SEMANTICS IN (A METRIC) SPACE:
COMPOSITIONAL DISTRIBUTIONAL MODELS
FOR COMPLEX NATURAL LANGUAGE STRUCTURES

Paolo Annesi

Supervisor:
Prof. Roberto Basili

Rome, October 2013
I wasn’t lonely. I experienced no self-pity.

I was just caught up in a life in which I could find no meaning.

Charles Bukowski
Acknowledgements

I would like to thank my supervisor, prof. Roberto Basili, who supported me in these years, for giving me three valuable things: method, strength and sleeplessness.

To all my friends and labmates. There is no need to list: you are all essential to me.

To Roberta.

To Raffaella and Alessandro.

To my mom and dad, my sister and my nephew.
Dealing with semantic information in the current global web era means to solve large scale problems about the integration of structured and unstructured data. Natural Language is the common ground for synthesizing/generating, composing and delivering meaning in these different forms. It is the core platform where semantic integration can happened.

Computational treatment of Natural Language semantics thus becomes a crucial target for research in computer science. In this area we have been studying the integration of well known Distributional Models of lexical semantics, compositional methods in truth functional meaning representation theories and the mathematical paradigms of statistical learning theory (in particular Tree Kernel functions), in order to provide effective metrics for semantic inference over texts.

The aim of this work is to define spaces where the semantics of complex Natural Language expressions can be represented for direct inference tasks (such as textual inference or similarity estimation) or Machine Learning of complex linguistic interpretation processes (e.g. paraphrasing or Semantic Role Labeling).

Experiments on component based algebraic operators, acting on pairs of linguistic objects, as well as extension of traditional Tree Kernels are presented in this thesis. Extensive results suggest that our Compositional Distributional Model can be successfully applied in a variety of linguistic applications.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4 Generalization capability of CDS Model, and application in an IR</td>
<td>64</td>
</tr>
<tr>
<td>scenario for ranking and clustering</td>
<td></td>
</tr>
<tr>
<td>4.5 DCS model based on Support Subspace for Textual Similarity</td>
<td>69</td>
</tr>
<tr>
<td>5 Distributional Compositional Semantics in Tree Kernel Based Learning</td>
<td>77</td>
</tr>
<tr>
<td>5.1 Tree Kernels</td>
<td>81</td>
</tr>
<tr>
<td>5.1.1 Syntactic Tree Kernels</td>
<td>83</td>
</tr>
<tr>
<td>5.1.2 The Partial Tree Kernel</td>
<td>85</td>
</tr>
<tr>
<td>5.1.3 The Smoothed Partial Tree Kernel</td>
<td>87</td>
</tr>
<tr>
<td>5.2 Towards a Compositional version of a SPTK</td>
<td>90</td>
</tr>
<tr>
<td>5.2.1 Lexical syntactic marking of node for compositionality modeling</td>
<td>91</td>
</tr>
<tr>
<td>5.2.2 Special Marking of specific grammatical phenomena</td>
<td>95</td>
</tr>
<tr>
<td>5.3 Mark-Up Rules for Constituency Trees</td>
<td>98</td>
</tr>
<tr>
<td>5.3.1 Binary branches</td>
<td>98</td>
</tr>
<tr>
<td>5.3.2 Flat or (n)-ary branches</td>
<td>102</td>
</tr>
<tr>
<td>5.4 Extending the SPTK with compositional information: the Compositional SPTK</td>
<td>104</td>
</tr>
<tr>
<td>5.5 Experimenting the CSPTK for Semantic Textual Similarity</td>
<td>110</td>
</tr>
<tr>
<td>5.5.1 Similarity functions</td>
<td>111</td>
</tr>
<tr>
<td>5.5.2 Experimental Setup</td>
<td>113</td>
</tr>
<tr>
<td>5.5.3 Evaluating STs features in an unsupervised scenario</td>
<td>115</td>
</tr>
<tr>
<td>5.5.4 Combining STSs with SV Regression</td>
<td>119</td>
</tr>
<tr>
<td>5.6 CSPTK for the Question Classification Task</td>
<td>122</td>
</tr>
<tr>
<td>5.6.1 General experimental setup</td>
<td>125</td>
</tr>
</tbody>
</table>
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.6.2 Results</td>
<td>126</td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>129</td>
</tr>
<tr>
<td>A Appendix A</td>
<td>135</td>
</tr>
</tbody>
</table>
List of Tables

2.1 Sense and Denotation in Frege, from [Penco(2004)] ........................................ 15
2.2 Componential Analysis, from [Penco(2004)] .................................................... 16
2.3 Generative rules, from [Penco(2004)] ................................................................. 18

4.1 A hypothetical semantic space for *buy*, *car* and *time* .............................. 49
4.2 Resulting compositional vectors for *buy-car* and *buy-time* by adopting 
   Mitchell&Lapata additive and multiplicative models .......................................... 49
4.3 Features corresponding to dimensions in the k=10 dimensional support 
   space of bigrams *buy car* and *buy time* ......................................................... 54
4.4 Example of Mitchell and Lapata dataset for the three syntactic relations 
   verb-object (VO), adjective-noun (AdjN) and noun-noun (NN) ............................. 60
4.5 Spearman’s ρ correlation coefficients across Mitchell and Lapata models 
   and the projection-based models proposed in Section 4.2. Topical 
   Space and Word space refer to the source spaces. is used as input to 
   the LSA decomposition model. ............................................................................ 62
4.6 Cross-linguistic dataset ....................................................................................... 65
4.7 Ambiguity reduction analysis .............................................................................. 68
4.8 Results over the STS task .................................................................................. 75

5.1 Unsupervised results of Pearson correlation for Kernel-based features 
   adopted in SEMEVAL Task 6 over the training dataset ...................................... 116
5.2 Unsupervised results of Pearson correlation for Kernel-based features 
   adopted in SEMEVAL Task 6 over the test dataset ............................................. 116
5.3 Unsupervised results of Pearson correlation for lexical features adopted 
   in SEMEVAL Task 6 over the training dataset .................................................. 119
LIST OF TABLES

5.4  Unsupervised results of Pearson correlation for lexical features adopted in SEMEVAL Task 6 over the test dataset ................................. 119
5.5  Results over the Core STS task ....................................................... 122
5.6  Results over the Core STS task ....................................................... 122
5.7  The coarse and fine-grained question classes ................................. 123
5.8  Results in the Question Classification task ..................................... 127
A.1  Complete list of POS elements ...................................................... 137
A.2  Complete list of GT elements ....................................................... 138
A.3  Complete list of Mark-Up rules ..................................................... 139
List of Figures

2.1 A simplified constituency representation for the sentence *Sam saw Kim* 17

2.2 Example of a constituency parse tree associated to sentence “*Cognac is a brandy made in Poitou-Charentes*” ............... 20

2.3 Example of a dependency graph associated to sentence “*Cognac is a brandy made in Poitou-Charentes*” .................. 21

2.4 Beta-reductions in the interpretation of the sentence *Sam saw Kim* .... 24

4.1 Cross-linguistic Gaussian distribution of infra (red) and inter (green) clusters of the proposed operators (continuous line) with respect to verbs only operator (dashed line) ......................... 67

4.2 Example of dependency graph ..................................................... 72

5.1 Charniak constituency tree of the sentences (a) *I make a gift to my friends* and (b) *I make room for my friends* .... 78

5.2 Compositional constituency tree of the sentences (a) *I make a gift to my friends* and (b) *I make room for my friends* .... 79

5.3 Some tree fragments obtained for the sentence “*Time flies like an arrow*” with the STK ......................................................... 84

5.4 Some tree fragments obtained for the sentence “*Time flies like an arrow*” with the PTK ......................................................... 87

5.5 Marking of a Compositional constituency tree of the sentences *A man is riding a bicycle*, derived through the Charniak parser ....... 93

5.6 Compositional Charniak constituency tree of the sentences *A man with glasses is talking* ....................................................... 96
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.7</td>
<td>Compositional enrichment of the constituency tree for the sentences A little girl is playing a grand piano on the stage from a tree derived by the Charniak parser</td>
<td>97</td>
</tr>
<tr>
<td>5.8</td>
<td>Marking the Charniak constituency tree of the phrase (a) big pepper and (c) a woman in the resulting (b) and (d) respectively compositional Charniak constituency tree</td>
<td>99</td>
</tr>
<tr>
<td>5.9</td>
<td>Marking the Charniak constituency tree</td>
<td>101</td>
</tr>
<tr>
<td>5.10</td>
<td>Marking the Charniak constituency tree</td>
<td>103</td>
</tr>
<tr>
<td>5.11</td>
<td>Lexical Centered Tree (LCT)</td>
<td>112</td>
</tr>
<tr>
<td>A.1</td>
<td>Charniak constituency tree of the sentences “I make room for my friends”</td>
<td>136</td>
</tr>
</tbody>
</table>
Introduction

Meaning is at once the most obvious and most mysterious feature of human language. Semantics is the study of meaning expressed by elements of a language, i.e. words. But the meaning of words can not be directly observed, it does not seem to be located in any place. The meaning of words or sentences has to be bind to things, abstractions or frame that they are referring to. More than 3,000 years of speculation by philosophers and linguists have failed to crack the central conundrum of meaning. Studies over composition in semantics, range from philosophical matter and attitudes to meaning to logical approaches. In order to understand the basis of theories and models of semantics, the necessary philosophical and linguistic background is presented in Sections 2.1 and 2.2.

Moreover, nowadays various and often mutually incompatible approaches have been proposed in order to represent meaning in a computationally tractable way. In symbolic approaches relations between entities in the world are exploited by the grammatical features of sentences, expressing the truth conditions of sentences. Orthogonally, distributional models of meaning are inspired by Firth’s “You shall know a word by the company it keeps”, modeling a term in function of its lexical co-occurences. Thus, in Chapter 3 distributional approaches to account lexical meaning are introduced.

Since distributional models seems to capture paradigmatic and syntagmatic phe-
Chapter 1. Introduction

nomena of isolated words, how to compose distributional representation of the lexicon in order to account the meaning of a sentence is still challenging. On the other hand qualitative symbolic approaches lacks of a semantic representation of individual words. In Chapter 4, state-of-art compositional distributional models to account the meaning of more complex structures like grammatical bigrams or entire sentences are discussed. Lexical co-relation and grammatical roles both correspond to different ways of “knowing how to use” language, thus both aspects provide relevant information when we understand the meaning of an expression. Hence they can be assumed as viable components for a new representation modelling natural language semantics.

Composition in distributional models, as well as the Thesis contribution in this direction, inspired by the philosophy of meaning as context and vector-based techniques are quickly presented in the next two Section, as an overview of this work.

1.1 Composition in Distributional Models

The analysis of the properties of words has been amplified by the development of statistical techniques applied over text corpora since 90’s. Distributional semantic models (DSM) are based on the assumption that the meaning of a word can be inferred from its usage, i.e. its distribution in text. The idea is to capture something about the nature of words that should be included in representations of their meaning. Vector-based semantic representations place meaning within the realm of mathematics and algebra: through a statistical analysis of the contexts in which words occur it is assumed possible to build semantic representations in form of high-dimensional vector spaces. Since the introduction of Landauer and Dumais in \cite{Landauer&Dumais1997} and Schütze in \cite{Schütze1998}, DSMs have been an active area of research in computa-
1.1. Composition in Distributional Models

tional linguistics and a promising technique for solving the lexical acquisition bottleneck by unsupervised learning. Over the years they have been used in various tasks, e.g. identify synonyms [Landauer and Dumais(1997)] [Rapp(2004)], construct thesaurus [Lin(1998b)], word sense induction and discrimination [Schütze(1998a)], model human similarity judgements [McDonald(2000)], semantic classification [Versley(2008)].

However, it is very difficult to reconcile these techniques with existing theories of meaning in language, which revolve around logical and ontological representations. According to logical theories [Kamp and Reyle(1993)] [Blackburn and Bos(2005a)], sentences should be translated to a logical form that can be interpreted as a description of the state of the world.

On the contrary, vector-based techniques are closer to the philosophy of “meaning as context”, relying on the Wittgenstein’s (1953) “meaning just is use” and Firth’s “you shall know a word by the company it keeps” and the distributional hypothesis of Harris (1968), that words will occur in similar contexts if and only if they have similar meanings. Model-theoretic philosophy of meaning describe natural language from the word level to the sentence level, while philosophy of meaning as context inspired vector based techniques.

In these years attention has been focused on the question of how to combine words representation in order to characterize a model for sentence semantics. Currently, there is no theory that explains how these vectors can be used to represent phrases and sentences, since these models are typically directed at the representation of isolated words. In [Duffy et al.(1989)] it has been demonstrated that the priming of sentence terminal words is due by a combination of content words within the lexicon, moreover [Mor-
Chapter 1. Introduction

ris(1994) showed how this priming has dependencies on the syntactic relations in the preceding context.

Composition in formal semantics has always been modeled in terms of function of words that operate on other words: the meaning of a sentence is incrementally build by applying these functions according with the syntactic structure [Montague(1974)]. Point of connection between syntactic types and functional types is represented by the adoption of the lambda calculus as a beta-reduction function in terms of substituting variables with words.

The most common approaches in the early literature on composition in distributional semantics has to be retrieved in [Foltz et al. (1998), Kintsch (2001), Landauer and Dumais (1997)], where words are “composed” by summing their vectors.

Symbolic logic approaches like in [Blackburn and Bos (2005a)] follow Montague by basing the process of semantic composition on syntactic structure as a function (i.e. a modifier) applied to an argument (i.e. an head). The same notion of function application has been proposed by [Coecke and Kissinger (2010)] and followed by [Grefenstette and Sadrzadeh (2011), Guevara (2010), Baroni and Zamparelli (2010)] in different ways. Two broad classes of composition models has been proposed by Mitchell and Lapata in [Mitchell and Lapata (2010a)] in which additive and multiplicative compositional models are investigated for a simple lexical bi-grams phrases similarity task.

1.2 Thesis contributions

Distributional models have been widely used in many Natural Language Processing tasks, i.e. Question Answering, Semantic Textual Similarity, Information Retrieval, achieving an effective representation and generalization schema of words in isolation.
1.2. Thesis contributions

As seen in Section 1.1, the composition of words in phrases or sentences is still a challenging task. Empirical distributional methods account for the meaning of syntactic structures by combining words according to algebraic operators (e.g. tensor product) among vectors that represent lexical constituents.

In this Thesis, a novel approach for semantic composition based on space projection techniques over the basic geometric lexical representations is proposed. In line with Frege’s context principle, the meaning of a phrase is modeled in terms of the subset of properties shared by the co-occurring words. Here this subset is exploit by the more representative features of a compound in a distributional space.

In order to account for a representation of simple syntactic bi-grams, a distributional compositional semantic model based on space projection guided by syntagmatically related lexical pairs is defined and discussed in Chapter 4. Syntactic bi-grams are projected in the so called Support Subspace: this geometric perspective aimed at emphasizing the semantic features shared by the compound words and better capturing phrase specific aspects of the involved lexical meanings. The approach presented here focuses on first selecting the most important components for a specific word pair in a relation and then modeling their similarity. This captures their meanings locally relevant to the specific context evoked by the pair. Compositional similarity scores are correspondingly derived by applying some basic metrics into the Support Subspaces identified for each lexical bi-gram.

Such that model has been tested first in the phrase similarity task proposed in [Mitchell and Lapata(2010a)] and seems very effective for the syntactic structures of VO, NN and AdjN, achieving state-of-the-art results in this task as described in Section 4.3. This represents a novel perspective on compositional models over vector rep-
representations with respect to shallow vector operators (e.g. additive, or multiplicative, tensorial algebraic operations) as proposed elsewhere.

In order to better assess and generalize the capability of the proposed model, further work on other compositional prediction tasks has been carried out. A first applicability study of such compositional models in typical IR systems was presented in Section 4.4. The operators’ generalization capability was measured proving that compositional operators can effectively separate phrase structure in different semantic clusters. The robustness of such operators has been also conformed in a cross-linguistic scenario, i.e. in the English and Italian Language, as discussed in Section 4.4.

