Bush, Republican Party
isCitizenOf	4,865	Paul Cézanne, France
isLeaderOf	2,886	Vladimir Putin, Russia
isMarriedTo	4,208	Bill Clinton, Hillary Rodham Clinton
isMemberOf	1,257	wordnet_person, wordnet_people
isOfGenre	106,797	The Godfather (novel), wordnet_novel
isPartOf	5,022	wordnet_location, wordnet_space
isSubstanceOf	728	wordnet_pigment, wordnet_paint
livesIn	14,710	Isaac Newton, England
locatedIn	60,261	Philadelphia, Pennsylvania
madeCoverFor	951	John Howe, The Conan Chronicles
musicalRole	15,516	Paul Anka, wordnet singing
originatesFrom	11,497	Elvis Presley, Memphis, Tennessee
participatedIn	7,530	Nazi Germany, Battle of the River Plate
politicianOf	6,198	Bill Clinton, Arkansas
produced	41,747	Francis Ford Coppola, Apocalypse Now
publishedOnDate	11,831	The Citadel (novel), 1937
worksAt	1,401	Stephen Cook, University of Toronto
wrote	12,469	Margaret Mitchell, Gone with the Wind

Table 6.2: Some of selected YAGO relation types and their number of instances.

6.3.3 Freebase

To access to the Wikipedia documents, we use Freebase (version March 27, 2010), which is a dump of the full text of all Wikipedia articles. It has been sentence-tokenized by Metaweb Technologies. For our experiments, we use 100,000 articles of which only 28,074 contain at least one relation for a total of 68,429 of relation instances. These connect 744,060 entities, 97,828 dates and 203,981 numerical attributes. Statistics are shown in Table 6.3.

In Freebase articles, Wikipedia entities like *Person*, *Organization* or

Location are marked whereas numbers or dates are not. This prevents to extract interesting relations between entities and dates, e.g. *John F. Kennedy was born on May 29, 1917* or between entities and numerical attributes, e.g. *The novel Gone with the wind has 1037 pages.* Thus, we designed 18 regular expressions to extract dates and other 25 rules to extract numerical attributes, which range from integer numbers to ordinal numbers, percentage, monetary, speed, height, weight, area, time, and ISBN.

ACE corpus	
Category name	Number
Documents	443
Entities	12,037
Relations	5,784

DS corpus	
Category name	Number
Documents	28,074
Entities	744,060
Dates	97,828
Numerical attributes	203,981
Relations	68,429

Table 6.3: General statistics on the ACE 2004 and DS dataset.

6.3.4 Distant Supervision and generalization

DS for RE is based on the following assumption, if (i) a sentence is connected *in some way* to a database of relations and (ii) it contains the pair of entities participating in such relation then it is likely that such sentence expresses the relation. For our DS, we relax (i) by allowing for the use of an external DB of relations such as YAGO and any document of Freebase. The alignment between YAGO and Freebase is implemented by the Wikipedia page link: for example the link *http://en.wikipedia.org/wiki/James_Cameron* refers to the entity *James_Cameron*.

A simplified version of our approach is the following: for any YAGO relation instance, scan all the sentences of all Wikipedia articles to test point (ii). Unfortunately, this procedure is impossible in practice since there are millions of relation instances in YAGO and millions of Wikipedia

articles in Freebase, i.e. an order of magnitude of 10^{14} iterations[2]. Thus we use a more efficient procedure formally described in Alg. 6.3.1: for each Wikipedia article in Freebase, we scan all of its NEs. Then, for each pair of entities seen in the sentence, we query YAGO to retrieve the relation instance connecting these entities.

Figure 6.2 shows a text derived from Freebase, annotated with the YAGO relation *directed*. In the text, the relation between *Star Wars Episode IV: A New Hope* and *George Lucas* describes the relationship between the second entity and the first where the person is the director of the film.

It should be noted that, our approach solves most of the problems for DS pointed out in (Bunescu and Mooney, 2007). Indeed, such issues are due to the sampling method used to acquire DS sentences: NEs were used as query to a search engine, whose weighting schemes introduce a bias. As, we utilize whole documents and extract from them all possible positive and negative relation instances, no artificial feature (e.g. word) distribution is generated.

6.3.5 Mapping relations between YAGO-ACE

The YAGO knowledge base created from Wordnet and Wikipedia contains 99 relations whereas the ACE 2004 corpus only defines 7 relation types between 7 entity types. To further measure the impact of our Wikipedia dataset and the relations learnt, we mapped 30 relations of YAGO into those of ACE 2004. Surprisingly, we have found a fair correlation between the two different sources, which can help to validate our DS approach. The projection is shown in Table 6.4.

[2]Assuming 100 sentences for each article.

Algorithm 6.3.1: ACQUIRE_LABELED_DATA()

$DS = \emptyset$
$YAGO(R)$: *Instances of Relation R*
for each $\langle Wikipedia\ article : W \rangle \in Freebase$
do {
  $\mathcal{S} \leftarrow$ *set of sentences from* W
  for each $s \in \mathcal{S}$
  do {
    $\mathcal{E} \leftarrow$ *set of entities from* s
    for each $E_1 \in \mathcal{E}$ **and** $E_2 \in \mathcal{E}$ **and** $R \in YAGO$
    do {
      if $R(E_1, E_2) \in YAGO(R)$
      then $DS \leftarrow DS \cup \{s, R^+\}$
      else $DS \leftarrow DS \cup \{s, R^-\}$
    }
  }
}
return $(\mathcal{DS})$

Star Wars Episode IV: A New Hope, is a 1977 American epic space opera film, written and directed by **George Lucas**.