In order to establish a measure of semantic similarity to support complex tasks of textual inference, our CDS model is used in a Semantic Textual Similarity task. In Section 4.5 the task is modeled as a Support Vector (SV) regression problem, where a similarity scoring function between text pairs is acquired from examples. The semantic relatedness between sentences is modeled in an unsupervised fashion through different similarity functions, each capturing a specific semantic aspect of the STS, e.g. syntactic vs. lexical or topical vs. paradigmatic similarity. Support Subspace compositional metric for lexical bi-grams is used together with other similarity functions, to train the SV regressor that combines the different models and learns a scoring function that weights individual scores in a unique resulting STS. This system provides a highly portable method as it does not depend on any manually built resource (e.g. WordNet) nor controlled, e.g. aligned, corpus.

Finally, in Chapter 5 we try to scale our DCS model to a sentence scenario in which vectors are no longer two (i.e. the lexical bi-grams), but three or more, making our metric sensitive to compositional syntax. For this model we take inspiration from the
1.2. Thesis contributions

theory of tree kernels [Collins and Duffy(2001)] and the extension proposed by [Croce et al.(2011c)], the idea is to exploit semantic smoothing over the lexical leaves of a parse tree, by using the recursive matching of a tree kernel enriched with a lexical-semantic (LSA) metric. The lexical representation in this view is distributed over all non-terminal nodes of a parse tree. Each non-terminal node is made corresponding to an head and a modifier, and these labels represent the core lexical information of that node. Head-modifier pairs are a surrogate of lexical bi-grams introduced in our DCS model of Chapter 4 in this way they can be composed by a Kernel function. Lexical syntactic marking of non-terminal nodes of a constituency parse tree is presented in Section 5.3, while the Compositionality extension of the SPTK (called CSPTK) introduced by [Croce et al.(2011c)] is defined in Section 5.4. This line of research is still in progress while first empirical results of the CSPTK in STS and QA tasks are presented in Section 5.5.
This chapter tries to cover the background necessary to deal in a computationally meaningful way with the problem of modeling, processing (i.e. extracting or generating) compositional forms of language semantics. We begin with a general view on the notion of semantics, differentiating a range of philosophical attitudes to meaning and identify some of the approaches which follow from these conceptions. A key contrast among these different models is the difference between symbolic and non-symbolic representations. While emphasizing this dichotomy we describe some of the vector binding mechanisms that attempt to bridge the gap between these paradigms. The discussion then focus on the topic of semantic composition, covering what is known about its function in natural languages as well as its modeling in a computational setting. Finally, existing distributional models of semantics are discussed in some depth as they are central for the contribution of this thesis. The way lexical vectors combine in order to account for compositionality in these models is finally examined.
2.1 Theories and Models of Semantics

Semantics is the study of meaning, and the question of what exactly a meaning is therefore forms part of its foundation. However, rather than there being a single agreed conception of what constitutes meaning, there are, in fact, a diversity of definitions and proposals.

The development of a theory of meaning inevitably requires subscription to a philosophy of what meaning is. We are interested in describing representations resulting from techniques that make use of context in order to determine meaning, therefore it is natural that we look for a philosophy in which meaning is closely connected to context. The closest we have found is in the ideas of Frege and Wittgenstein.

While we will discuss in the next section these core contributions to our idea of semantics, it is worth noticing here that traditional natural language processing research has not been always inspired directly by these authors, but has followed lines closer to general paradigms in computer science both inspired by automata theory, as pioneered in the studies of Noam Chomsky or knowledge representation from artificial intelligence. The influence of Chomsky is in fact the major responsible for the centrality of the grammar-based approaches to natural language processing, whereas linguistic competences have been seen as embedded in rule systems able to compute language through recursive generative mechanisms from finite resources (i.e. rule sets). On the other hand, AI pushed for representational approaches to syntax and semantics, whereas theorem proving as logical deduction has been elected as the core paradigm for ontological and epistemological reasoning. On the one side Chomsky emphasized questions such as competence and complexity issues of the underlying rule systems,
while expressivity of the formalisms for knowledge representation has been mostly studied in computational semantics as a core mechanisms governing language understanding, interpretation and use.

For these reasons we will begin our survey from logical models of semantics from which directly descend a large part of the current computational semantics literature.

2.1.1 Logic models of meaning

The most influential figure in the history of the project of systematizing the notion of meaning (in both of the ways outlined in the introduction) is Gottlob Frege. Frege set out to develop the idea of logic as a tool to analyze scientific languages as well as the natural languages. Today is not even conceivable to study natural languages and scientific theories without the aid of some logical-mathematical formalism. Frege’s work in the philosophy of language builds on what is usually regarded as his greatest achievement, the invention of the language of modern symbolic logic. In the logic developed by Boole, sentences were constructed from predicates using a small number of operators corresponding to traditional forms of judgment, such as universal affirmative judgments, which are of the form “All Fs are G”. Proper names, such as Socrates, were regarded as predicates, that is, as being of the same logical type as expressions like is mortal. Thus, the famous argument

\[
\text{Socrates is a man.} \\
\text{All man are mortal.} \\
\text{Therefore, Socrates is mortal.}
\]

might have been represented as:

\[
\text{All H are M.}
\]
All $S$ are $H$.

All $S$ are $M$.

The correctness of the argument then follows from the validity of the form of syllogism. Frege’s way of representing generality required him to reject this traditional identification of names and predicates. Frege, searching for universal the Leibnizian logical calculation framework, generalized the concept of function: a sentence is constructed from functions, with genetical objects as arguments and values. Thus the sentence “Socrates is mortal” is represented in his logic as: $M(s)$, where $M$ is the function is mortal, and the generalization “Everything is mortal” as: $M(x)$, where the singular term Socrates has been replaced by a variable. Frege unified the stoic and aristotelic traditions by inventing a formalism for generality expressions and extending the functional notion to terms like each and some. For the sentence “All humans are mortal”, Frege introduces the universal quantifier $\forall$ thus if the function $F$ represents “is a man” the syllogism becomes

$$F(s)$$

$$\forall x : F(x) \implies M(x)$$

$$M(s)$$

By introducing this new notation, especially the universal and existential (i.e. $\exists$) quantifiers, Frege work constitutes a huge advance on the syllogistic logic which had dominated philosophy since the time of Aristotle.

By constructing this new logic, Frege developed in [Frege(1960)] a conceptual content analysis, named sense. Sense is here that part of language that is relevant to the logic, thus for deduction. Rhetorical distinctions related to style, i.e. the tone or the
2.1. Theories and Models of Semantics

color of the language are unrelated to sense.

In this way sense is what you can grasp through different aspect of the same lan-
guage and through different languages. Frege most famous example relies on the prob-
lem of how explain the difference of cognitive values in identity statements like:

\[
a = a \\
\text{(2.1a)}
\]

\[
a = b \\
\text{(2.1b)}
\]

Truth of 2.1a which is analytic and a-priori is easily accepted by anyone; agree to
the truth of a statement with the form in 2.1b is more complicated. Frege pointed out
the matter by using the phrases \textit{morning star} and \textit{evening star}. These phrases have the
same referent, but different meanings. Using them in a sentence makes this difference
obvious, thus

\textit{The morning star is the morning star}

\textit{The morning star is the evening star}

The first sentence does not tell us anything new, while the second sentence does. A
referential theory of meaning does not predict this difference, since the terms \textit{morning
star} and \textit{evening star} indicate both Venus, the last luminous planet to disappear in
the morning and the first to appear in the evening. Frege’s solution requires to find a
difference in \textit{cognitive significance} between \(a = a\) and \(a = b\), consisting in a difference
in the way in which the designated objects are given, thus a difference in the \textit{mode of
presentation}.

According to Frege an assertion of identity like \(a = b\) implies that the same object
is presented in three different ways: the sign or the linguistic expressions (i.e. the
name), the **sense** (i.e. the mode of presentation) and the **denotation** (i.e. the object it picks out).

By holding that every name has both a sense and a denotation, Frege actually holds that every expression has both a sense and a denotation, by defining the **thought** as the sense of a sentence and the **truth-value** as his reference, connecting meaning and truth. Thus, the sense of a sentence, is what we grasp when we understand an utterance, that is, the conditions under which it is true. This idea will become central to the philosophy of language and in the formal semantics.

> *If we replace a name occurring in a sentence with another name having the same denotation, but a different sense, the denotation remains the same; but we realize that in such cases the proposition is changed*

Take for example the following two sentences that must have the same denotation, but clearly express different propositions:

*The morning star is a celestial body illuminated by the sun*

*The evening star is a celestial body illuminated by the sun*

Since these two sentences do not differ with respect to the denotation of any of their parts, their denotation will be the same. Therefore, the different propositions they contain must be the respective senses of the two sentences. Frege concludes that each type of linguistic expression in his logic has a sense and a denotation as in Table 2.1.

In order to define the sense and the denotation of an utterance, Fregean semantics relies on an assumption called Frege’s principle or **principle of compositionality** for which
2.1. Theories and Models of Semantics

Table 2.1: Sense and Denotation in Frege, from [Penco(2004)]

<table>
<thead>
<tr>
<th>Sense</th>
<th>Proper Noun</th>
<th>Predicate</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>The way of giving the denotation</td>
<td>The way of giving the denotation</td>
<td>Thought</td>
<td></td>
</tr>
<tr>
<td>Denotation</td>
<td>Object</td>
<td>Concept</td>
<td>True value</td>
</tr>
</tbody>
</table>

The denotation (sense) of a complex expression (including a sentence) is a function of (i.e., is determined by) the denotations (senses) of its constituent expressions.

In order to prove that, Frege uses the Leibniz principle of substitutivity that state:

*If in a true statement (or sentence) some expression e is replaced by (i.e. substituted with) an expression co-referential with it (i.e. with an expression sharing the referent with e), a true statement (sentence) results.*

Thus identical expressions (with the same reference) can replace one another without prejudice the truth of everything. If in the statement “the morning star is a planet” we replace a term with the same reference, for example evening star, the reference of the whole (i.e. the utterance truth-value) does not change. The principle of compositionality and the law of substitutivity still work for statements, the truth-value of a statement depends on the truth-value of the components, and by replacing an utterance with another coreferential (i.e. with the same truth value) the truth of everything does not change.

Notice that the above perspective on semantics is not the only one. It has a strong
ground in logics according to an informational account of meaning as truth condition and a correspondingly strong root in the AI literature regarding representations as an acceptable psychological and representational account of meaning.

### 2.1.2 Semiotic and linguistics

Next to the Frege’s logic, other formalisms to account the representation of language have been derived from linguistic and semiotic. Saussure in the early years of ’900 spoke about language as a structure in which each element has a role and a place. Each lexical item corresponds to a form and a content (i.e. for Saussure a signifier and a meaning). The study of lexical meaning has been developed with the structuralism and the componential analysis, consisting in decompose word meanings in minimal elements called semantic primitives, as shown in Table 2.2.

<table>
<thead>
<tr>
<th></th>
<th>Male</th>
<th>Adult</th>
<th>Human</th>
</tr>
</thead>
<tbody>
<tr>
<td>Man</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Woman</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Baby-boy</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Baby-girl</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 2.2: Componential Analysis, from [Penco(2004)]

Another trend that moves away from structuralism is based on the study of cognitive processes where we admit the existence of common conceptual component among the human being. This kind of ideas are the basis of the Fillmore frame semantics, that will affect cognitive semantics.
2.1. Theories and Models of Semantics

While in Europe has been developing structuralism, in the United States a new revolution in linguistics relies on the development of logic was carried on by Noam Chomsky. Chomsky’s path-breaking theory occasioned the reconstruction of language as a formal algebraic structure [Chomsky(1956)]. Chomsky proposed to account for a language via a set of formal generative rules, the recursive application of which to a given initial symbol generates all and only syntactically well-formed sentences of the language. Chomsky’s novum was that he proposed organizing the rules into a hierarchical system allowing for systematical generation, and basing all this upon setting up of the grammar as a real mathematical structure. Chomsky introduced a formal system whose vocabulary consists of lexical items and theoretical symbols (i.e. non-terminal symbols). For example $S = \text{Sentence}$, $NP = \text{Noun Phrase}$, $VP = \text{Verb Phrase}$, $N = \text{Name}$, $V = \text{Verb}$ and $Art = \text{Article}$. Sentences derived from generative rules form the set of nuclear the sentences of the language, from which more complex sentences are derived, through transformation rules.

In Figure 2.1 the sentence “Sam saw Kim” is structured in form of a tree as a synthetic representation of the application of the generative rules listed in Table 2.3.

The tree representation is also useful to disambiguate ambiguous sentences, by

---

**Figure 2.1:** A simplified constituency representation for the sentence *Sam saw Kim*
Chapter 2. A survey on Semantics, Natural Language and Computation

<table>
<thead>
<tr>
<th>Generative Part</th>
<th>Transformational Part</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VOCABULARY:</strong></td>
<td><strong>AXIOMS:</strong></td>
</tr>
<tr>
<td><em>non-terminal symbols:</em></td>
<td>nuclear phrases</td>
</tr>
<tr>
<td>S, NP, VP, N, V, Det</td>
<td></td>
</tr>
<tr>
<td><em>terminal symbols:</em></td>
<td></td>
</tr>
<tr>
<td>Sam, saw, Kim, ...</td>
<td></td>
</tr>
<tr>
<td><strong>GENERATIVE RULES:</strong></td>
<td><strong>TRANSFORMATION RULES:</strong></td>
</tr>
<tr>
<td>(rewriting rules)</td>
<td>(X - V active - Y)</td>
</tr>
<tr>
<td>S → NP + NV</td>
<td>→</td>
</tr>
<tr>
<td>NP → Det + N</td>
<td>(Y - V passive - from X)</td>
</tr>
<tr>
<td>NP → KimSam</td>
<td>. . .</td>
</tr>
<tr>
<td>VP → V + NP</td>
<td></td>
</tr>
<tr>
<td>Det → The</td>
<td></td>
</tr>
<tr>
<td>V → saw</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.3: Generative rules, from [Penco(2004)]

discerning the deep structure from the surface structure.

2.2 Syntactic representation and its role in modeling semantic information

Grammatical analysis does not focus only on lexical items as isolated events, as language imposes constraints on word order. Syntax is the study of such regularities and constraints. One fundamental idea is that words are organized into phrases, groupings of words to form the so-called constituents. For example, nouns can be generalized by large fragments called Noun Phrase (i.e. NP): these include single words like Cognac.
2.2. Syntactic representation and its role in modeling semantic information

or Poitou-Charentes as well as phrases like a brandy. This structure can be recursively grouped in order to define a complex constituency structure of the sentence, e.g. a brandy made in Poitou-Charentes. The most commonly used mathematical systems for modeling constituent structures in English and other natural languages are based on the notion of formal grammars. The idea of basing a grammar on constituent structures was first formalized in [Chomsky(1956)], thus providing the operational definition for the combinatorial formations of meaningful natural language sentences by humans, as introduced in 2.1.2. One of the most used formal grammar in Computational Linguistics is the Context-Free Grammar (CFG), that consists of a set of rules or productions that express the ways symbols can be grouped and ordered together. Moreover it constraints also the use of words: for example, the following productions express that a NP can be composed of either a proper noun NNP or a determiner (DT) followed by a NN; this can be expressed by the following CFG rules: $NP \rightarrow NNP \mid DT \, NN$. The symbols used in a CFG are divided into two classes. The symbols that correspond to words in the language (Cognac, a, or brandy) are called terminal symbols; the lexicon is the set of rules that introduce these terminal symbols. Nonterminal symbols, or just nonterminals, are the symbols which can be replaced. In each context-free rule, the item to the right of the arrow ($\rightarrow$) is an ordered list of one or more terminals and non-terminals, while to the left of the arrow there is a single non-terminal symbol. A CFG includes a start symbol, here $S$, a designated member of the set of nonterminals from which all the strings in the language may be derived by successive applications of the production rules. The language defined by a CFG is precisely the set of terminal strings that can be so derived. Constituency parse trees are usually employed to represent all context-free rules activated by a sentence: Figure 2.2 shows the tree generated from the syntac-
Chapter 2. A survey on Semantics, Natural Language and Computation

Figure 2.2: Example of a constituency parse tree associated to sentence “Cognac is a brandy made in Poitou-Charentes”

tic analysis of the sentence “Cognac is a brandy made in Poitou-Charentes”. Notice that each pre-terminal reflects the Part-of-Speech of the covered word, while the non-terminal nodes are provided by CFG rules such as $\text{NNP} \rightarrow \text{Cognac} | \text{Poitou-Charentes}$, $\text{VP} \rightarrow \text{VBN PP}$ or $\text{VBN} \rightarrow \text{made}$.