Figure 6.2: A text derived from Wikipedia Freebase, annotated with YAGO relation with all entity mentions in bold.

6.4 Joint Learning Paradigms

We model RE using state-of-the-art kernel methods: syntactic structures are used to represent relation instances whereas kernel functions measure the similarity between pairs of them. Such functions correspond to scalar products between implicit feature vectors in the space of substructures. Additionally, we define a joint model between the RE classifier trained on ACE and trained on DS data such that we can merge together the

YAGO relations	Projection
actedIn	ART
bornIn	PHYS
created	ART
dealsWith	EMP-ORG
diedIn	PHYS
directed	ART
discovered	ART
graduatedFrom	EMP-ORG
happenedIn	PHYS
hasAcademicAdvisor	PER-SOC
hasCapital	PHYS
hasChild	PER-SOC
hasCurrency	ART
hasOfficialLanguage	ART
hasProduct	ART
hasProductionLanguage	ART
wrote	ART

YAGO relations	Projection
hasSuccessor	PER-SOC
hasWonPrize	ART
influences	PER-SOC
interestedIn	ART
isAffiliatedTo	EMP-ORG
isCitizenOf	GPE-AFF
isLeaderOf	EMP-ORG
isMarriedTo	PER-SOC
livesIn	PHYS
locatedIn	PHYS
madeCoverFor	ART
originatesFrom	PHYS
participatedIn	ART
politicianOf	Other-AFF
produced	ART
worksAt	EMP-ORG

Table 6.4: 33 YAGO relation types projected into ACE.

information from the two datasets on similar relation type.

6.4.1 RE based on Kernel Methods

State-of-the-art ACE RE, i.e. (Zhang et al., 2006; Nguyen et al., 2009b), uses tree kernels applied to constituent and dependency syntactic structures, extracted from the sentences expressing the target relations. Given a parse tree, the path-enclosed tree (PET) is used as input of a tree kernel function. PET is the smallest common sub-tree including the two entities of a relation. Figure 4.1.a shows the constituent tree and figure 4.1.b shows a fragment of the dependency tree of the sentence: *In Massachussets, U.S. financiers are working overtime.* The dashed frame in Figure 4.1.a surrounds PET associated with the two mentions, *financiers* and *Massachussets*. Moreover, to improve the representation, two extra nodes T1-PER and T2-LOC, denoting the type PERSON and LOCATION, are

added to the parse tree, above the two target NEs, respectively.

In our experiments, we use the model defined in (Zhang et al., 2006), which combines a syntactic tree kernel applied to constituent parse trees and a polynomial kernel over feature extracted from the entities:

$$CK_1 = \alpha \cdot K_P + (1 - \alpha) \cdot TK, \tag{6.1}$$

where α is a coefficient to give more or less impact to the polynomial kernel, K_P, and TK is the syntactic tree kernel (Collins and Duffy, 2001) applied to PET.

We also use the best model in (Nguyen et al., 2009b), which combines the advantages of the two parsing paradigms by adding six sequence kernels. These are applied to paths derived from the dependency tree and enriched with node labels of the constituent tree as follows:

$$CSK = \alpha \cdot K_P + (1 - \alpha) \cdot (TK + \sum_{i=1,..,6} SK_i), \tag{6.2}$$

where SK_i are the sequence kernels applied to the structure i defined in (Nguyen et al., 2009b).

In our application domain there are many different categories of name entities, e.g. Editor, President, Employer, and so on. Thus the typically available NE types, e.g. Person, Organization, Location, Time, Numbers, do not provide much selective information. For this purpose, we also provide adapted kernels by simply removing the category label in the nodes of the trees and in the sequences. This data transformation corresponds to define different kernel functions (Cristianini and Shawe-Taylor, 2000).

6.4.2 Joint Distantly and Directly Supervised Model

An interesting test of the quality of our DS data can be carried out by using it for ACE RE experiments. This way, we can use the gold and

well annotated dataset of ACE to accurately measure the impact of DS data. For this purpose, we define a joint model as follows: first, we select the portion of hand-labeled ACE 2004 corpus containing common relations (see the mapping in Section 6.3.5).

Second, we create a huge labeled dataset under distant supervision assumption (described in Section 6.3.4) from Wikipedia news articles and YAGO knowledge base. Thanks to the projection from YAGO to ACE relations, we generate the two datasets under the same set of labels. This way, labeled data can be automatically acquired from a huge corpus and used to enrich ACE relation extractors.

Third, we train (i) the M_{ace} RE model on ACE dataset and (ii) the M_{mixed} model on ACE dataset mixed with the labeled data from Wikipedia (by using for example CSK).

Next, as standard SVM classifiers do not provide calibrated posterior probabilities we apply Platt transformation (Platt, 2000) improved by (Lin et al., 2007) with an additional sigmoid function. This allows us to map the SVM outputs of the two models M_{ace} and M_{mixed} into probabilities.