All these rules, and many others, are expressed together in this complete syntactic structure that explains the entire sentence and provides a set of tree fragments, i.e. the building blocks of the constituency parse tree. Each tree fragment can be exploited as a feature of our learning method as it specified the (grammatical) relations among different words. As an example, in order to answer to question “What French province is Cognac produced in?”, we can notice that the $\text{PP} \rightarrow \text{IN NP}$ fragment actually covers the target answer, i.e. in Poitou-Charentes, and it is connected to the $\text{VP}$ structure that covers the verb $\text{made}$. The objective of a learning model would be to determine the Parse Tree Path that connect the $\text{PP}$ node to the $\text{made}$ leaf to effectively retrieve the target information. The sentence syntactic structure can be thus exploited providing to language learning systems the information extracted from the parse trees of training
2.2. Syntactic representation and its role in modeling semantic information

![Dependency Graph Example](image)

Figure 2.3: Example of a dependency graph associated to sentence “Cognac is a brandy made in Poitou-Charentes”

Even if the constituency representation is well-founded on linguistic theory, and many annotated corpora have been produced, e.g. the Penn Treebank introduced in [Marcus et al. (1993)], as well as many constituency parser, e.g. [Collins (1997), Manning and Schütze (1999), Charniak (2000)], in the last decade a different grammar formalism called Dependency grammar became quite important in language processing, as discussed in [Nivre (2006)]. In such formalism, no attention is paid to constituents and phrase structure rules. Instead, the syntactic structure of a sentence is described purely in terms of words and binary syntactic relations between these words. Dependency grammars often draw heavily from the work of Tesnière in [Tesnière (1959)].

As shown in Figure 2.3, the resulting structure is not a tree anymore, but a graph, and it is called Dependency Graph. Notice that no non-terminal or phrasal nodes are considered; each link in the parse tree holds between two lexical nodes (augmented with the special **ROOT** node). Here the graph consists of edges between words, and the presence of an edge between word pairs denotes a grammatical cooperation between those words, like for example *Cognac* and the verb *is* or the determiner *a* and *brandy*. The edges are directed, and the direction of an edge between two words denotes which
of them determines the grammatical behavior of the complete structure. In each relation the word from which the edge starts is called governor, while the pointed one is called dependent: for example, the word is represents the governor while Cognac is the dependent. The edges are labeled, and the label associated with an edge between two words denotes the nature of their grammatical relationship, or grammatical function, that represents a first form of semantic relation among words, as Cognac is the Logical Subject (SUBJ) of the sentence main verb. The syntactic structure provides an interesting source of information in order to understand how the question “What French province is Cognac produced in?” is associated to the example sentence. The locative prepositional construction (LOC) label that relates the verb made with the graph partition containing in Poitou-Charentes is a more explicit hint of the semantic relation between with the verb, i.e. a propositional construction. Dependencies with labeled edges and constituents with labeled phrases can be seen as representations of two different views of the organization of a sentence, where either the nature of cooperation or the hierarchical organization is made explicit.

2.2.1 Computational account of compositionality in natural language semantics

Compositionality is an essential property for the adoption of a full truth-functional view on meaning representation within the Fregean program. In this view, the meaning of any structured linguistic expression corresponds to a (functional) composition of the meanings related to the individual composing structures. This reflects both the representational view on natural language semantics, as compositions can occur between psychologically plausible representations of the component sub-structures, and the informational view whereas all the referents of the substructures combine in a truth
2.2. Syntactic representation and its role in modeling semantic information

functional manner. In any rigorous account of natural language meaning compositionality plays the role of a cornerstone, even where controversial theories about the basic building blocks (e.g. lexical semantic interpretation of words or terminological expressions) are still developed.

A significant by-product of a truth-functional approach to NL semantics is represented by the computational semantics research program (e.g. corresponding to the perspective pursued in Blackburn and Bos(2005b)). Computational semanticists design the interpretation of a linguistic utterance in terms of a syntax-driven deduction process aimed at formally correspond to the former. Here lambda calculus provides the formal language to express:

- the representation of the lexical elements, i.e. the formulas corresponding to the meaning of individual lexical entries (e.g. $n$-ary predicates for verbs)
- the logical machinery in which deduction steps corresponds to beta-reductions among lambda expressions corresponding to complex structures within a sentence, such as the projections of non terminal nodes in the syntactic representation
- a systematic reference ontology in which: (1) lexical rules and axioms are distinct components of a comprehensive theory of the grammar and lexicon, and (2) semantics is build in strict agreement with the syntactic structure postulated by an underlying grammar

In the lambda calculus formalisms, the combinatorics of the formal language let the semantic interpretation of complex expressions correspond to a chain of beta-reductions, i.e. individual composition steps applied to the underlying sub-expressions.
In Fig. 2.4 we report the interpretation of the simple declarative sentence, i.e. *John saw Kim*, as through beta-reductions acting over the constituency-based syntactic representation, i.e. the parse tree, of Fig. 2.1.

Here lexical rules, such as the fact that Sam is a ground symbol representing the individual called *Sam* or the verbal predicate that corresponds to the verb *saw*, are captured as unification in specific pre-terminal nodes. Correspondingly more complex structures related to non-terminal nodes in the parse tree (e.g. partially saturated VPs), are obtained through beta-reduction steps that are responsible for the slot filling process. As a consequence, every non terminal node in the tree (as part and consequence of the syntactic theory about the underlying (English) language) is made corresponding to a partial meaning representation structure, i.e. an individual, a set of individuals or a ground formula. For example, the VP node in the tree corresponds to the formula $\lambda x. \text{saw}(x,k)$, i.e. the set of all entities that saw Kim. Formulas provide a representation of the possibly complex sense of a sentence, while making available ref-
2.2. Syntactic representation and its role in modeling semantic information

erence information through the interpretation of individual symbols within the target world/context.

The process here outlined allows thus to support a full logical treatment of the linguistic inferences (e.g. entailment between logic forms) or extra-linguistic decisions (e.g. truth values of a sentence obtainable through the observation of a reference world or the actions corresponding to imperative statements) implied by an utterance.

Although controversial views on the correct treatment of specific linguistic phenomena (e.g. presuppositions depending on lexical properties) are still faced by current linguistic research ([Chierchia and McConnell-Ginet(2000)]), the truth functional perspective implicit in the Fregean program and here surveyed, is still the backbone of all modern semantic language processing systems, even when strong non-symbolic components/tools (such as statistical parsers or semantic role labeling systems) are applied in the language processing chain.
As introduced in Chapter 2, Frege’s principle asserts that the meaning of a complex expression is determined by the meanings of its parts and the way in which those parts are combined. According to this principle to understand a sentence is to understand the meaning of its smallest components (i.e. words) and the way they are combined (i.e. the syntactic structure). Indeed, by applying the principle of *semantic compositionality* cited in Chapter 2, a function over the meaning of each atomic constituent and the mode of their combination is required in order to reconstruct the meaning of a complex expression. In this way model of meaning has been focused on atomic units (i.e. words or lemmas): this either being seen as a necessary step towards conquering compositional meaning, or having provided sufficient gains in its own right to distract from what is generally considered a harder problem. Thus an effective way to represent lexical meaning and a model to combine words representation together which takes account of their syntactic relations are required.

Distributional approaches to meaning acquisition have received many attention in the last decades and their empirical implementation have achieved some degree of success in modeling lexical semantics. These approaches rely on distributional properties of linguistic entities as the building blocks of semantics. Fundamentally these ap-
proaches are based on a set of assumptions about the nature of language and meaning referred to as the distributional hypothesis.

This hypothesis is often stated in terms like words which are similar in meaning occur in similar contexts [Rubenstein and Goodenough(1965)]; words with similar meanings will occur with similar neighbors if enough text material is available [Schutze and Pedersen(1995)]; a representation that captures much of how words are used in natural context will capture much of what we mean by meaning [Landauer and Dumais(1997)]; and words that occur in the same contexts tend to have similar meanings [Pantel(2005)], just to quote a few representative examples.

By equating meaning with context, in empirical lexical semantics it is simply possible to record the contexts in which a word occurs in a collection of texts (a corpus) to create a summary of the distributional history of the word that can then be used as a surrogate of its semantic representation.

This idea has its theoretical roots in various traditions, including American structuralist linguistics, British lexicology and certain schools of psychology and philosophy like in [Wittgenstein(1953), Harris(1954), Firth(1961)].

The main statement behind the distributional hypothesis seems clear enough: there is a correlation between distributional similarity and meaning similarity, which allows us to utilize the former in order to estimate the latter. This seems somewhat at odds with the Frege’s principle; whereas the latter links the meaning of an utterance to its internal context, the former focuses on the external; together these create an apparently circular delegation of responsibility concerning the residence of meaning.

The most direct realization of the distributional hypothesis in computational linguistics are the so called Distributional Semantic Models (DSMs) (Schütze(1998b)).
3.1 Distributional Representation of Word Meaning

Landauer and Dumais (1997), Turney and Pantel (2010), Sahlgren (2006), Clarke (2012), Erk (2012), Lund et al. (1995). These are automatic approaches to acquire and generalize lexical information directly from data. Such acquisition is managed through the distributional analysis of large scale corpora. Linguistic phenomena, here words, are modeled according to a geometrical perspective, i.e. points in a high-dimensional space representing semantic concepts, in such a way that similar, or related, concepts are near each another in the space. In a DSM, each word is represented by a mathematical vector. The values in the vector components are functions of the number of times that the words occur in the proximity of various linguistic contexts in a corpus.

In the rest of this chapter, Section 3.1 introduces the notion of Distributional Models of Lexical Semantics. Different approaches to acquire meaningful models for semantic compositionally are discussed in Section 3.2 while algebraic methods for space dimensionality reduction are discussed in Section 3.3.

3.1 Distributional Representation of Word Meaning

It is widely accepted that lexical information (as features directly derived from word forms) is crucial for acquiring meaningful information from texts. In Information Retrieval (IR) and Natural Language Processing (NLP) tasks, the lack of a proper lexical generalization is one of the main causes for performance drops when large scale data collections are targeted. For example, in a IR system a user can express its specific user need with a natural language query like “... buy a car ...”. This request can be satisfied by documents expressing the abstract concept of buying something when, in particular, the focus of the action is a car. This information can be expressed inside a document collection in many different forms, e.g. the quasi-synonymic expression “... purchase
an automobile . . .”. According to the lexical overlap with respect to original query, a bag-of-word based system would instead retrieve different documents, containing expressions such as “. . . buy a bag . . .” or “. . . drive a car . . .”. A proper semantic generalization is thus needed, in order to derive the correct composition of the target words, i.e. an action like buy and an object like car.

Another example is represented by the Semantic Textual Similarity (STS) task, in which is required to understand the similarity rate between sentences. For example sentences like “i made a toy for my friends” and “i fabricated a toy for my friends” need to be related since the verb to make is here used in the meaning of to fabricate something. Notice that all the other words are the same, thus the discriminant factor is delegated to the meaning of the two verbs. The more the representation of the two verbs are similar, the more is the global similarity rate for the sentences. Therefore a semantic generalization of the meaning to make and to fabricate is required. By introducing a third sentence, extremely different from the first one, like “i make room for my friends”, the idiomatic expression to make room has to be distinguished in meaning from the action to make toy. Here a bag-of-word based system discriminate the two sentences only by the discriminative power of the representation of words room and toy, considering the verb make identical in both the sentences. A semantic discrimination between the verb to make used together with the objects room and toy is further required in order to distinguish between two acception of the same verb.

Although large scale general-purpose lexicons are available, e.g. WordNet [Miller(1995)], their employment in specific tasks is not satisfactory: coverage in domain (or corpus)-specific tasks is often poor and domain adaptation is difficult. Moreover, differences in vocabulary and writing style across domains can cause one produced resource to be
3.1. Distributional Representation of Word Meaning

poorly representative of a new domain. These lexicons then suffer from lack of specific information, that is often the main responsible of significant drops in out-of-domain classification tasks. Consequently, every statistical model that exploits such information will have a limited generalization capability. The reusability and adaptation of a model dependent from such information would be largely reduced.

However, in many tasks a complete description of the words involved in the target problem is not even required. A meaningful model of a linguistic phenomenon can be acquired just according to an effective notion of similarity (the kernel function) among observations (the training examples) [Collins and Duffy(2001)]. The same idea is here applied to model the lexical semantic information. The focus is not to provide a representation explaining the behavior of every word, but instead a computational model able to describe semantic relationships among words that can be effective in complex NL inferences.

Moreover, the quality of the lexical information would be effective if it can be adapted to different domains, i.e. reflecting the specific information needed for a target task. These requirements will be considered to provide a method that automatically acquires lexical information from the statistical analysis of the word distribution in (domain specific) texts, thus reflecting the language used for the target task.

Distributional approaches represent lexical semantics through the analysis of observations in large-scale corpora. The idea is to acquire an artificial representation of a target word \( w \), considering all other words co-occur with \( w \), such that two words sharing the same co-occurrences will be represented in a similar manner. A lexical similarity function can be thus defined in terms of similarity between these representations. Notice that a good approximation of the words distributional information can
be achieved if a sufficient amount of observations is gathered. Several large scale corpora can be exploited in English, e.g. the British National Corpus (BNC) \cite{Aston and Burnard(1998)} made of 100 million words, the GigaWord \cite{Graff(2003)}, made of 1.75 billion words or the ukWaC corpus \cite{Baroni et al.(2009)}, made of 2 billions word. Other corpora are available also for other languages, e.g. itWaC a 2 billions word for Italian. In this thesis, a distributional representation of words is acquired according to a geometrical perspective, i.e. words are represented as vectors whose components reflect the corresponding contexts. This allows to define a high-dimensional space known as \textbf{Word Space}, where the distance among instances (i.e. words) reflects the lexical similarity, as described in \cite{Schütze(1993)}:

\begin{quote}
\textbf{Vector similarity is the only information present in Word Space: semantically related words are close, unrelated words are distant}
\end{quote}

The word-space model is a spatial representation of word meaning. Words are vectors represented by points in this space and if two words have similar contexts, they will have a similar representations and they will be close in the space. As stated in \cite{Sahlgren(2006)} entities need to occupy (different) locations in a conceptual space in order to possess spatiality (i.e. \textit{entities-are-locations}) and they need to be close each other to be similar (i.e. \textit{similarity-is-proximity}):

\begin{quote}
\textbf{Meanings are locations in a semantic space, and semantic similarity is proximity between the locations}
\end{quote}

From a linguistic perspective, they are likely to be related by some type of generic semantic relation, either paradigmatic (e.g. synonymy, hyperonymy, antonymy) or syntagmatic (e.g. meronymy, conceptual and phrasal association).
3.2 Designing Word Spaces for semantic composition

Semantic spaces have been used for over a decade, demonstrating their quality in numerous tasks and applications: they have been widely used for representing the meaning of words or other lexical entities, as discussed in [Pado and Lapata(2007)], [Basili and Pennacchiotti(2010)] and [Turney and Pantel(2010)], with successful applications in lexical disambiguation, as in [Schütze(1998b)], harvesting thesauri, as in [Lin(1998a)] and Name Entity Classification, as in [Zanoli et al.(2009)].

3.2 Designing Word Spaces for semantic composition

In Word Spaces the notion of semantic relatedness between words is a function of the distance between their geometrical representation in the space. From a computational perspective, a matrix $M$ is defined, whose rows describe words as vectors $\vec{w}_i$, columns describe the corpus contexts $\vec{c}_j$ and each entry $w_{ij}$ is a measure associating words and contexts. Given two words $w_1$ and $w_2$, the term similarity function can be estimated according to the Euclidean Distance or the Cosine Similarity between the corresponding projections $\vec{w}_1, \vec{w}_2$, i.e

$$\cos(w_1, w_2) = \frac{\vec{w}_1 \cdot \vec{w}_2}{\|\vec{w}_1\|\|\vec{w}_2\|}$$  \hspace{1cm} (3.1)

that measures of the angle between such vectors.

One open issue is that a definition of $c_j$ and the association measure’s estimation has not yet been addressed. This problem is not trivial as different semantic aspects of the involved words are considered by changing the representation space. For example, by employing different notions of context, we can assume two words similar when they appear in the same documents [Salton et al.(1975)], [Landauer and Dumais(1997)] or in the same sentences (modeled as word co-occurrences in short windows [Sahlgren(2006)]) or even in the same syntactic structures [Pado and Lap-
Chapter 3. Distributional Approaches to Language Semantics

ata(2007). Obviously, different context types define geometric spaces with different semantic properties and different generalization grains in the resulting similarity estimation.

Moreover, different NLP tasks require different types of lexical generalization. A wider context will provide a shallower generalization while a smaller one will capture more specific lexical aspects of words, as well as their syntactic behavior. In a typical Information Retrieval task, i.e. document classification task, where the aim is to map each document in a class reflecting the text topic (e.g. sport, economy or science), a topic-oriented form of similarity (i.e. topical similarity) is thus required. Such a model can be employed to relate words like “bank”, “acquisition”, “purchase” “money” or “sell”, as they address one single “economic” topic. On the contrary, paradigmatic relations could be more appropriate for other tasks like Semantic Textual Similarity, since a fine grain representation of lexicon is desirable in order to foster synonyms or hypernyms phenomena in the resulting vector space, avoiding topical clusters that could be too much vague. As for the example above, the aim is to relate together verbs like to make and to fabricate, thus similar words in these kind of spaces must share at least one acception typical of their own grammatical type. Finally in our view, to govern lexicon compositionality, a DM must be able to collect all the paradigmatic acception of a word in order to select the one typical of the composition, i.e. by selecting the acception of to build up something for the phrase make toy, and on the contrary by selecting the acception of to leave a space for the phrase make room. The more a DM specify and represent all the lexicon acception, the more is useful to select the right sense of a composition, thus a paradigmatic representation is still required.

In the following sections, two different kinds of context are investigated:
3.2. Designing Word Spaces for semantic composition

- the Topical space;

- the Word-based space;

3.2.1 Topical Space

A document-based geometric space represents words by focusing on coarse grain textual elements, capturing contextual information by expressing the distribution of words across documents [Salton et al.(1975)]. Two words will have a similar geometric representation if they tend to appear in the same documents of a corpus. In Information Retrieval this notion is usually employed to represent texts via linear combinations of (usually orthonormal) vectors corresponding to their component words. This space can be computationally represented as a so-called word-by-document matrix having as many rows as the number of (unique) target words to represent, and having as many columns as the number of different documents in the underlying corpus. Individual scores, associating words and documents, are computed according the term frequency-inverse document frequency ($tf-idf$) schema, [Salton et al.(1975)]. The $tf-idf$ value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control for the fact that some words are generally more common than others. As a result, two words tending to occur in the same documents will have a similar set of component initialized (i.e. value in the same columns). In such way words like “bank” or “acquire” have the same representation because they tend to appear in documents concerning the same economical topics, thus sharing a topical relation.
Chapter 3. Distributional Approaches to Language Semantics

3.2.2 Word-based Space

We will refer to the so-called word-based spaces, in which target words are represented by gathering probabilistic information of their co-occurrences calculated in a fixed range window over all sentences.

This particular space aims at providing a distributional lexical model while capturing paradigmatic relations between lexicon. Paradigmatic relations concern substitution, and relate entities that do not co-occur in the text. It is a relation in absentia and holds between linguistic entities that occur in the same context but not at the same time, like “make” and “built” or like “car” and “vehicle”. Paradigmatic relations are substitutional relations, which means that linguistic entities have a paradigmatic relation when the choice of one excludes the choice of another. A paradigm is thus a set of such substitutable entities.

In such that models vectors represents a target word \(tw\), while vector components correspond to the entries \(f\) of the vocabulary \(V\) (i.e. the features that are individual words). Dimensions with a not-null contributions are words appearing in a \(n\)-windows around the \(tw\) [Sahlgren(2006)]. To better understand, let us consider the adjectives beautiful, attractive and pretty. They are synonyms, i.e. words that can be mutually exchanged in texts, in most cases without altering the corresponding meaning, e.g. in phrases like “the beautiful girl”, “the attractive girl” or “the pretty girl”. Just considering this small examples, we can notice that these words co-occur with the word girl. If synonyms can be exchanged in the language in use, in a large-scale document collection they will tend to co-occur with the same words. If vector dimensions correspond to words in the corpus, in a Word-based space \(tw\)s co-occurring with the same set of words are similarly represented, having initialized almost the same set of geometrical
3.2. Designing Word Spaces for semantic composition

components. This is not valid only for synonyms, as words involved in a paradigmatic relation have the same properties. If two words like knife of rifle can be exchanged in texts, they share a consistent subset of co-occurring words.

Then, in this words-by-words matrix each item is a co-occurrence counts between a tw (a row) and other words in the corpus, within a given window of word tokens. The window width $n$ is a parameter allowing the space to capture different lexical properties: larger values for $n$ tend to introduce more words, i.e. possibly noisy information, whereas lower values lead to sparse representations more oriented to paradigmatic properties. Moreover, in order to capture a first form of syntactic information, words co-occurring on the left are treated separately from words co-occurring on the right. It allows, for example, to provide a better representation for transitive or intransitive verbs. In a sentence like “the beautiful girl entered the bar”, we say that beautiful co-occurs with the in a left widow of size one, with girl in a right window of size one, with entered in a right window of size two, with the in a right windows of size three and bar in a right window of size four. In some works (e.g. [Mitchell and Lapata(2008)]) pure co-occurrence counts are adopted as weights for individual features $f_i$, where $i = 1, ..., N$ and $N = |V|$; in order to provide a robust weighting schema and penalize common words, whose high frequency could imply an unbalanced representation, Pointwise Mutual Information (PMI) [Fano and Hawkins(1961), Turney and Pantel(2010)] scores are here adopted as in (e.g. [Pantel and Lin(2002)]), thus

$$pmi(w, i) = \log_2 \frac{p(w, f_i)}{p(w) \cdot p(f_i)}$$

$i = 1, ..., N$

A vector $w = (pmi_1, ..., pmi_N)$ for a word $w$ is thus built over all the words $f_i$ belonging to the dictionary. When $w$ and $f$ never co-occur in any window their $pmi$ is by default set to 0.
3.3 Embedding Lexical Semantics in lower dimensional spaces

The quality of a Word Space is tied to the amount of information analyzed and the more contextual information is provided, the more accurate will be the resulting lexical representation. However, some problems of scalability arise when the number of the space dimension increases. From a computationally perspective, a space with thousand dimensions make the similarity estimation between vector expensive. Consequently, even a simple operation, e.g. the search of the most similar words to a target word can be prohibitive. Moreover, from a geometric perspective, the notion of similarity between vectors is sparsely distributed in high-dimensional space. It is known as the curse of dimensionality. [Bengio et al.(2005)]: in this scenario, the higher is the number of dimensions, the lower is the variance of distances among data, reducing the expressiveness of this information for further inferences.

Fortunately, employing geometric representation for words enables the adoption of dimensionality reduction techniques to reduce the complexity of the high-dimensional space. Such techniques allow to exploit data (i.e. words and contexts) distribution and topology in order to acquire a more compact representation and more meaningful data-driven metrics. The main distinction between techniques for dimensionality reduction is the distinction between linear and nonlinear techniques. Linear techniques assume that the data lie on or near a linear subspace whose dimensions are smaller than the original space. Nonlinear techniques instead assume that data lie on an embedded nonlinear manifold within the higher-dimensional space [Lee and Verleysen(2007)].

Latent Semantic Analysis [Deerwester et al.(1990), Landauer and Dumais(1997)] is an example of linear dimensionality reduction technique and uses the Singular Value
3.3. Embedding Lexical Semantics in lower dimensional spaces

Decomposition [Golub and Kahan(1965)] to find the best subspace approximation of the original word space, in the sense of minimizing the global reconstruction error projecting data along the directions of maximal variance.

In this thesis, LSA is applied to capture second order dependencies between features $f$, i.e. applying semantic smoothing to possibly sparse input data.

3.3.1 Latent Semantic Analysis

An example of linear dimensionality reduction technique is Latent Semantic Analysis [Landauer and Dumais(1997)]. The original word-by-context matrix $M$ is decomposed through Singular Value Decomposition (SVD) [Golub and Kahan(1965)] into the product of three new matrices: $U$, $S$, and $V$ so that $S$ is diagonal and $M = USV^T$. $M$ is approximated by $M_k = U_kS_kV_k^T$ in which only the first $k$ columns of $U$ and $V$ are used, and only the first $k$ greatest singular values are considered. This approximation supplies a way to project a generic term $w_i$ into the $k$-dimensional space using $W = U_kS_k^{1/2}$, where each row $w_k^i$ corresponds to the representation vectors $w_i$. The original statistical information about $M$ is captured by the new $k$-dimensional space which preserves the global structure while removing low-variant dimensions, i.e. distribution noise.

The lexical similarity can still be computed in such reduced space with the cosine similarity expressed in Equation 3.1 in a space with a reduced number of dimensions (e.g. $k = 100$) where the notion of distance is more significant with respect to the original space. These newly derived features may be considered latent concepts, each one representing an emerging meaning component as a linear combination of many different original contexts.
Chapter 3. Distributional Approaches to Language Semantics

It is worth noticing here that the application of SVD to different spaces results in very different latent topics. The emerging of special directions in the space as caused by different linguistic contexts (e.g. from documents to short windows around words) has thus significantly different linguistic implications. When larger contexts are used, the resulting latent topics act as primitive concepts to characterize document topics, i.e. aspects of the domain knowledge related to the corpus. When shorter contexts are adopted in $M$, latent topics characterize primitive concepts needed to distinguish short phrases: they thus tend to capture paradigmatic word classes, for which syntactic substitutability holds.

In order to determine lexical information provided by the proposed distributional models, an empirical analysis of the latent semantic topics obtained by SVD over the different source spaces, i.e. topical, word-based and syntactic-based space, has been carried out to find the possible generalizations obtained in these cases. Different distributional models are acquired from the ukWaC \cite{Baroni et al. (2009)} corpus, a large scale Web document collection made by 2 billion tokens. All \textit{tws} occurring more than 200 times (i.e. more than 50,000 words) are represented and different approaches discussed in Section 3.2 are applied to define the word-by-context matrix $M$. They are described as follows:

- **Topical Space**: the entire corpus has been split so that each column of $M$ represents a document. Each matrix item contains the \textit{tf-idf} score of a target word ($tw$) with respect to each corresponding document. It means that contexts, i.e. the matrix columns, are documents in the ukWaC corpus and two words are related if they tend to co-occur in the same documents. Two similar words in this

\footnote{Note that SVD emphasizes directions with maximal covariance for $M$, i.e. term clusters for which it is maximal the difference between contexts, i.e. short syntagmatic patterns.}
3.3. Embedding Lexical Semantics in lower dimensional spaces

space share a sintagmatic relation. For instance, the words *cut* and *knife* are syntagmatically similar since they typically co-occur within the same context. In this case it is said a relation *in praesentia*.

- **Word-based Space**: a co-occurrence word-based space provides a more specific notion of similarity and contexts are not documents anymore, but instead other words in the corpus. It means that two words are related if they co-occur with other words in the ukWaC corpus in a window of size $n$. Individual co-occurrence scores are weighted according to the Pointwise Mutual Information (PMI). The windows size could be tight, i.e. $n = 3$ or wider to encompass a range of an entire sentence, e.g. $n = 5, 7$ or $10$. This kind of matrix highlights paradigmatic similarity between words: similar words in this space may be substituted for one another in a particular context. For example, in the context expressed by the sentence *I read the book*, the word *book* can be replaced by *magazine* with no violation of the semantic well-formedness of the sentence, and therefore the two words can be said to be paradigmatically similar. This particular context dimension is selected to have a more precise representation and better capturing paradigmatic relations between words. Paradigmatic relations hold between linguistic entities that occur in the same context but not at the same time, that's why such that substitutional relation between words its a relation *in absentia*. The larger the window size, the more the relation between words in these kind of spaces refer to sintagmatic aspects.

The SVD reduction, to a dimension of 250, is finally applied to two spaces, i.e. topical and word-based, represented respectively by two different matrix $M_t$ and $M_w$. This empirical evaluation consists in the projection of a noun and a verb, i.e. *car* and
Chapter 3. Distributional Approaches to Language Semantics

*buy.v*, into the reduced space and the selection of the most 5 similar words according to the cosine similarity measure, expressed in Equation \[3.1\]. By projecting the noun *car.n* in the topical space, the 5 most similar words are *chevrolet.n, subcompact.n, plastic-body.n, four-door.n* and *mazda.n*, in the word-based space are *vehicle.n, motorcycle.n, motorbike.n, bike.n* and *scooter.n*. For the verb *buy.v* the most similar words are *jump.v, bet.n, squeeze.v, big.n* and *hefty.j*, according to the topical space, *sell.v, purchase.v, supply.v, stock.v* and *cheap.j* according to the word-based space. The example seems to show that paradigmatic generalizations are captured in the word-based space, whereas *car.n* and *buy.v* are correctly generalized in synonyms (such as *vehicle.n/motorcycle.n* and *sell.v/purchase.v*). The document space instead seems to suggest topical similarity (such as *chevrolet.n/four-doors.n* vs. *car.n* or *bet.n/big.n* vs. *buy.v*) that is much more vague. Notice how a generic verb like *buy.v* that is used in many different contexts (i.e. documents), could be brought near other generic verbs like *squeeze.v* or *jump.v*. Therefore, since for some words like *big.n* or *bet.n* or *hefty.n* the topical relation is clear, e.g. “to make a hefty purchase”, for the letters it is more complicated to find a relationship, except to say that for some reason both words appears in the same document of the corpus.

It is not trivial to provide a judgment on the best space for every language learning task, as different semantic relations between lexemes, i.e. topical or paradigmatic relations, may contribute differently depending on the target problem.
Distributional Compositional Semantic Models as a model for meaning

While compositional approaches to language understanding have been largely adopted, semantic tasks are still challenging for application in the Natural Language Processing area. Studies on NL semantics based on traditional logic-based approaches rely on Frege’s principle, allow an algebra on the discrete propositional symbols to represent the meaning of arbitrarily complex expressions. Computational models of semantics based on symbolic logic representations can account naturally for the meaning of sentences, through the notion of compositionality for which the meaning of complex expressions can be determined by using the meanings of their constituent and the rules to combine them. Despite the fact that they are formally well defined, logic-based approaches have limitations in the treatment of ambiguity, vagueness and cognitive aspects intrinsically connected to natural language. For instance, the sentence Meanwhile, the bank closed could either refer to the closing time of an office, or to the “cease to operate” sense of a bankrupt. Logic-based approaches present strict limitation towards these tasks and bring often inadequate tool to model and overcome the uncertainty of phenomena like select the proper interpretation of a specific verb-object pair.

In Chapter[3] has been described how the meaning representation of words is achieved
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

by the formulation of distributional approaches that seem to model lexicon in isolation or in context. It has also been discussed that the (distributional) meaning of a word is a summary of the contexts in which the word can occur. In such that computational model of word meaning, distributional patterns of words collected over large text data are used to represent semantic similarity between words in terms of spatial proximity. Distributional models for lexical meaning are typically directed at representing words singularly. However it is possible to ask whether these approaches can be extended to account for the meaning of phrases and sentences as well.

While much research has been directed at the most effective ways of constructing representations for individual words, there has been far less consensus regarding the representation of larger constructions such as phrases and sentences.

Despite their success, single word vector models are severely limited to achieve the above goal, since they do not capture compositionality. Compositional semantics allows to govern the recursive interpretation of sentences or phrases, while vector space models and, mostly, semantic space models, such as LSA, represent lexical information in metric spaces where individual words are represented according to the distributional analysis of their co-occurrences over a large corpus.

Empirical compositional distributional methods account for the meaning of syntactic structures by combining words according to algebraic operators (e.g. the tensor product) acting over the corresponding lexical constituents. Anyway, semantic similarity seems to be more complex than simply a relation between isolated words. Methods for constructing representations for phrases or sentences through vector composition has recently received a wide attention in literature (e.g. [Mitchell and Lapata(2008)])

In this Chapter, a novel approach for semantic composition based on space projec-
tion techniques over the basic geometric lexical representations is proposed. Thus first of all, in Section 4.1 an overview over the most important Distributional Compositional Semantic (DCS) models is presented.