Finally, we linearly combine the probability of the two classifiers as follows:

$$P(C|r) = \alpha \cdot P(C|r, C_1) + \beta \cdot P(C|r, C_2), \tag{6.3}$$

where C_i is the output of classifier i, α and β are the weights learned from a validation set to encode the importance of the classifier for detecting the relation r. This combination provides a more robust model with respect to domain change.

6.5 Experiments with standard RE

The aim of the experiments is to demonstrate that the DS data acquired with our algorithm (Alg. 6.3.1) produces reliable and practical usable relation extractors. For this purpose, we test our state-of-the-art RE on DS data and on the joint DS and ACE data. Then we test their applicability by carrying out end-to-end RE evaluation for ACE and ACE+DS by using our automatic Named Entity Recognizer.

6.5.1 Experimental setup

We used the English portion of the ACE 2004 corpus including 443 documents, annotated with seven entity types and seven relation types. We obtained 5,784 positive and 55,650 negative examples when generating pairs of entity mentions as candidate relations. We employed the Stanford Parser (Klein and Manning, 2003) to produce parse trees. The candidate relations are generated by iterating all pairs of entity mentions in the same sentence.

Regarding the DS data extraction (see tables 6.3), we used two PCs, one with Intel X5270 3.50GHz CPU, 32GB RAM, another with 3.40GHz CPU and 8GB RAM to run the Algorithm 6.3.1. We processed about 25,000 Wikipedia documents per day per machine. When we added the generation of structures and features, the whole procedure required one day to process 5,000 Wikipedia documents (per machine). Thus, it took about 10 days to create the dataset and the computational learning files.

To train and test our binary relation classifier, we used SVMs, where relation detection is formulated as a multiclass classification problem. We employed *one vs. rest*, selecting the instance with largest margin as the final label. We used the Tree Kernel toolkit[3] (Moschitti, 2004) as SVM platform to implement CK_1 and CSK (see Section 4.3). The training

[3]http://disi.unitn.it/ moschitt/Tree-Kernel.htm

phase with convolution kernels on syntactic parse tree and diverse sequence kernels on the large DS data took 3 days.

For testing on ACE data, we applied 5-fold cross-validation and evaluated single classifiers with the average of Precision, Recall and F1 on the 5-folds. The overall accuracy is measured with the mean of the Micro-Average (All) over the 5-folds.

For testing on Wikipedia, as DS data may be incorrect, we created a test set by sampling 200 articles from Freebase (these articles are not used for training). An expert annotator then examined one sentence at a time and took all possible pairs of entities, where the latter were already marked in the sentence. For each pair of entities, the considered 52 relations from YAGO (and used in our RE system) are marked as positive or negative, respectively. The annotator obtained 2,601 relation instances used for evaluation.

Regarding NE recognition, we applied CRFs to Wikipedia data but we could not use the whole amount of data. Thus we sampled 18,198 Wikipedia articles, selecting 4/5 for training and the rest for testing. The training phase took 14 hours and 30 minutes, whereas the classification took less than 10 minutes.

6.5.2 Results

Results on Wikipedia

Table 6.5 shows the performance of individual classifiers as well as the overall Micro-average F1 for our adapted CSK on the Wikipedia test set. We note that the global measure reaches a Precision of 91.42% with a Recall of 62.57%, giving an F1-score of 74.29%. Although, the setting and task were different a referent result is the F1 of 61% obtained in (Hoffmann et al., 2010). We also evaluated CK_1 obtaining Micro-average Precision, Recall

Category	**Pr**	**Re**	**F_1**
bornIn	67.57	53.17	59.51
bornOnDate	97.99	95.22	96.58
created	92.00	68.56	78.57
dealsWith	92.31	73.47	81.82
diedIn	100.00	18.52	31.25
diedOnDate	95.00	82.61	88.37
directed	85.19	51.11	63.89
establishedOnDate	80.83	62.18	70.29
graduatedFrom	75.00	27.27	40.00
happenedIn	100.00	21.05	34.78
hasCapital	93.69	61.54	74.29
hasChild	73.33	42.31	53.66
hasOfficialLanguage	88.24	44.12	58.82
hasProductionLanguage	100.00	36.36	53.33
hasSuccessor	95.65	25.00	39.64
hasWonPrize	86.67	41.94	56.52
influences	90.00	22.50	36.00
interestedIn	100.00	22.22	36.36
isAffiliatedTo	86.32	71.30	78.09
isMarriedTo	95.00	30.65	46.35
livesIn	100.00	34.62	51.43
locatedIn	87.85	78.33	82.82
participatedIn	77.27	42.50	54.84
politicianOf	95.45	28.77	44.21
wrote	82.61	42.22	55.88
All	**91.42**	**62.57**	**74.29**

Table 6.5: Evaluation of extraction of 52 relations on the Wikipedia manually annotated test set; the performance of 25 out 52 individual relations is also shown.

and F1 of 85.50%, 61.01% and 71.21%, respectively. This lower result suggests that the combination of dependency and constituent syntactic structures is very important (+3.08 absolute percent points).

Using Wikipedia Relational Extractors to improve on ACE

In the ACE program, relations are defined between pairs of entities. These not only refer to NEs but also to mentions, e.g. indicated by a common noun or noun phrase, or represented by a pronoun. In contrast, Wikipedia instances mainly refer to NEs, e.g. *Leonardo Da Vinci*, *Canada* or *Titanic*, and we do not use pronominal references for building RE instances. Thus, we carried out two kinds of experiments: using (i) RE task as defined in ACE with all kind of entities and (ii) only relations between named entities. We have observed that the NE relations only exist for the classes: Physical (PHYS), Employment/Membership/Subsidiary (EMP-ORG) and GPE Affiliation (GPE-AFF).