In line with Frege’s context principle, the meaning of a phrase is modeled in terms of the subset of properties shared by the co-occurring words. Since we are not committed to the limiting view that word and phrases vectors must live in the same contextual space, in the geometric perspective here pursued, syntactic bi-grams are projected in the so-called Support Subspace, aimed at emphasizing the semantic features shared by the compound words and better capturing phrase-specific aspects of the involved lexical meanings. In Section 4.2 a DCS model based on lexical vector projection over such that Support Subspace is defined, while metrics to compute compositionality scores between lexical phrases are discussed in Section 4.3. Finally in Sections 4.4 and 4.5 the generalization capability of the model evaluated first in an Information Retrieval scenario and than in a Semantic Textual Similarity task is investigated.

4.1 Semantic Models in Distributional spaces

Vector-based models typically represent isolated words and ignore grammatical structure [Turney and Pantel(2010)]. They have thus a limited capability to model compositional operations over phrases and sentences.

In order to overcome these limitations a so-called Distributional Compositional Semantics (DCS) model is needed. A compositional model based on distributional analysis should provide semantic information consistent with the meaning assignment typical of human subjects. For example, it should support synonymy and similarity judgments on phrases, rather than only on single words. The objective should be a
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

measure of similarity between quasi-synonymic complex expressions, such as “... buy a car ...” vs. “... purchase an automobile ...”. Another typical benefit should be a computational model for entailment, so that the representation for “... buying something ...” should be implied by the expression “... buying a car ...” but not by “... buying time ...”. Distributional compositional semantics (DCS) needs thus a method to define: (1) a way to represent lexical vectors $u$ and $v$, for words $u, v$ dependent on the phrase $(r, u, v)$ (where $r$ is a syntactic relation, such as verb-object), and (2) a metric for comparing different phrases according to the selected representations $u, v$.

Existing models are still controversial and provide general algebraic operators over lexical vectors and sophisticated composition methods, e.g. based on tensor products or quantum logic [Clark and Pulman(2007), Rudolph and Giesbrecht(2010a), Smolensky(1990a), Widdows(2008)]. A general framework for vector composition has been proposed by Mitchell and Lapata in [Mitchell and Lapata(2008)] and further developed in [Guevara(2010)] and in [Baroni and Zamparelli(2010)]. In the next section an overview of these methods is presented, while in Section 4.2 and 4.3 our approach based on projecting lexical vector into Support Subspace is defined and investigated.

4.1.1 Distributional Compositional Semantic Models

Distributional methods have been recently extended to better account compositionality, in the so called distributional compositional semantics (DCS) approaches. In [Smolensky(1990b)] compositionality of two vector $u$ and $v$ is accounted by the tensor product $u \otimes v$, i.e. a matrix whose components are all the possible products $u_i v_j$ of the components of vectors $u$ and $v$. The problem here is that a tensor product representation of a lexical vector composition has a higher dimensionality of the original vectors, thus
4.1. Semantic Models in Distributional spaces

Further techniques proposed the binding of two vectors as a vector which has the same dimensionality as its components [Plate(1991)]. Another common method to account for the meaning representation of a sentence in which individual words are represented by vector is the common one proposed in [Foltz et al. (1998)]: lexical vectors are summed, keeping the resulting vector with the same dimension of the latter. Since syntactic structure is here ignored, i.e. the model is order independent, a variation on the vector addition has been proposed in [Kintsch(2001)] to model how the meaning of a predicate varies depending on the arguments it operates upon. Kintsch add not only the vectors representing the predicate and its argument but also the neighbors associated with both of them. Mitchell and Lapata follow Foltz and assume that the contribution of syntactic structure can be ignored, while the meaning of a phrase is simply the commutative sum of the meanings of its constituent words. In [Mitchell and Lapata(2008)] semantic composition has been formulated as a function of two vectors $u$ and $v$, their syntactic relation $R$ and the factor $K$ defined as any additional knowledge or information which is needed to construct the semantics of their composition. Thus it follows that

$$p = f(u, v, R, K)$$  \hspace{1cm} (4.1)

where $p$ denotes the resulting composition vector. By holding $R$ fixed and by ignoring $K$, specific models from this general framework could be derived, reducing the class of models to

$$p = f(u, v)$$  \hspace{1cm} (4.2)

Assuming that $p$ lies in the same space as $u$ and $v$ [Mitchell and Lapata(2008)] defines two general classes of composition models, linear additive models:

$$p = Au + Bv$$  \hspace{1cm} (4.3)
where \( A \) and \( B \) are weight matrices, and multiplicative models:

\[
p = C u v
\]  

(4.4)

where \( C \) is a weight tensor projecting the \( uv \) tensor product onto the space of \( p \). This perspective clearly leads to a variety of efficient yet shallow models of compositional semantics compared in \[Mitchell and Lapata(2008)\]. Two simplified models are derived from these general forms, thus the additive model became:

\[
p^+ = \alpha u + \beta v
\]  

(4.5)

that is the addition of the two lexical vectors to which apply the weights \( \alpha \) and \( \beta \)

\[
p^i = u \odot v
\]  

(4.6)

that is the pointwise multiplication and the symbol \( \odot \) represents multiplication of the corresponding components, i.e. \( p_i = u_i \cdot v_i \). Since the cosine similarity function is insensitive to the vectors magnitude, in \[Mitchell and Lapata(2008)\] a more complex asymmetric type of function called dilation is introduced. It consists in multiplying vectors \( v \) by the quadratic factor \( u \cdot u \) and \( v \) by a stretching factor \( \lambda \) as follows:

\[
p^d = (u \cdot u)v + (\lambda - 1)(u \cdot v)u
\]  

(4.7)

Notice that either \( u \) can be used to dilate \( v \), or \( v \) can be used to dilate \( u \). The best dilation factor \( \lambda \) for the dilation models is studied and tuned in \[Mitchell and Lapata(2008)\], where a framework for vector composition is presented and a range of potential composition functions is explored.

According to equation (4.5) the addition of the two vectors in Table 4.1 representing \( \text{buy} \) and \( \text{car} \) would be \( \text{buy} + \text{car} = [17 \ 17 \ 3 \ 19 \ 6] \). This model assumes that
4.1. Semantic Models in Distributional spaces

<table>
<thead>
<tr>
<th></th>
<th>cheap</th>
<th>rent</th>
<th>consume</th>
<th>lease</th>
<th>save</th>
</tr>
</thead>
<tbody>
<tr>
<td>buy</td>
<td>12</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>car</td>
<td>5</td>
<td>11</td>
<td>1</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>time</td>
<td>1</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 4.1: A hypothetical semantic space for buy, car and time

Table 4.2: Resulting compositional vectors for buy-car and buy-time by adopting Mitchell&Lapata additive and multiplicative models

composition is a symmetric function of the constituents; in other words, the order of constituents essentially makes no difference. While this might be reasonable for certain structures, a list perhaps, a model of composition based on syntactic structure requires some way of differentiating the contributions of each constituent. Moreover notice that the syntactic structure is ignored in the composition. These features are still valid for the model in Equation 4.6 for which \( \text{buy} \odot \text{car} = [60 \ 66 \ 2 \ 90 \ 8] \). Finally notice that the resulting composition vectors for buy car and buy time are still comparable and similar in the original space: the corresponding compound vector for both the multiplicative and the additive models are shown in Table 4.2. Anyhow, dilation and point-wise multiplication seem to best correspond with the intended effects of syntactic interaction, as experiments in Mitchell and Lapata(2008) and in Mitchell and Lapata(2009) demonstrate. The multiplicative approach strikes best results also on the task of paraphrasing noun-verb combinations with ambiguous verbs.

In Erk and Pado(2008), the concept of a structured vector space is introduced,
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

where each word is associated to a set of vectors corresponding to different syntactic dependencies. Noun component of a composition between verb and noun is here given by an average of verbs that the noun is typically object of. Every word is thus expressed by a tensor, and tensor operations are imposed. Again, the multiplicative model seems to perform best their experiments. In [Thater et al. (2010)] a similar geometric approach over a space of second-order distributional vectors is presented. Vectors represent the words typically co-occurring with the contexts in which the target word appears. The primary concern of this study is to model the effect of the context on word meaning. A parallel strand of research also seeks to represent the meaning of larger compositional structures using matrix and tensor algebra, like in [Smolensky (1990b)] and in [Rudolph and Giesbrecht (2010b)].

In [Guevara (2010)] a regressor is trained for adjective-noun (AN) compositionality. Co-occurrence informations of ANs are collected from windows around the contexts in which they occur in such a way that composite items are treated as single tokens in a Word Space. Pairs of adjective-noun vector concatenations are used as input in training data, whilst corpus-derived AN vectors as output. Additive composition model in Equation 4.4 is applied and $A$ and $B$ are estimated by using Partial Least Squares Regression (PLRS). According to Guevara, observed ANs are nearer, in the space of observed and predicted test set ANs, to the ANs generated by his model than to those from the alternative approaches.

Similarly to Guevara, in [Baroni and Zamparelli (2010)] and [Coecke et al. (2010)] composition is characterized from formal semantics in terms of a function application, where the distributional representation of one element in a composition (the functor) is not a vector but a function. Given that linear functions can be expressed by matrices
and their application by matrix-by-vector multiplication, in this lexical function, i.e. \textit{lexfunc}, model, the functor is represented by a matrix $B$ to be multiplied with the argument vector $v$. More precisely, in \cite{Baroni2010}, ANs composition is derived from Equation 4.4 with $A$, i.e. the matrix multiplying the adjective vector, set to 0, thus:

$$p = Bv$$

where the resulting AN vector is $p$, while $v$ is a noun vector and $B$, estimated by using PLRS, the weight matrix representing the adjective. A hybrid Logico-Distributonal Model is presented in \cite{Grefenstette2011}: given a sentence with a verb and its $i$ arguments, the resulting representations is a rank-$i$ tensor $S$, thus

$$S = V \odot (a_1 \otimes a_2 \otimes a_3 \otimes \cdots \otimes a_i)$$

where $V$ is a a rank $i$ tensor representing the verb and the second factor is the Kronecker product of the vectors representing the verb arguments. Finally an approach based on vector permutation and Random Indexing technique, i.e. \cite{Sahlgren2006}, is presented in \cite{Basile2011}.

The main differences among these studies lies in (1) the lexical vector representation selected (e.g. some authors do not even commit to any representation, but generically refer to any lexical vector, as in \cite{Grefenstette2011}) as well as in (2) the adopted compositional algebra, i.e. the system of operators defined over such vectors. In most work, operators do not depend on the involved lexical items, but a general purpose algebra is adopted. Since compositional structures are highly lexicalized, and the same syntactic relation gives rise to very different operators with respect to the different involved words, a proposal that makes the compositionality operators dependent on individual lexical vectors is hereafter discussed.
4.2 Space Projections as Distributional Models for Semantics

Let’s start to discuss the above compositional model over the example used in section 4.1.1 where we want to model the semantic analogies and differences between ”... buy a car ...” and ”... buy time ...”. The involved lexicals are buy, car and time, while their corresponding vector representation will be denoted by \( w_{\text{buy}} \), \( w_{\text{car}} \) and \( w_{\text{time}} \). The major result of most studies on DCS is the definition of the function \( \circ \) that associates to \( w_{\text{buy}} \) and \( w_{\text{car}} \) a new vector \( w_{\text{buy}} \circ w_{\text{car}} = w_{\text{buy,car}} \).

We consider this approach misleading since vector components in the word space are tied to the syntactic nature of the composed words and the new vector \( w_{\text{buy,car}} \) should not have the same type of the original vectors. Mathematical operations between the two input vectors (e.g. point wise multiplication \( \odot \) as in Eq. 4.6) produce a vector for a structure (i.e. a new type) that possess the same topological nature of the original vectors. As these latter are dedicated to express arguments, i.e. a verb and its object in the initial space, the syntactic information (e.g. the relation and the involved POS) carried independently by them is neglected in the result. For example, the structure ”... buy a car ...” combines syntactic roles that are different and the antisymmetric relationship between the head verb and the modifier noun is relevant. The vectorial composition between \( w_{\text{buy}} \) and \( w_{\text{car}} \), as proposed in Eq. 4.6 [Mitchell and Lapata(2010a)], even if mathematically correct, results in a vector \( w_{\text{buy,car}} \) that does not exploit this syntactic constraint and may fail to express the underlying specific semantics.

Notice also that the components of \( w_{\text{buy}} \) and \( w_{\text{car}} \) express all their contexts, i.e. interpretations, and thus senses, of buy and car in the corpus. Some mathematical
operations, e.g. the tensor product between these vectors, are thus open to misleading contributions, brought by not-null feature scores of \( buy_i \) vs. \( car_j \) \((i \neq j)\) that may correspond to senses of \( buy \) and \( car \) that are not related to the specific phrase “\( buy \ a \ car \)”.

On the contrary, in a composition, such as the verb-object pair \( (buy, car) \), the word \( car \) influences the interpretation of the verb \( buy \) and vice versa. The model here proposed is based on the assumption that this influence can be expressed via the operation of projection into a subspace, i.e. a subset of original features \( f_i \). A projection is a mapping (a selection function) over the set of all features. A subspace local to the \( (buy, car) \) phrase can be found such that only the features specific to its meaning are selected. It seems a necessary condition that any correct interpretation of the phrase has to be retrieved and represented on the subspace of the properties shared by the proper sense of individual co-occurring words. In order to separate these word senses and neglect irrelevant ones, a projection function \( \Pi \) must identify these common semantic features. The resulting subspace has to preserve the compositional semantics of the phrase and it is called support subspace of the underlying word pair.

Consider the bigram composed by the words \( buy \) and \( car \) and their vectorial representation in a co-occurrence \( N \)–dimensional Word Space. Notice that different vectors are usually derived for different POS tags, so that the verbal and nominal use of \( buy \) are expressed by two different vectors, i.e. \( buy.V \) and \( buy.N \). Every component of the vectors in a word space expresses the co-occurrence strength (in terms of frequency or \( pmi \)) of \( buy.V \) with respect to one feature, i.e. a co-occurring POS tagged word such as \( cost.N, pay.V \) or \( cheaply.ADV \). The support space selects the most important features for both words, e.g. \( buy.V \) and \( car.N \). Notice that this captures the intersective nature
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

<table>
<thead>
<tr>
<th>Buy-Car</th>
<th>Buy-Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>cheap::Adj</td>
<td>consume::V</td>
</tr>
<tr>
<td>insurance::N</td>
<td>enough::Adj</td>
</tr>
<tr>
<td>rent::V</td>
<td>waste::V</td>
</tr>
<tr>
<td>lease::V</td>
<td>save::In</td>
</tr>
<tr>
<td>dealer::N</td>
<td>permit::N</td>
</tr>
<tr>
<td>motorcycle::N</td>
<td>stressful::Adj</td>
</tr>
<tr>
<td>hire::V</td>
<td>spare::Adj</td>
</tr>
<tr>
<td>auto::N</td>
<td>save::V</td>
</tr>
<tr>
<td>california::Adj</td>
<td>warner::N</td>
</tr>
<tr>
<td>tesco::N</td>
<td>expensive::Adj</td>
</tr>
</tbody>
</table>

Table 4.3: Features corresponding to dimensions in the k=10 dimensional support space of bigrams buy car and buy time

of the scalar product to which contributions come from feature with non zero scores in both vectors. Moreover, the feature score is a weight, i.e. a function of the relevance of a feature for the represented word.

As an example, let us consider the phrase “... buy time ...”. Although the verb buy is the same of ”... buy a car ...”, its meaning (i.e. to do something in order to achieve more time) is clearly different. Since vector $w_{\text{buy}}$ expresses at least both possible meanings of the verb buy, different subspaces must be evoked in a distributional model for buy car vs. buy time.

Ranking features from the most important to the least important for a given phrase (i.e. pair $u$ and $v$) can be done by sorting in decreasing order the components $p_i = u_i \cdot v_i$, i.e. the addends in the scalar product. This leads to the following useful:

**Definition** ($k$-dimensional support space). A $k$—dimensional support subspace for a word pair $(u, v)$ (with $k \ll N$) is the subspace spanned by the subset of $n \leq k$
4.2. Space Projections as Distributional Models for Semantics

indexes $I^k(u, v) = \{i_1, ..., i_n\}$ for which $\sum_{t=1}^{n} u_{i_t} \cdot v_{i_t}$ is maximal.

We will hereafter denote the set of indexes characterizing the support subspace of order $k$ as $I^k(u, v)$.

Table 4.3 reports the $k = 10$ features with the highest contributions of the point wise product of the pairs $(buy, car)$ and $(buy, time)$. It is clear that the two pairs give rise to different support subspaces: the main components related with buy car refer mostly to the automobile commerce area unlike the ones related with buy time mostly referring to the time wasting or saving.