Table 6.6 presents the combination results. Overall, using Wikipedia data improves the state-of-the-art of standard RE from 64.74% to 67.59%. Moreover, if we focus on *proper* NE relations, i.e. of the type indicated in point (ii), the relation extractors improve from 73.94% to 76.23%. These results are interesting as show that (a) we can improve the best systems with DS and (b) the RE learned from Wikipedia can be mapped into those defined by expert linguists on ACE. We also tested a model learned from only DS data. For space reason, we do not report the complete results: as expected, its overall F1 is lower than the model trained on only ACE (about 10 absolute percent points less).

6.6 End-to-end Relation Extraction

6.6.1 Motivation

In this section, we describe the experiments using automatic NEs. Previous work, e.g. (Zhang et al., 2006; Zhou et al., 2007; Nguyen et al., 2009b) performed extraction using gold entity features such as entity types (*Per-*

Category	**Pr**	**Re**	**F_1**
PHYS	56.28	44.51	49.71
PER-SOC	88.12	59.8	71.25
EMP-ORG	80.82	76.73	78.72
ART	80.68	39.2	52.76
Other-AFF	62.73	17.11	26.89
GPE-AFF	76.55	32.32	45.45
DISC	80.15	59.85	68.53
All	74.47	57.26	64.74

Table 6.6: Results on ACE 2004 considering all the type of entities and all the 7 ACE relations.

Category	**Pr**	**Re**	**F_1**
PHYS	58.22	48.44	52.88
PER-SOC	91.06	64.74	75.68
EMP-ORG	81.76	76.66	79.13
ART	80.68	37.14	50.86
Other-AFF	62.73	17.11	26.89
GPE-AFF	78.49	32.26	45.73
DISC	80.15	59.85	68.53
All	77.65	59.84	67.59

Table 6.7: Improvement on ACE 2004 considering all the type of entities and all the 7 ACE relations.

Category	**Pr**	**Re**	**F_1**
PHYS	72.06	67.12	69.50
EMP-ORG	85.71	80.00	82.76
GPE-AFF	78.95	75.00	76.92
All	72.41	75.54	73.94

Table 6.8: Results on ACE 2004 on the three relations between named entities where relation extractors are trained using only ACE data.

Category	Pr	Re	F_1
PHYS	72.46	68.49	70.42
EMP-ORG	90.00	81.82	85.71
GPE-AFF	83.33	75.00	78.95
All	80.16	72.66	76.23

Table 6.9: Improvement on ACE 2004 on the three relations between named entities where relation extractors are trained using only ACE+Wikipedia data.

son, *Location*, *Organization*), entity subtypes (*Nation*, *Population-Center* for *GPE*). For example, in the sentence *Bush went to Washington*, the type of the first named entity, *Bush*, is PERSON and for the second named entity, *Washington*, is LOCATION. When accurate, such features improve performance. In case of fully automatic systems they introduce noise and in Wikipedia they are not available. Thus, we removed all gold entity features (entity type, entity subtype, mention type, and LDC mention type) from ACE annotations. We modeled tree and sequence kernels based on constituent and dependency parse trees along with a few features that can be extracted automatically such as the string and the headword of the entity.

Note that in (Nguyen et al., 2009b; Zhou et al., 2007; Zhang et al., 2006), even for tree kernels, the tree structures were also integrated with entity types. Therefore, in the parse trees in Figure 4.1, we replaced entity types PER, ORG, LOC with a generic type ETYPE 6.3.

6.6.2 Entity Extraction from ACE and Wikipedia

For entity extraction, we followed the design in (Nguyen et al., 2010) by applying CRF++ [4]. We performed automatic entity extraction from seven classes from ACE 2004 and entity detection from Wikipedia. While ACE

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

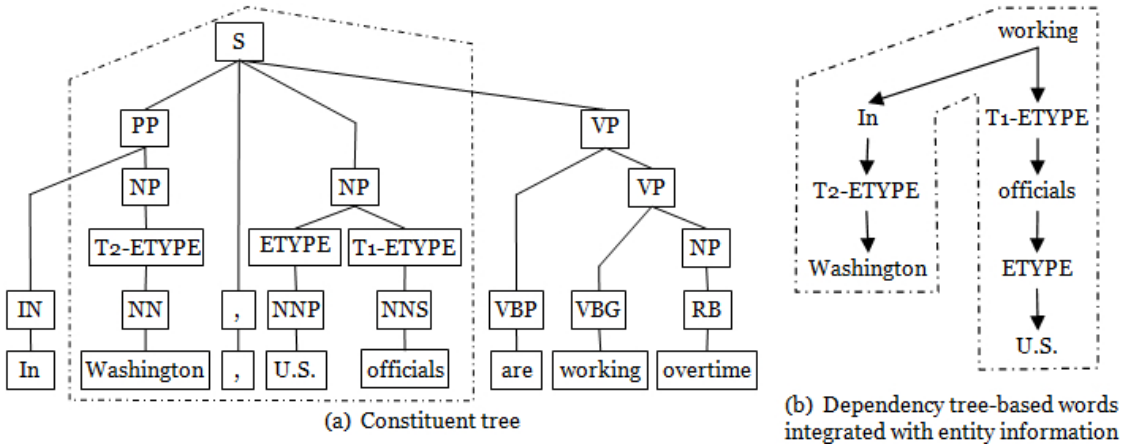


Figure 6.3: The constituent and dependency parse trees integrated with entity information

documents have been annotated with seven classes *Person, Organization, Facility, Location, GPE, Vehicle, Weapon*, for Wikipedia we used Freebase as learning source, where entities have been annotated in each Wikipedia article. Note that for Wikipedia, the entity detection has been done for only entities, like *Person, Organization, Location.* For dates and numerical attributes, we used the patterns as described in Section 6.3.4. The results reported in Table 6.11 are rather lower than in standard NE recognition. We should consider that our NER also tags mentions in ACE, which is a hard task whereas for Wikipedia, the entity instances from YAGO potentially belong to thousands of different categories. Although, we do not attempt to categorize entities, this indicates higher complexity in boundaries detection of NEs.

6.6.3 RE from Automatic Entity Extraction

Web data entities are often not annotated and not available as in hand-labeled corpora like ACE or in Wikipedia pages. In this new experiment, we move to a novel task where entities are detected and classified automatically from a classifier. This way, we aim at designing an end-to-end RE system,

where entities are not known beforehand. We also introduce a new task, that is extraction of Wikipedia relations from any web text, i.e. detection of Wikipedia instances from any web page and not only from Wikipedia articles (where links often exist for Wikipedia instances).

Setting	**Entity Extraction**	**Gold features and Gold NEs**	**No gold features**	
			Gold NEs	**Automatic NEs**
Precision	77.84	76.60	74.47	70.27
Recall	70.26	67.00	57.26	47.52
F1	73.85	71.50	64.74	56.70

Table 6.10: Results on end-to-end RE from ACE.

Setting	**Entity Extraction**	**No gold features**	
		Gold NEs	**Automatic NEs**
Precision	68.84	91.42	82.16
Recall	64.56	62.57	56.57
F1	66.63	74.29	67.00

Table 6.11: Results on end-to-end RE from Wikipedia.

The results are shown in Table 6.10 and Table 6.11. We note that the gold entity features lead to very good F1. When we remove these, the F1 decreases from 71.50% to 64.74%. Nevertheless, without gold entity features, RE from Wikipedia still achieves very good performance, i.e. an F1 of 74.29%.

Chapter 7

Future Work

7.1 Reranking Approach for RE

It is often advantageous to produce the top N candidates instead of just the top *1*, since a secondary model can be employed with arbitrary features to re-order the top N and hopefully improves the quality of the top ranked candidate. If there exists a reranking scheme that, for each sentence, can pick a better candidate from the top N hypotheses produced by the SVMs classifier, the performance of the system then can be improved by integrating global features to discriminate from good to bad hypotheses.

7.1.1 From One Vs. Rest to *N-Best* hypotheses

State-of-the-art kernel methods applied to RE (Zhang et al., 2006; Nguyen et al., 2009b) make use of Support Vector Machines (SVMs) to produce binary classifiers, then employ the *one vs. rest* strategy to learn multi-class classification. This strategy is carried out by selecting the instance with largest margin as the final answer. However, NLP literature has shown that, the first candidate is not always the best answer, that may be seen in the next *10*, *20*, or *50* candidates. Therefore, it is worthwhile studying an appropriate reranking framework for RE.

We use the state-of-the-art ACE RE (Nguyen et al., 2009b) that combines the advantages of the two parsing paradigms by adding six sequence kernels. These are applied to paths derived from the dependency tree and enriched with node labels of the constituent tree, as described in section 4.3. Then, we apply Platt transformation (Platt, 2000) to transform from SVMs scores to probabilities. We generate relation hypotheses made on an entire sentence in Alg. 7.1.1. Additionally, since it is not needed to generate hypotheses for every relation instances, in our approach, we use a more efficient procedure formally described in Alg. 7.1.2 to select which relation instance to put in the hypotheses to be *reranked*.

Algorithm 7.1.1: GENERATE_HYPOTHESES()

$\mathcal{O} = \emptyset$
$\mathcal{RT} \leftarrow$ *set of seven relation types*
$h_1 :$ *first sequence derived from One vs. rest*
$\mathcal{O} \leftarrow \mathcal{O} \cup \{h_1\}$
for each $\langle$ *hypothesis* $: h\rangle \in \mathcal{O}$
do {
  for each $\langle$ *relation instance* $: R\rangle \in h$
  do {
    $P \leftarrow$ *probability of* R
    for all $T \in RT$
    do {
      $T_{max} \leftarrow T$ *with maximum probability less than* P
    }
    $R_{new} \leftarrow R$ *with type* T_{max}
    $h_{new} \leftarrow h$ *with* R *replaced by* R_{new}
    $\mathcal{O} \leftarrow \mathcal{O} \cup \{h_{new}\}$
  }
}
for all $h \in \mathcal{O}$
do {$h_{max} \leftarrow h$ *with maximum probability*
return (h_{max})

Algorithm 7.1.2: RELATION_SELECTION()

$H = \emptyset$
$thres0 \leftarrow$ *threshold on relation* $NONE$
$thres1 \leftarrow$ *threshold on relation not* $NONE$
for each $\langle$ *relation instance* $: R\rangle \in s$
do
- $T \leftarrow$ *relation type of* R
- $P \leftarrow$ *probability of* R
- **if** $(T is NONE$ **and** $P \leq thres0)$
- **or** $(T \neq NONE$ **and** $P \leq thres1)$
- **then** $\mathcal{S} \leftarrow \mathcal{S} \cup \{T\}$

return $(\mathcal{S})$

7.1.2 Candidate Set Size and Oracle Performance

In reranking strategy, it is important to study the influence of the candidate set size on the quality of the reranked output. An interesting question is what the upper bound on reranker performance is the oracle performance. Table 7.1 shows the oracle peroformance according to the candidate set size.