Similarity judgments about a pair can be thus computed within its support subspace. Given two pairs the similarity between syntactic equivalent words (e.g. nouns with nouns, verbs with verbs) is measured in the support subspace derived by applying a specific projection function. In the above example, the meaning representation of buy and car is obtained by projecting both vectors in their own subspace in order to capture the (possibly multiple) senses supported by the pair. Then, compositional similarity between buy car and the latter pairs (e.g. buy time) is estimated by (1) immersing $w_{buy}$ and $w_{time}$ in the selected ”...buy car...” support subspace and (2) estimating similarity between corresponding arguments of the pairs locally in that subspace. As exemplified in Table 4.3, two pairs give rise to two different support spaces, so that there are two ways of projecting the two pairs. In order to provide precise definitions for these notions, formal definitions will be hereafter provided through linear algebra operators.

**Space projections and compositionality.** Support spaces (of dimension $k$) are isomorphic to projections in the original space. A projection $\Pi^k(u, v)$ can be used and provides a computationally simple model for expressing the intrinsic meaning of any
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

underlying phrase \((u, v)\). Given the source vectors of a compound \((u, v)\), space projection depends on both the two involved lexical items and selects only their "common" features: these concurrently constraint the suitable lexical interpretation local to the phrase. The core dimensions of a compound could be identified by the set

\[
C(u, v) = \{ i \in \{1, \ldots, n\} | u_i \cdot v_i \geq 0 \}
\]

where \(l = |C(u, v)|\) components sharing the same sign have been selected. A \(k\)-sized Support Subspace, with \(k \leq l\), is a \(k\)-dimensional space where the selected indexes are a subset of \(C\), thus

\[
I^k(u, v) = \{ i_1, \ldots, i_k | u_t \in C(u, v) \text { such that } \forall t \in [1, \ldots, l] \quad u_{i_t} \cdot v_{i_t} \geq u_{i_{t+1}} \cdot v_{i_{t+1}} \}
\]

Given a pair \((u, v)\), a unique matrix \(M^k_{uv} = (m^k_{uv})_{ij}\) is defined for a given projection into the \(k\)-dimensional support space of any pair \((u, v)\) according to the following definition:

\[
M^k_{uv} = (m_{uv})_{ij} = \begin{cases} 
1 & \text{iff } i = j \in I^k(u, v) \\
0 & \text{otherwise.}
\end{cases}
\]

The vector \(\tilde{u}\) projected in the support subspace can be thus estimated through the following matrix operation:

\[
\tilde{u} = \Pi^k(u, v) \quad \tilde{u} = M^k_{uv} u
\]

A special case of the projection matrix is given when no \(k\) limitation is imposed to the dimension and all the positive addends in the scalar product are taken. This maximal support subspace, denoted by removing the superscript \(k\), i.e. as \(M_{uv} = (m_{uv})_{ij}\), is defined as follows:

\[
(m_{uv})_{ij} = \begin{cases} 
0 & \text{iff } i \neq j \text{ or } u_i \cdot v_i \leq 0, \\
1 & \text{otherwise.}
\end{cases}
\]
From Eq. [4.13] it follows that the support subspace components are those with positive product.

**Definition.** *Left and Right Projections.* Two phrases \((u, v)\) and \((u', v')\) give rise to two different projections, defined as follows

\[
\begin{align*}
\text{(Left projection)} & \quad \Pi^k_1 = \Pi^k(u, v) \\
\text{(Right projection)} & \quad \Pi^k_2 = \Pi^k(u', v')
\end{align*}
\] (4.14)

We will denote the two projection matrices as \(M^k_1\) and \(M^k_2\), correspondingly. In order to achieve a unique symmetric projection \(\Pi^k_{12}\), it is possible to define the corresponding combined matrix \(M^k_{12}\) as follows:

\[
M^k_{12} = (M^k_1 + M^k_2) - (M^k_1 M^k_2)
\] (4.15)

where the mutual components that satisfy Eq. [4.11] (or Eq [4.13]) are employed as \(M^k_{12}\) (or \(M_{12}\) respectively).

**Compositional Similarity Judgments.** The projection function that locates the support subspace of a word pair \((v, o)\), whose syntactic type is *verb-object*, i.e. \(VO\), will be hereafter denoted by \(\Pi_{VO}(v, o)\). Given two word pairs \(p_1 = (v, o)\) and \(p_2 = (v', o')\), we define here a compositional similarity function \(\Phi(p_1, p_2)\) as a model of the similarity between the underlying phrases. As the support subspace for the pair \(p_1\) is defined by the projection \(\Pi_1\), it is possible to immerse the latter pair \(p_2\) by applying Eq. [4.12] This results in the two vectors \(M_1v'\) and the \(M_1o'\). It follows that a compositional similarity judgment between two verbal phrase over the left support subspace can be expressed as:

\[
\Phi^{(o)}_{p_2}(p_1, p_2) = \Phi^{(o)}_1(p_1, p_2) = \frac{\langle M^k_1 v, M^k_2 v' \rangle}{\|M^k_1 v\| \|M^k_2 v'\|} \circ \frac{\langle M^k_1 o, M^k_2 o' \rangle}{\|M^k_1 o\| \|M^k_2 o'\|}
\] (4.16)
where first cosine similarity between syntactically correlated vectors in the selected support subspaces are computed and then a composition function \( \circ \), such as the sum or the product, is applied. Notice how the compositional function over the right support subspace evoked by the pair \( p_2 \) can be correspondingly denoted by \( \Phi_2^{(o)}(p_1, p_2) \). A symmetric composition function can thus be obtained as a combination of \( \Phi_1^{(o)}(p_1, p_2) \) and \( \Phi_2^{(o)}(p_1, p_2) \) as:

\[
\Phi_{12}^{(o)}(p_1, p_2) = \Phi_1^{(o)}(p_1, p_2) \circ \Phi_2^{(o)}(p_1, p_2)
\]

where the composition function \( \circ \) (again the sum or the product) between the similarities over the left and right support subspaces is applied. Notice how the left and right composition operators \( \circ \) may differ from the overall composition operator \( \circ \), as we will see in experiments. The above definitions in fact characterize several projection functions \( \Pi^k \), local composition function \( \Phi_1^{(o)} \) as well as global composition function \( \Phi_{12}^{(o)} \). It is thus possible to define variants of the models presented above according to four main parameters:

**Support Selection.** Two different projection functions \( \Pi \) have been defined in Eq. 4.11 and Eq. 4.13 respectively. The **MAXIMAL SUPPORT** \( \Pi \) denotes the support space defined in Eq. 4.13. The **k-DIMENSIONAL SUPPORT** defined in Eq. 4.11 is always denoted by the superscript \( k \) in \( \Pi^k \) instead.

**Symmetry of the similarity judgment.** A **SYMMETRIC** judgment (denoted by simple \( \Phi_{12} \)) involves Eq. 4.17 in which compositionality depends on both left and right support subspaces. In an **ASYMMETRIC** projection the support subspace belonging to a single (left \( \Phi_1 \), or right \( \Phi_2 \)) pair is chosen. In all the experiments we applied Eq. 4.16.
4.3. A simple CDS metric for lexical pairs composition

by only considering the left support subspace, i.e. $\Phi_1$.

**Symmetry of the support subspace.** A support subspace can be build as:

- an INDEPENDENT SPACE, where different, i.e. left and right, support subspaces are built, through different projection functions $M_1$ and $M_2$ independently

- a UNIFIED SPACE, where a common subspace is built according to Eq. 4.15, and denoted by the projection matrix $M_{12}$

**Composition function.** The composition function $\Phi^\circ$ in Eq. 4.16 and 4.17 can be the product or the sum as well. We will denote $\Phi^+_i$ or $\Phi_i$ as well as $\Phi^+$ and $\Phi^-$ to emphasize the use of sum or product in Eq. 4.16 and 4.17. The only case in which no combination is needed is when the unified support space (as in Eq. 4.15) is used, and thus no left or right $\Pi_i$ is applied, but just $\Pi_{12}$.

4.3 A simple CDS metric for lexical pairs composition

The aim of this first evaluation is to estimate if the proposed class of projection based methods for distributional compositional semantics is effective in capturing similarity judgments over phrases and syntactic structures. We tested our method over binary phrase structures represented by verb-object, noun-noun ad adjective-noun. Evaluation is carried out over the dataset proposed by [Mitchell and Lapata(2010a)], which is part of the GEMS 2011 Shared Evaluation. It consists of a list of 5,833 adjective-noun (AdjN), verb-object (VO) or noun-noun (NN) pairs, rated with scores ranging from 1 to 7. In Table 4.4 examples of pairs and scores are shown: notice how the similarity between the (VO) offer support and provide help is higher than the one between achieve end and close eye. The correlation of the similarity judgements output by a DCS model
Table 4.4: Example of Mitchell and Lapata dataset for the three syntactic relations verb-object (VO), adjective-noun (AdjN) and noun-noun (NN)

against the human judgements is computed using Spearman’s $\rho$, a non-parametric measure of statistical dependence between two variables proposed by Mitchell and Lapata(2008).

We employed two different word spaces derived from a corpus, i.e. ukWak [Baroni et al. (2009)], including about 2 billion tokens. Each space construction proceeds from an adjacency matrix $M$ on which Singular Values decomposition ([Deerwester et al. (1990)]) is then applied. Part-of-speech tagged words have been collected from the corpus to reduce data sparseness. Then all target words $tws$ occurring more than 200 times are selected, i.e. more that 50,000 candidate features. A first space, called sentence-based space, is derived by applying SVD to a $M=\text{term} \times \text{sentence}$ adjacency matrix. Each column of $M$ represents thus a sentence of the corpus, with about 1,500,000 sentences and $tf-idf$ scores for words $w$ in each row. The dimensions of the resulting SVD matrix in the sentence-based space is $N = 250$.

The second space employed is a word space built from the ukWak co-occurrences where left contexts are treated differently from the right ones for each target word $tw$. 

<table>
<thead>
<tr>
<th>Type</th>
<th>First Pair</th>
<th>Second Pair</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>VO</td>
<td>support offer</td>
<td>provide help</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>use knowledge</td>
<td>exercise influence</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>achieve end</td>
<td>close eye</td>
<td>1</td>
</tr>
<tr>
<td>AdjN</td>
<td>old person</td>
<td>right hand</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>vast amount</td>
<td>large quantity</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>economic problem</td>
<td>practical difficulty</td>
<td>3</td>
</tr>
<tr>
<td>NN</td>
<td>tax charge</td>
<td>interest rate</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>tax credit</td>
<td>wage increase</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>bedroom window</td>
<td>education officer</td>
<td>1</td>
</tr>
</tbody>
</table>
4.3. A simple CDS metric for lexical pairs composition

Each column in $M$ represents here a word $w$ in the corpus and in rows we found the $pmi$ values for the individual features $f_i$, as captured in a window of size $±3$ around $w$. The most frequent 20,000 left and right features $f_i$ are selected, so that $M$ expresses 40,000 contexts. SVD is here applied to limit dimensionality to $N = 100$.

Comparative analysis with results previously published in [Mitchell and Lapata(2010a)] has been carried out. We also recomputed the performance measures of operators in [Mitchell and Lapata(2010a)] (e.g. M&L multiplicative or additive models of Eq. 4.5 and 4.6) over all the word spaces specifically employed in the rest of our experiments.

Table 4.5 reports M&L performances in first three rows. In the last row of the Table the max and the average interannotator agreement scores for the three categories derived through a leave one-out resampling method, are shown. For each category with a set of subjects responses of size $m$, a set of $m − 1$ (i.e., the response data of all but one subject) and a set of size one (i.e., the response data of the single remaining subject) are derived. The average rating of the set of $m − 1$ subjects is first calculated and then Spearman’s $\rho$ correlation coefficient with respect to the singleton set is computed. Repeating this process $m$ times results in an average and maximum score among the results (as reported in row 6). The distributional compositional models discussed in this paper are shown in rows 4 and 5, where different configurations are used according to the models described in Section 4.2. For example, the system denoted in Table 4.5 as $\Phi_{12}^{(+)}$, $\Phi_{i}^{(+)}$, $\Pi_{i}^{k}$ ($k=40$), corresponds to an additive symmetric composition function $\Phi_{12}^{(+)}$ (as for Eq. 4.17) based on left and right additive compositions $\Phi_{i}^{(+)}$ ($i = 1, 2$ as in Eq. 4.16), derived through a projection $\Pi_{i}^{k}$ in the support space limited to the first $k = 40$ components for each pair (as for Eq. 4.14).
### Table 4.5: Spearman’s $\rho$ correlation coefficients across Mitchell and Lapata models and the projection-based models proposed in Section 4.2. Topical Space and Word space refer to the source spaces. is used as input to the LSA decomposition model.

<table>
<thead>
<tr>
<th>Model</th>
<th>AdjN</th>
<th>NN</th>
<th>VO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mitchell &amp; Lapata,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Mitchell and Lapata(2010a)]</td>
<td>.36</td>
<td>.39</td>
<td>.30</td>
</tr>
<tr>
<td>Multiplicative</td>
<td>.46</td>
<td>.49</td>
<td>.37</td>
</tr>
<tr>
<td>Dilation</td>
<td>.44</td>
<td>.41</td>
<td>.38</td>
</tr>
<tr>
<td>Mitchell &amp; Lapata</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Topical SVD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive</td>
<td>.53</td>
<td>.67</td>
<td>.63</td>
</tr>
<tr>
<td>Multiplicative</td>
<td>.29</td>
<td>.35</td>
<td>.40</td>
</tr>
<tr>
<td>Dilation</td>
<td>.44</td>
<td>.49</td>
<td>.50</td>
</tr>
<tr>
<td>Mitchell &amp; Lapata</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Word Space SVD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive</td>
<td>.69</td>
<td>.70</td>
<td>.64</td>
</tr>
<tr>
<td>Multiplicative</td>
<td>.38</td>
<td>.43</td>
<td>.42</td>
</tr>
<tr>
<td>Dilation</td>
<td>.60</td>
<td>.57</td>
<td>.61</td>
</tr>
<tr>
<td>Sentence Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Phi^{(+)}_i$, $\Pi^{k}_i$ ($k=20$)</td>
<td>.58</td>
<td>.62</td>
<td>.64</td>
</tr>
<tr>
<td>$\Phi^{(+)}_{12}$, $\Phi^{(+)}_i$, $\Pi^{k}_i$ ($k=40$)</td>
<td>.55</td>
<td>.71</td>
<td>.65</td>
</tr>
<tr>
<td>$\Phi^{(+)}_{12}$, $\Phi^{(+)}_i$, $\Pi^{k}_i$ ($k=10$)</td>
<td>.49</td>
<td>.65</td>
<td>.66</td>
</tr>
<tr>
<td>Word Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Phi^{(\cdot)}_i$, $\Pi^{k}_i$ ($k=30$)</td>
<td>.70</td>
<td>.71</td>
<td>.63</td>
</tr>
<tr>
<td>$\Phi^{(\cdot)}_{12}$, $\Phi^{(\cdot)}_i$, $\Pi^{k}_i$ ($k=40$)</td>
<td>.68</td>
<td>.68</td>
<td>.64</td>
</tr>
<tr>
<td>$\Phi^{(\cdot)}_{12}$, $\Phi^{(\cdot)}_i$, $\Pi_i$</td>
<td>.70</td>
<td>.65</td>
<td>.61</td>
</tr>
<tr>
<td>Agreement among</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>.88</td>
<td>.92</td>
<td>.88</td>
</tr>
<tr>
<td>Avg</td>
<td>.72</td>
<td>.72</td>
<td>.71</td>
</tr>
</tbody>
</table>
4.3. A simple CDS metric for lexical pairs composition

First, Mitchell and Lapata operators applied onto our sentence and word space models over perform results previously presented in [Mitchell and Lapata(2010a)] (i.e. row 2 and 3 vs. row 1). This is mainly due to the benefits of the SVD modeling adopted here. The use of pmi scores in word spaces or tf-idf values in sentence spaces, then subject to the SVD factorization, is beneficial for the multiplicative and additive models proposed in the past.