Size	Pr	Re	$\mathbf{F_1}$
1	74.20	69.35	71.70
2	89.46	70.05	78.57
3	86.68	73.12	79.32
4	88.32	74.25	80.68
5	90.06	73.41	80.89
6	90.34	75.03	81.98
7	90.47	75.84	82.51
8	90.55	76.31	82.82

Table 7.1: Oracle performance as a function of candidate set size

7.2 Potential People Search Engine

7.2.1 Motivation

Information Retrieval (IR) is concerned with finding the documents that answer a specific information need within a given corpus. Text retrieval is referred to finding relevant information in a text collection. It is especially important, because the most frequently wanted information is often textual, and techniques for retrieving textual information can be useful for retrieving other media information, when there is companion text. Given a set of unordered text documents, the task of Text Retrieval (TR) can be defined as using a user query (i.e., a description of user's information need) to identify a subset of documents that satisfy this query. IR research studies this problem from different aspects, which can be grouped into three categories: ranking algorithms, data structures and user interfaces. Ranking algorithms estimate the relevance of a document with respect to a query, so that the documents can be ordered according to some scores. Data structures are concerned with storing the corpus to allow a fast computation of the ranking function. User interfaces try to design an intuitive interface to query the system and to present the resulting rankings.

One limitation of text-based IR systems is in the capability of using entities or relationships in user search and navigation. These features are often associated with the user information need and promise to provide an effective IR user interface in terms of semantic relations and concepts. Therefore, entities/relations based user query is expected. Besides this, with respect to a query, current SEs present resulting documents in the form of URL list. Another weak point of current IR systems lies in the usage of shallow representations of texts or more complex representations such as conceptual or directed bipartite graph. One drawback of such representations is the lack of syntactic, structural information for query/doc-

ument representation and for relevance estimation in document ranking.

7.2.2 Problem Statement

Since the 90s, IE have been dominated with statistical machine learning approaches in diverse algorithms and paradigms. The learning algorithm comes in various forms, some examples are Hidden Markov Models (HMMs), Conditonal Random Fields (CRFs), or Support Vector Machines (SVMs). The learning paradigm may be supervised from the full set of labeled examples or semi/weakly supervised from very few labeled data. Nevertheless, single learning algorithm yields not-enough significant results. Furthermore, due to the growing ubiquity of unlabeled data, learning with unlabeled data has drawn increasing attention in machine learning. Some combination techniques have been proposed to meet these requirements with encouraging results that have been widely seen in various IE tasks.

Meanwhile, in the IR community, we deal with the problem that is the integration of syntactic/semantic structures to tackle the linguistic nature of queries and documents. We demonstrate that relational features derived from syntactic parse trees and entity/relation structures are useful for the task of People Search (PS). Person name disambiguation has long been an important problem in text mining and web search. There exist prevalent occurrences of identical person names on different web pages but actually refer to distinct people, being able to resolve the person name references on web content is essential for various applications.

In this research direction we deal with two research problems: 1) the necessity of combining learning models on one of two issues related to the nature of learning: i) difference in the basic learning algorithms or ii) different paradigms; 2) the lack of unified representation for syntactic/semantic structures in search.

Chapter 8

Conclusions

The research presented in this thesis has focused on the design, implementation and evaluation of machine learning models for the key tasks in information extraction: named entity recognition and relation extraction. The learned extraction models have shown improved extraction performance as a result of their ability to exploit different kinds of encoding features, and novel useful structures and evidence. Moreover, in a significant attempt to reduce required supervision from purely supervised learning approaches, we design a distant supervised RE where both the relation providers and the training instances are gathered from external repositories. These allow for potentially obtaining larger training data and many more instances, defined in different sources, without requiring heavy supervision.

We have first analyzed the impact of structural and flat representation, and the advantages of kernel-based approaches for modeling the 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 for Italian and reach com-

parable performance for English.

There are multiple opportunities for future work in this area. The discrimination from good to bad hypotheses is the key point that leads to improved performance of any task on reranking. With an appropriate reranking scheme, one may try to apply it on a broader range of related tasks. The results gotten with NER reranking model is promising for various NLP works to come.

We have then studied the use of several types of syntactic information: constituent and dependency syntactic parse trees. A relation is represented by taking the path-enclosed tree (PET) of the constituent tree or of the path linking two entities of the dependency tree. For the design of automatic relation classifiers, we have investigated the impact of dependency structures to the RE task. Our novel structures, which account for the two syntactic parses, are experimented with the appropriate convolution kernels and show significant improvement with respect to the state-of-the-art in RE.

It would be interesting to see the performance of our proposed kernels in combination with more features derived from external knowledge such as ontological, lexical resource or WordNet (Basili et al., 2005; Bloehdorn et al., 2006) or shallow semantic trees, (Moschitti and Bejan, 2004; Giuglea and Moschitti, 2006; Moschitti et al., 2007). It is also feasible to design a new tree-based structures, to combine the information of both constituent and dependency parses. From dependency trees we can extract more precise but also more sparse relationships (which may cause overfit). From constituent trees, we can extract subtrees constituted by non-terminal symbols (grammar symbols), which provide a better generalization (with a risk of underfitting).

In the last part of this thesis, we have proposed several contributions to Relation Extraction: (i) a new approach to distant supervision (DS) to

create training data using relations defined in different sources, i.e. YAGO, and potentially using any Wikipedia document; (ii) adaptation and experimentation of state-of-the-art models based on two syntactic paradigms for RE from Wikipedia pages; (iii) a mapping from Wikipedia to ACE relations; and (iv) end-to-end systems applicable both to Wikipedia pages as well as to any natural language text.

The results show (1) a high F1 of 74.29% on extracting 52 YAGO relations from any Wikipedia document (not only from *Infobox* related pages); this improves on previous work by 13.29 absolute percent points; (2) the importance of using both dependency and constituent structures (+3.08% when adding dependency information to RE based on constituent trees); (3) Wikipedia data can be used for ACE RE since when it is jointly used with the best RE model, the latter improves, e.g. by 2.85 (67.59-64.74); (4) the end-to-end system is useful for real applications as it shows a meaningful accuracy, i.e. 67% on 52 relations.

Bibliography

References

[Agarwal and Rambow2010] Apoorv Agarwal and Owen Rambow. 2010. Automatic detection and classification of social events. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1024–1034, Cambridge, MA, October. Association for Computational Linguistics.

[Agichtein and Gravano2000] Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*.

[Baroni et al.2004] 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.

[Basili et al.2005] Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2005. Effective use of WordNet semantics via kernel-based learning. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 1–8, Ann Arbor, Michigan, June. Association for Computational Linguistics.

[Bender et al.2003] 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.

[Bikel et al.1997] 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.

[Bloehdorn et al.2006] Stephan Bloehdorn, Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2006. Semantic kernels for text classification based on topological measures of feature similarity. In *Proceedings of ICDM*, pages 808–812, December.

[Brin1998] Sergey Brin. 1998. Extracting patterns and relations from world wide web. In *Proceeding of WebDB Workshop at 6th International Conference on Extending Database Technology*, pages 172–183.

[Bunescu and Mooney2005a] Razvan Bunescu and Raymond Mooney. 2005a. 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.

[Bunescu and Mooney2005b] Razvan C. Bunescu and Raymond J. Mooney. 2005b. Subsequence kernels for relation extraction. In *Proceedings of NIPS*, pages 171–178.

[Bunescu and Mooney2007] Razvan Bunescu and Raymond Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 576–583, Prague, Czech Republic, June. Association for Computational Linguistics.

[Cancedda et al.2003] Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. 2003. Word sequence kernels. *journal of Machine Learning Research*, pages 1059–1082.

[Carreras et al.2003a] 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.

[Carreras et al.2003b] 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.

[Charniak and Johnson2005] 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.

[Chieu and Ng2002] 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.

[Chieu and Ng2003] 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.

[Chinchor and Robinson1998] Nancy Chinchor and Patricia Robinson. 1998. Muc-7 named entity task definition. In *The MUC*.

[Collins and Duffy2001] Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proceedings of NIPS*, pages 625–632.

[Collins and Duffy2002] 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.

[Collins2000] Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of ICML*, ICML '00, pages 175–182. Morgan Kaufmann Publishers Inc.

[Collins2002] 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.

[Cristianini and Shawe-Taylor2000] Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, Cambridge, United Kingdom.

[Culotta and Sorensen2004] Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 423–429, Barcelona, Spain, July.

[Curran and Clark2003] 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.

[DARPA1987 1995] DARPA. 1987-1995. In *Proceedings of the Message Understanding Conferences (MUCs)*. Morgan Kaufmann.

[DeJong1982] G. F. DeJong. 1982. An overview of the frump system. In Lehnert and Ringle, editors, *Strategies for Natural Language Processing*, Hillsdale HJ. Erlbaum.

[Doddington et al.2004] George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) programtasks, data, and evaluation. In *Proceedings of LREC*, pages 837–840, Barcelona, Spain.

[Doddington1999 2008] George R. Doddington. 1999-2008. Automatic content extraction (ace).

[Etzioni et al.2008] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Commun. ACM*, 51:68–74, December.

[Florian et al.2003] 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.

[Giuglea and Moschitti2006] Ana-Maria Giuglea and Alessandro Moschitti. 2006. Semantic role labeling via framenet, verbnet and propbank. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 929–936, Sydney, Australia, July. Association for Computational Linguistics.

[Hasegawa et al.2004] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 415–422, Barcelona, Spain, July.

[Haussler1999] David Haussler. 1999. Convolution kernels on discrete structures. In *Technical Report UCS-CRL-99-10*, University of California, Santa Cruz.

[Hoffmann et al.2010] Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 286–295, Uppsala, Sweden, July. Association for Computational Linguistics.

[Huang2008] 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.

[Johansson and Moschitti2010] 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.

[Kambhatla2004] Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, pages 178–181, Barcelona, Spain, July. Association for Computational Linguistics.