The best performances are achieved by the projection based operators proposed in this paper. The word space version (denoted by $\Phi^{(\pm)}$, $\Pi_{12}^k$ ($k=30$)) gets the best performance over two out of three syntactic patterns (i.e. AdjN and NN) and is close to the best figures for VO. Notice how parameters of the projection operations influence the performance, so that different settings provide quite different results. This is in agreement with the expected property for which different syntactic compositions require different vector operations.

If compared to the sentence space, a word space, based on a small window size, seems better capture the lexical meaning useful for modeling the syntactic composition of a pair. The subset of features, as derived through SVD, in a resulting support space is very effective as it is in good agreement with human judgements ($\rho=0.71$) A sentence space leads in general to a more topically-oriented lexical representations and this seems slightly less effective. In synthesis it seems that specific support subspaces are needed: a unified additive model based on a Word Space is better for adjective-noun and compound nouns while the additive symmetric model based on a sentence space is much better for verb-object pairs.

A general property is that the results of our models are close to the average agreement among human subjects, this latter representing a sort of upper bound for the
underlying task. It seems that latent topics (as extracted through SVD from sentence and word spaces) as well the projections operators defined by support subspaces provide a suitable comprehensive paradigm for compositionality. They seem to capture compositional similarity judgements that are significantly close to human ones.

4.4 Generalization capability of CDS Model, and application in an IR Scenario for ranking and clustering

In this second evaluation, the generalization capability of the employed operators will be investigated. A verb (e.g. perform) can be more or less semantically close to another verb (e.g. other verbs like solve, or produce) depending on the context in which it appears. The verb-object (VO) composition specifies the verb’s meaning by expressing one of its selectional preferences, i.e. its object. In this scenario, we expect that a pair such as perform task will be more similar to solve issue, as they both reflect an abstract cognitive action, with respect to a pair like produce car, i.e. a concrete production. This kind of generalization capability is crucial to effectively use this class of operators in a QA scenario by enabling to rank results according to the complex representations of the question. Moreover, both English and Italian languages can be considered to demonstrate the impact in a cross language setting. Figure 4.6 shows a manually developed dataset. It consists of 24 VO word pairs in English and Italian, divided into 3 different semantic classes: Cognitive, Ingest Liquid and Fabricate.

This evaluation aims to measure how the proposed compositional operators group together semantically related word pairs, i.e. those belonging to the same class, and separate the unrelated pairs. Figure 4.11 shows the application of two models, the Additive (eq. 4.5) and Support Subspace (Eq. 4.17) ones that achieve the best results in
4.4. Generalization capability of CDS Model, and application in an IR Scenario for ranking and clustering

<table>
<thead>
<tr>
<th>Semantic Class</th>
<th>English</th>
<th>Italian</th>
</tr>
</thead>
</table>
| Cognitive          | perform task  
solve issue  
handle problem  
use method  
suggest idea  
determine solution  
spread knowledge  
start argument     | svolgere compito  
risolvere questione  
gestire problema  
applicare metodo  
suggerire idea  
trovare soluzione  
divulgare conoscenza  
iniziare ragionamento |
| Ingest Liquid      | drink water  
ингestion syrup  
pour beer  
swallow saliva  
assume alcohol  
taste wine  
sip liquor  
take coffee  | bere acqua  
ingerire sciroppo  
versare birra  
inghiottire saliva  
assumere alcool  
assaggiare vino  
assaporare liquore  
prendere caff |
| Fabricate          | produce car  
complete construction  
fabricate toy  
built tower  
assemble device  
construct building  
manufacture product  
create artwork  | produrre auto  
completare costruzione  
fabbricare giocattolo  
edificare torre  
assemblare dispositivo  
costruire edificio  
realizzare prodotto  
creare opera    |

Table 4.6: Cross-linguistic dataset
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

the previous experiment. The two languages are reported in different rows. Similarity distribution between the geometric representation of the verb pairs, with no composition, has been investigated as a baseline. For each language, the similarity distribution among the possible 552 verb pairs is estimated and two distributions of the infra and intra-class pairs are independently plotted. In order to summarize them, a Normal Distribution $N(\mu, \sigma^2)$ of mean $\mu$ and variance $\sigma^2$ are employed. Each point represents the percentage $p(x)$ of pairs in a group that have a given similarity value equal to $x$. In a given class, the VO-VO pairs of a DCS operator are expected to increase this probability with respect to the baseline pairs V-V of the same set. Viceversa, for pairs belonging to different classes, i.e. intra-class pairs. The distributions for the baseline control set (i.e. Verbs Only, V-V) are always depicted by dotted lines, while DCS operators are expressed in continuous line.

Notice that the overlap between the curves of the infra and intra-class pairs corresponds to the amount of ambiguity in deciding if a pair is in the same class. It is the error probability, i.e. the percentage of cases of one group that by chance appears to have more probability in the other group. Although the actions described by different classes are very different, e.g. Ingest Liquid vs. Fabricate, most verbs are ambiguous: contextual information is expected to enable the correct decision. For example, although the class Ingest Liquid is clearly separated with respect to the others, a verb like assume could well be classified in the Cognitive class, as in assume a position.

The outcome of the experiment is that DCS operators are always able to increase the gap in the average similarity of the infra vs. intra-class pairs. It seems that the geometrical representation of the verb is consistently changed as most similarity
4.4. Generalization capability of CDS Model, and application in an IR Scenario for ranking and clustering

Figure 4.1: Cross-linguistic Gaussian distribution of infra (red) and inter (green) clusters of the proposed operators (continuous line) with respect to verbs only operator (dashed line)
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

<table>
<thead>
<tr>
<th>Model</th>
<th>English Probability of Error</th>
<th>English Ambiguity Decrease</th>
<th>Italian Probability of Error</th>
<th>Italian Ambiguity Decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>VerbOnly</td>
<td>.401</td>
<td>-</td>
<td>.222</td>
<td>-</td>
</tr>
<tr>
<td>Additive</td>
<td>.030</td>
<td>92.3%</td>
<td>.101</td>
<td>54.2%</td>
</tr>
<tr>
<td>SupportSubspace</td>
<td>.036</td>
<td>91.0%</td>
<td>.082</td>
<td>62.9%</td>
</tr>
</tbody>
</table>

Table 4.7: Ambiguity reduction analysis

distributions suggest. The compositional operators seem able to decrease the overlap between different distributions, i.e. reduce the ambiguity.

Figure 4.1 (a) and (c) report the distribution of the ML additive operator, that achieves an impressive ambiguity reduction, i.e. the overlap between curves is drastically reduced. This phenomenon is further increased when the Support Subspace operator is employed as shown in Figure 4.1 (b) and (d): notice how the mean value of the distribution of semantically related word is significantly increased for both languages.

The probability of error reduction can be computed against the control groups. It is the decrease of the error probability of a DCS relative to the same estimate for the control (i.e. V−V) group. It is a natural estimator of the generalization capability of the involved operators. In Table 4.7 the intersection area for all the models and the decrement of the relative probability of error are shown. For English, the ambiguity reduction of the Support Subspace operator is of 91% with respect to the control set. This is comparable with the additive operator results, i.e. 92.3%. It confirms the findings of the previous experiment where the difference between these operators is negligible. For Italian, the generalization capability of support subspace operator is more stable, as its error reduction is of 62.9% with respect to the additive model, i.e. 54.2%.
4.5 **DCS model based on Support Subspace for Textual Similarity**

The investigation over compositional approaches to language understanding and the definition of flexible and robust models for text semantic similarity tasks is still an open challenge. In Chapter 4.2, in order to determine semantic metrics for phrases, a distributional compositional model based on space projection guided by syntagmatically related lexical pairs has been defined.

This represents a novel perspective on compositional models over vector representations that captures lexical meanings locally relevant to the specific context evoked by each phrase. The proposed projection-based method of DCS seems to be very effective for syntactic structures like \textit{VO}, \textit{NN} and \textit{AdjN} and empirical results show its robustness in terms of providing semantic information consistent with the meaning assignment that is typical of human subjects. In a lexical pair similarity task, e.g. \cite{Mitchell and Lапata(2010b)} and in an IR scenario, such that model seems robust in representing lexical pairs and in separating pairs of different semantic classes.

Since lexical pairs contain syntactic relations of quite a different kind, we would expect a different Support Subspace for each syntactic relation, according with \cite{Baroni and Zamparelli(2010)} that focuses their attention only on a \textit{AdjN} compositional model and even train a single model for each adjective in their corpus. Investigation on defining specific models based on Support Subspace for each different type of lexical pairs is still missing and a work in progress.

Moreover is fair to ask how to use this model to account for the meaning of an entire sentence. A recursive application of such that model in which select the proper Support Subspace evoked by all the lexicon of a sentence, is possible, but the question...
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

is how effective could be a thigh resulting sentence support subspace, maybe composed only by few remaining components. Anyway, as for Word Space, robust to represent only single words in isolation, the DCS model introduced above seems to represent general (i.e. syntactically free) pairs.

Therefore a deeper investigation on how to i) account for the meaning of a sentence and ii) select the right Support Subspace depending on pairs syntactic relations are need. In order to investigate point i) in this Section the DCS model defined in Section 4.2 is used in a more complex task in order to assess the compositional operators’ capability in modeling semantic relatedness between entire sentences.

An effective method to compute similarity between short texts or sentences has many applications in Natural Language Processing [Mihalcea et al. (2006)] and related areas such as Information Retrieval, e.g. to improve the effectiveness of a semantic search engine [Sahami and Heilman (2006)], or databases, where text similarity can be used in schema matching to solve semantic heterogeneity [Islam and Inkpen (2008)].

In order to measure the benefit of our DCS model in accounting the meaning of sentences, we targeted the Semantic Textual Similarity (STS) task proposed at SemEval, as presented in [Agirre et al. (2012)]. Semantic Textual Similarity (STS) measures the degree of semantic equivalence between two phrases or texts. Similarity scores between sentence pairs are here provided by annotators: scores range between 0 (uncorrelated pairs) and 5 (identical pairs). Competing systems were asked to provide scores (not necessarily in the same range) whereas performances were measured through the Pearson Correlation with respect to human judgments. As text similarity strictly depends on the similarity at lexical level as well as on the equivalence of more complex syntagmatic structures, the STS is ideal for evaluating the impact of compositional measure
4.5. DCS model based on Support Subspace for Textual Similarity

of semantic similarity in a realistic setting. We want to measure the contribution of the Support Subspace operators with respect to simple distributional models that do not account for syntactic information.

In the SemEval STS challenge, five datasets made of sentences derived from different corpora and modeling different aspects of similarity have been provided: MSRvid includes 1,500 short text pairs, from the Microsoft Research Video Description Corpus (MSR-Video); MSRpar is made of 1,500 sentence pairs from Microsoft Research Paraphrase Corpus; SMTeuroparl contains about 1,200 sentences of WMT2008 development dataset, derived from the Europarl corpus and it is made of syntactically complex sentences; OnWn comprised 750 pairs of glosses from OntoNotes 4.0 [Hovy et al. (2006)] and WordNet 3.1senses; SMTnews contains 351 pairs of news conversation sentence pairs from WMT.

A shallow baseline model, named as SUM hereafter, has been defined. It uses the additive operator between the distributional representation of lexical vectors, as described in [Mitchell and Lapata (2008)]: a sentence is represented by the sum of its word vectors and the overall similarity function between two sentences is the cosine similarity between their corresponding composed vectors. The word space derived from the ukWaC is employed to represent every word in the experiment.

STS against two sentences has at least three challenges in the compositionality perspective. First, a basic measure of lexical semantic similarity is needed, i.e. similarity at the lexical level. Second a compositional measure of similarity is needed between basic structures, e.g. syntactically typed bigrams. Third, we need a method to match

---

1As the distributional information of very frequent words, such as Preposition or Articles, is not very informative for the overall representation, only Nouns, Verbs, Adjective and Adverbs are employed in the combination.
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

Figure 4.2: Example of dependency graph

In order to handle sentences, we thus first converted them in syntactic representations compatible with the compositional operators proposed. A dependency grammar based formalism captures binary syntactic relations between the words, expressed as nodes in a dependency graph. Given a sentence, the parse structure is acquired and different triples \((w_1, w_2, r)\) are generated, where \(w_1\) is the relation governor, \(w_2\) is the dependent and \(r\) is the grammatical type\(^2\). In the tests, only relations linked to words whose Part-of-Speech is Noun, Verb, Adjective or Adverbs are retained. Moreover, some simple heuristics are applied to simplify specific syntactic structures, e.g. in a Verb Chain structure, the auxiliary and the main verb are merged. For example, Figure 4.2 shows the dependency graph of the sentence “A man is riding a bicycle” and the extracted triples are: \((\text{ride.v,man.n,OBJ})\), \((\text{ride.v,bicycle.n,OBJ})\). The same is done with prepositions that are merged with their dependency label, so that lexicalized

\(^2\)In each relation the word from which the edge starts is called governor, while the pointed one is called dependent.
4.5. DCS model based on Support Subspace for Textual Similarity

prepositional types (e.g. BYPMOD) are used, e.g. \((\text{travel, bike}, \text{BYPMOD})\) models “... traveling by bikes ...”.

Finally, a strategy is needed to select the obtained triples in the two sentences and to derive pairs on which different compositional operators will be applied. In this investigation, we defined a simple approach based on the notion of Soft Cardinality, inspired by the work of [Jimenez et al. (2012)]. Differently from the classical Set Cardinality, that counts the number of different elements in a set, Soft Cardinality provides a softer way to count, based on the notion of similarity between elements. Given a triple set \(T = \{t_1, \ldots, t_n\}\) extracted from a sentence \(S\) and a similarity measure \(sim(t_i, t_j)\), the Soft Cardinality is estimated as:

\[
|S|_{sim}' \approx \sum_{t_i} \sum_{t_j} \frac{1}{sim(t_i, t_j)^p}
\]  \hspace{1cm} (4.18)

For example, given a set of three different elements, Soft Cardinality of a set with two very similar (but not identical) elements and one dissimilar should be a real number closer to 2 instead of 3. As in [Jimenez et al. (2012)], parameter \(p\) in Eq. (4.18) controls the “softness” of the cardinality: with \(p = 1\) element similarities are unchanged while higher value will tend to the Classical Cardinality measure. In this experiment we set \(p = 2\). Notice that differently from the previous usage of the Soft Cardinality notion, we did not apply it to sets of individual words, but to the sets of dependencies (i.e. triples) derived from the two sentences. The \(sim\) function here can be thus replaced by any compositional operator among the ones discussed in Section 4.2.

In the following experiment the Support Subspace model described in Eq. (4.17), i.e. \(\Phi_{12}^{(o)}(p_1, p_2) = \Phi_{1}^{(o)}(p_1, p_2) \odot \Phi_{2}^{(o)}(p_1, p_2)\), is employed, where a subspace dimension of \(k = 200\) has been chosen after tuning on the training set. Given two sentences,
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

the Soft Cardinality is employed to measure the amount of compositional information shared between the two texts. Higher values mean that the element in both sentences (i.e. triples) are different, while the lower values mean that common triples are identical or very similar, suggesting that sentences contain the same kind of information. Given the sets of triples $A$ and $B$ extracted from the two candidate sentences, our **Syntactic Soft Cardinality** (SSC) approach estimates the STS as:

$$SIM(A, B) = 2\frac{|A \cap B|'}{|A|' + |B|'} \approx 2\frac{(|A|' + |B|' - |A \cup B|')}{|A|' + |B|'} \quad (4.19)$$

as a “soft approximation” of Dice’s coefficient calculated on both sets. Notice that, since the intersection $|A \cap B|'$ tends to be too strict, we derive it from the union cardinality estimation $|A|' + |B|' - |A \cup B|'$.

The following three models were compared. The first consists in the **SUM** baseline that acts only on the lexical vectors on which a unique vector for a sentence is built. The SSC approach consists in applying Equation 4.19 to the syntactic triples of the two targeted sentences, oriented to model only their syntagmatic analogies. Different syntactic relations are thus allowed to contribute. Finally, a combination of the SUM baseline and the SSC model is provided as a linear combination with equal weights, named **SUM+SSC**: it allows to combine the contribution of the purely lexical information (modeled by SUM) and the syntactic and compositional representation of the SSC method. Table 4.8 shows the Pearson correlation results of the three model over the different datasets of the STS challenge. The last row averages all Person correlation scores across all the collections to give an idea of the general behavior of the methods.