[Klein and Manning2003] Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan, July. Association for Computational Linguistics.

[Lafferty et al.2001] 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.

[Leek1997] Timothy Robert Leek. 1997. Information extraction using hidden markov models.

[Liddell et al.1891] H.G. Liddell, R. Scott, and J.M. Whiton. 1891. *A lexicon abridged from Liddell & Scott's Greek-English lexicon*. Harper.

[Lin et al.2007] Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. 2007. A note on platts probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267–276.

[Lodhi et al.2002] 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.

[Magnini et al.2006] 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*.

[Marcus et al.1993] Mitchell P. Marcus, Beatrice Santorini, , and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330.

[Mayfield et al.2003] 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.

[McCallum and Li2003] 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.

[Metaweb Technologies2010] Metaweb Technologies. 2010. Freebase wikipedia extraction (wex), March.

[Miller et al.2000] Scott Miller, Heidi Fox, Lance Ramshaw, , and Ralph Weischedel. 2000. A novel use of statistical parsing to extract information from text. In *Proceedings of NAACL*, pages 226–233, Seattle, USA.

[Mintz et al.2009] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore, August. Association for Computational Linguistics.

[Moschitti and Bejan2004] Alessandro Moschitti and Cosmin Adrian Bejan. 2004. A semantic kernel for predicate argument classification. In Hwee Tou Ng and Ellen Riloff, editors, *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 17–24, Boston, Massachusetts, USA, May 6 - May 7. Association for Computational Linguistics.

[Moschitti et al.2007] Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, Prague, Czech Republic, June. Association for Computational Linguistics.

[Moschitti et al.2008] Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2008. Tree kernels for semantic role labeling. *Computational Linguistics*, 34:193–224, June.

[Moschitti2004] 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.

[Moschitti2006] 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.

[Nguyen and Cao2007] Truc-Vien T. Nguyen and Tru H. Cao. 2007. Vnkim ie: automatic extraction of vietnamese named entities on the web. *New Generation Computing*, 25:277–292, January.

[Nguyen and Moschitti2011] Truc Vien T. Nguyen and Alessandro Moschitti. 2011. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 277–282, Portland, Oregon, USA, June. Association for Computational Linguistics.

[Nguyen et al.2009a] Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009a. 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.

[Nguyen et al.2009b] Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009b. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1378–1387, Singapore, August. Association for Computational Linguistics.

[Nguyen et al.2010] 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.

[Platt2000] John C. Platt. 2000. Probabilities for sv machines. *Advances in Large Margin Classifiers*, pages 61–74.

[Popov et al.2003] Borislav Popov, Atanas Kiryakov, Dimitar Manov, Angel Kirilov, and Ognyanoff Miroslav Goranov. 2003. Towards semantic web information extraction. In *In proceedings of ISWC (Sundial Resort.*

[Ratinov and Roth2009] 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.

[Riedel et al.2010] Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of ECML-PKDD*, volume 6323 of *Lecture Notes in Computer Science*, pages 148–163. Springer Berlin / Heidelberg.

[Rifkin and Poggio2002] Ryan Michael Rifkin and Tomaso Poggio. 2002. *Everything old is new again: a fresh look at historical approaches in machine learning.* PhD thesis, Massachusetts Institute of Technology.

[Robert Müller et al.2001] Klaus Robert Müller, Sebastian Mika, Gunnar Rätsch, Koji Tsuda, , and Bernhard Schölkopf. 2001. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201.

[Roth and tau Yih2002] Dan Roth and Wen tau Yih. 2002. Probabilistic reasoning for entity and relation recognition. In *Proceedings of COLING*, Taipei, Taiwan.

[Schölkopf and Smola2001] Bernhard Schölkopf and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

[Shen et al.2004] 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.

[Suchanek et al.2007] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago - a core of semantic knowledge. In *Proceedings of WWW*, pages 697–706.

[Surdeanu et al.2003] 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.

[Tjong Kim Sang and De Meulder2003] 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.

[Vapnik1995] Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer–Verlag, New York.

[Vapnik1998] Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. John Wiley and Sons, New York.

[Wiegand and Klakow2010] Michael Wiegand and Dietrich Klakow. 2010. Convolution kernels for opinion holder extraction. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages

795–803, Los Angeles, California, June. Association for Computational Linguistics.

[Yates2009] Alexander Yates. 2009. Extracting world knowledge from the web. *IEEE Computer*, 42(6):94–97, June.

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

[Zelenko et al.2002] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 71–78. Association for Computational Linguistics, July.

[Zhang et al.2005] Min Zhang, Jian Su, Danmei Wang, Guodong Zhou, and Chew Lim Tan. 2005. Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In *International Joint Conference on Natural Language Processing*, pages 378–389.

[Zhang et al.2006] Min Zhang, Jie Zhang, Jian Su, and GuoDong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 825–832, Sydney, Australia, July. Association for Computational Linguistics.

[Zhao and Grishman2005] Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of ACL*, pages 419–426, Ann Arbor, Michigan, USA.

[Zhou and Su2002] 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.

[Zhou et al.2005] GuoDong Zhou, Jian Su, Jie Zhang, , and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of ACL*, pages 427–434, Ann Arbor, USA, June.

[Zhou et al.2007] GuoDong Zhou, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 728–736, Prague, Czech Republic, June. Association for Computational Linguistics.