As a general impression, it must be said that the SUM operator is achieving very good performances in some collections. Among the lexicalized models tested it was the best one. In the MSRvid dataset, syntactic information is not much discriminant as
4.5. DCS model based on Support Subspace for Textual Similarity

Table 4.8: Results over the STS task

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SUM</th>
<th>SSC</th>
<th>SUM+SSC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSRvid</td>
<td>.67</td>
<td>.50</td>
<td>.62</td>
</tr>
<tr>
<td>MSRpar</td>
<td>.39</td>
<td>.56</td>
<td>.54</td>
</tr>
<tr>
<td>SMTEuroparl</td>
<td>.47</td>
<td>.47</td>
<td>.51</td>
</tr>
<tr>
<td>OnWN</td>
<td>.67</td>
<td>.65</td>
<td>.69</td>
</tr>
<tr>
<td>SMTnews</td>
<td>.46</td>
<td>.41</td>
<td>.48</td>
</tr>
<tr>
<td><strong>AVG</strong></td>
<td>.53</td>
<td>.52</td>
<td>.57</td>
</tr>
</tbody>
</table>

almost all sentences are characterized by very simple structures, e.g. a SUBJ-VERB-OBJ pattern, and syntax seems not to add any useful information. The SUM approach is thus sufficient to provide the best results. It can be noticed that in several cases, e.g. “A man play guitar” and “A man play a flaute”, the SSC provides too high estimates (e.g. between 4 and 5) as they are too much polarized on the semantics underlying compositions (e.g. selectional preferences of a verb like play) rather than on the overall sentence similarity.

In other datasets of the SemEval corpus syntax is more informative, e.g. MSRpar, SMTEuroparl and OnWN. In particular, the MSRpar dataset was developed as a test set for paraphrasing: here parse trees provides direct constraints over paraphrases, as discussed in [Pang et al. (2003)]. It is exactly on this collections that the contribution of the SSC model is more effective. In general when syntax is more important as the coverage of the lexicon is lower, the SSC model is equivalent to SUM or better. In sentences like “That exploit works on unpatched Windows 2000 machines with Service Pack 3 and 4” against “Both Counterpane and iDefense contend that the exploit works effectively against Windows 2000 systems running Service Pack 3 and 4”, the use of proper nouns (Counterpane or iDefense) in only one sentence (i.e. the second) affects the quality of the SUM method: proper nouns are not well estimated in the distribu-
Chapter 4. Distributional Compositional Semantic Models as a model for meaning

tional analysis and also their contribution is asymmetric bringing to scores diverging from humans. On these sentences the SSC method is more robust and tighter to the most relevant syntagmatic relations “machine with Service Pack” vs. “running Service Pack”.

In other words the SSC seems to complete the contribution of the lexicon (well captured by the SUM method) by greatly increasing the overall robustness of the system across collections. The last row in Table 4.8 suggests in fact that the combined system SUM+SSC is outperforming the other models on average. It is interesting to notice that results improve of about 9% with combined models (i.e. SUM+SSC), that further confirms that the compositional operator proposed in the paper achieve more robust representations than well performing fully lexicalized models (e.g. SUM).

Notice that the presented approach for the STS estimation is fully unsupervised: training examples of the challenge have been here employed only to estimate the most suitable value of $k$ in Eq. 4.17.

In the next Chapter a brand new textual similarity kernel function based on Smoothed Partial Tree Kernel and DCS models based on subspace projection will be defined. Then, this function will be evaluated on the SemEval STS competition dataset, first in an unsupervised scenario (as for the SSC function presented above) in order to evaluate its contribution among the other Partial Tree Kernel functions, then in a supervised scenario where a SV regressor combines different textual similarity models, learning a scoring function that weights individual scores in a unique resulting STS.
Distributional Compositional Semantics in Tree Kernel Based Structured Learning

In Chapter 3, vectorial representation of lexicon based on distributional analysis has been introduced. A metric such as LSA has been discussed as useful to highlight relations between vectors in such a space. Furthermore, an algebra has been used to gather a metric sensible to grammatical relation, for simple syntactic bi-grams (e.g., VO). Thus, DCS model based on Support Subspace, introduced in Chapter 4, have proved to be a robust metric to account similarity between simple lexical pairs.

The objective is now to define a measure of semantic similarity in order to support complex tasks of textual inference, such as Question Classification or Textual similarity. The aim of this Chapter is to define a compositional model able to scale to more complex phenomena, by using the DCS model to account the meaning of an entire sentence in which vectors are three or more. The DCS model introduced in Chapter 4 capture bi-gram semantics, but it is not sensitive to the syntactic structure yet.

It is known from machine learning studies that the representation is distinct from the training function: the first is demanded to the Kernel, the latter to the SVM.

The purpose of this Chapter is to investigate a metric that is a valid kernel to account similarity between sentences. Starting point is the theory of arboreal Kernel [Collins and Duffy(2001)] which consider only the syntax information and seman-
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

tic Context Free rules, and the extension proposed by [Croce et al. (2011c)], called Smoothed Partial Tree Kernel (SPTK): here semantic information related to the lexical leaves of a parse tree is pursued for smoothing, by using the recursive tree kernel matching enriched by a lexical-semantic metric (LSA).

In our research perspective, the parse tree became the context in which the semantic of words is expressed. In [Croce et al. (2011c)] leaves are expressed by lexical vectors; in our prospective the semantic information will be distributed across all the parse tree as a carrier of the lexical composition, e.g. head/modifier, already explicit in dependency formalism.

Figure 5.1: Charniak constituency tree of the sentences (a) I make a gift to my friends and (b) I make room for my friends

Take as an example the constituency parse tree of Figure 5.1. The two sentences are syntactically very similar and while lexically they differ (unless stopwords) just for the words gift/room that, bind to the verb to make, evokes two different meanings for the sentences. In fact, if the first one is related to the making of a present for friends,
the latter is about “creating space” for someone. Notice how tree kernel functions tend to assign a strong correlation between such sentences, since the tree structures are similar, confining the rating process on a syntactical and symbolic level. In the SPTK approach, discrimination between word pair *gift/room* is delegated only on the LSA metric on the leaf level, still keeping a strong syntactic similarity between sentences.
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

The idea here is to propagate lexical semantic information over the entire parse tree, by building a compositional constituency tree as depicted in Figure 5.2. The entire process of marking these parse trees in such that way is described in the following Sections, moreover it is clear that such compositional marking could be very useful in discriminating sentences, by propagating semantic information over the tree. In Figure 5.2 the verbal phrases carry on two different semantic information expressed respectively by the lexical bigram \textit{make.v-gift.n} and \textit{make.v-room.n}. Notice that the bigrams represent respectively the most important information of the subtree (i.e. the head) and its modifier, and that they are compatible with the simple grammatical pairs of the DCS model of Chapter 4 has been tested on. By making non-terminal nodes dependent on both syntactic (e.g. the \texttt{VP} grammatical category) and semantic information, it is possible to formulate a brand new Kernel function based on this compositional tree representation, that takes into account for each node a distributional compositional metrics. Thus, the idea is to:

i. use the SPTK formulation in order to exploit the lexical information of the leaf,

ii. define a procedure to mark nodes of a constituency parse tree that allow to spread lexical bigrams across the non-terminal nodes,

iii. apply smoothing metrics sensible to the compositionality between the non-terminal labels,

iv. define a metric between nodes that relies on DCS models introduced in 4

Thus, this compositionally marked parse tree will be used in a new version of a SPTK that allows to model a compositional metric between lexical pairs (i.e. head/modifier)
related to non-terminal nodes. The resulting model has been called Compositionally Smoothed PTK (CSPTK) and discussed hereafter.

In Section 5.1, a quick summary of a specific class of Kernels, i.e. the Tree Kernels, that represent an effective kernel formulation to define language learning systems, is presented. Therefore, in the next Sections the Compositional SPTK is introduced (5.2), by dwelling on the heuristic marking of the parse tree (5.3) and the explanation of a new kernel similarity function (5.4) applied on the tree representation of Figure 5.2. In Section 5.5 the CSPTK model is investigated in the Semantic Text Similarity (STS) task introduced in Chapter 4.

5.1 Tree Kernels

Convolution Kernels, introduced in [Haussler(1999)], determine a similarity function among discrete structures, such as sequences, strings or trees. The estimation involves a recursive calculation over the “parts” of a structure, e.g. String Kernels (discussed in [Lodhi et al.(2002)]) and Tree Kernels, discussed in [Collins and Duffy(2002)].

String kernels (SKs) represents one of the first defined Convolution Kernels for texts similarity. In general, similarity is assessed by the number of (possibly non-contiguous) matching subsequences shared by two sequences. Non contiguous occurrences are penalized according to the number of gaps they contain. SKs have been applied with interesting results to the document classification tasks, where sentences have been modeled as sequences of characters [Lodhi et al.(2002)] or sequences of words [Cancedda et al.(2003)]. This kernel does not only account on lexical information (i.e. the words) but also on word ordering, capturing a first form of syntactic information.
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

Tree Kernels (TKs) allow to estimate the similarity among texts, directly from the sentence syntactic structures, represented by trees, e.g. the constituency parse trees introduced in section[2]. TKs are clearly more expressive than SK for NL tasks, especially for semantic processing tasks, where syntax is often a crucial feature. The idea is that a tree $T$ can be seen as a set of fragments and the similarity between two trees $T_1$ and $T_2$ can be derived from the number of shared subtrees.

Implicitly, a TK function establishes a correspondence between distinct fragments and dimensions in some fragment space $\mathcal{F} = \{f_1, \ldots, f_i, \ldots, f_{|\mathcal{F}|}\}$, i.e. the space of all the possible fragments. A projection function $\Phi(T)$ can be defined to map each $T$ in a vector whose $i$-th component counts the occurrences of individual fragments $f_i$ within the tree. As the number of possible fragments can largely grow, an explicit formulation of $\mathcal{F}$ could not be feasible. This problem is avoided just estimating the target TK function $TK(T_1, T_2) = \Phi(T_1)\Phi(T_2)$ in such space only on the basis of the number of common fragments between $T_1$ and $T_2$. To define such function, let $\chi_i(n_j)$ be an indicator function, equal to 1 if the target fragment $f_i$ is rooted at node $n_i$ and equal to 0 otherwise. A tree-kernel function over $T_1$ and $T_2$ is defined as:

$$TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$$  \hspace{1cm} (5.1)

where $N_{T_1}$ and $N_{T_2}$ are the sets of the $T_1$’s and $T_2$’s nodes respectively, and the recursive function $\Delta(n_1, n_2) = \sum_{i=1}^{(|\mathcal{F}|)} \chi_i(n_1)\chi_i(n_2)$ estimates the number of common fragments rooted in the $n_1$ and $n_2$ nodes. This similarity can be normalizes to have a similarity score between 0 and 1, a normalization in the kernel space, i.e.

$$TK'(T_1, T_2) = \frac{TK(T_1, T_2)}{\sqrt{TK(T_1, T_1) \times TK(T_2, T_2)}}$$

can be applied as it is still a valid kernel [Shawe-Taylor and Cristianini(2004a)].
5.1. Tree Kernels

The $\Delta$ function determines the richness of the kernel space and thus different tree kernels. Hereafter, we consider the equation to evaluate Syntactic Tree Kernel, introduced in [Collins and Duffy (2002)], and Partial Tree Kernel, introduced in [Moschitti (2006a)].

5.1.1 Syntactic Tree Kernels

Syntactic Tree Kernels (STKs) [Collins and Duffy (2002)] estimate the similarity among tree structures imposing linguistic constraints over the resulting computation, i.e. not all tree fragments are considered, but only those syntactically justified. As STKs aims to capture the syntactic information of sentences, e.g. represented through a constituency parse tree (CT), it implicitly determines a feature space $\mathcal{F}$ whose dimensions reflect only fragments motivated by valid syntactic production rules. The most commonly used system for modeling constituent structure in almost all natural languages is the Context-Free Grammar (CFG), as discussed in Section 2. A CFG consists of a set of rules or productions expressing the ways symbols are rewritten to form the tree. In Figure 5.3 (on the left) a constituency tree of the sentence “Time flies like an arrow” is shown. Here, the application of CFG rules such as $S \rightarrow NP\ VP$ is reflected from the fragment $(S(NP\ VP))$. The production rules cannot be ignored so that a node $n_1$ can be considered equal to a node $n_2$ only if they have the same label and same production rules. It means that a fragments $(S(NP\ NP\ VP))$ will not be compatible with $(S(NP\ VP))$.

The $\Delta_{STK}(n_1, n_2)$ is thus recursively computed as follows:

- if the productions at $n_1$ and $n_2$ are different then

$$\Delta_{STK}(n_1, n_2) = 0$$
• if the productions at $n_1$ and $n_2$ are the same, and $n_1$ and $n_2$ have only lexical leaf children then

$$\Delta_{STK}(n_1, n_2) = \lambda$$

• if the productions at $n_1$ and $n_2$ are the same, and $n_1$ and $n_2$ are not pre-terminals then

$$\Delta_{STK}(n_1, n_2) = \lambda \prod_{j=1}^{l(n_1)} (1 + \Delta_{STK}(c^j_{n_1}, c^j_{n_2}))$$

(5.2)

where $l(n_1)$ is the number of children of $n_1$ and $c^j_{n}$ is the $j$-th child of the node $n$. Unfortunately, the STK value tends to be largely dominated by the size of two input trees: if they are large in size, it is highly probable for the kernel to accumulate a large number of overlapping counts in computing their similarity. To alleviate such problem, in [Collins and Duffy(2001)] a scalability parameter is introduced: it is called decay factor, $0 < \lambda \leq 1$ as it scales the relative importance of tree fragments with their sizes.

![Figure 5.3: Some tree fragments obtained for the sentence “Time flies like an arrow” with the STK](image)

An excerpt of the fragments considered by the $\Delta_{STK}$ function over the sentence “Time flies like an arrow” is shown in Figure 5.3.
The syntactic constraints imposed by the STK avoid to consider fragments that are not linguistically motivated, thus do not reflect the criterion of “well-formed” sentences. Notice how, a partial difference between a production rule is considered a mismatch with no contribution to the overall kernel. This can be problematic especially when the training dataset is small and few linguistic configurations are made available. Consider the fragment (NP (DT NN)), covering a syntagm like “an arrow”, or (NP (DT JJ NN)), covering something like “a red arrow”: they clearly share some information that a STK would discard, i.e. (NP (DT NN)). These Partial Trees can be generated by the application of partial production rules of the grammar. Obviously the order of production in a language is important, i.e. (NP (NN DT)) will not arise a well-formed sentence. A production can be seen as a set of ordered and possibly non-contiguous subsequences, as well as in the String Kernel formulation. As an example (NP (DT JJ NN)) is composed form (NP (DT NN)), (NP (DT JJ)), (NP (JJ NN)), (NP (DT)), (NP (JJ)) and (NP (NN)), that are called Partial Trees.

This proliferation of fragments is obtained by removing constraints over production rules in the STK formulation, thus obtaining more general substructures. The computation of Partial Tree Kernels (PTKs), [Moschitti(2006a)], is carried out by the following $\Delta_{PTK}$ function:

- if the labels of $n_1$ and $n_2$ are different then

$$\Delta_{PTK}(n_1, n_2) = 0$$
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

\[ \Delta_{PTK}(n_1, n_2) = 1 + \sum_{I_1, I_2, l(I_1) = l(I_2)} \prod_{j=1}^{l(I_1)} \Delta_{PTK}(c_{n_1}(I_{1j}), c_{n_2}(I_{2j})) \]  \hspace{1cm} (5.3)

where \( c_{n_1} \) and \( c_{n_2} \) are the ordered child sequences of \( n_1 \) and \( n_2 \) respectively, while \( I_1 = \langle I_{11}, I_{12}, I_{13}, \ldots \rangle \) and \( I_2 = \langle I_{21}, I_{22}, I_{23}, \ldots \rangle \) are index sequences associated with the ordered child sequences such that \( c_{n_1}(I_{1j}) \) and \( c_{n_2}(I_{2j}) \) are the \( j \)-th children in the two sequences respectively. The function \( l(\cdot) \) returns the sequence length. Additionally, we add two decay factors: \( \mu \) for the height of the tree and \( \lambda \) for the length of the child sequences. Equation 5.3 can be rewritten as:

\[ \Delta_{PTK}(n_1, n_2) = \mu \left( \lambda^2 + \sum_{I_1, I_2, l(I_1) = l(I_2)} \lambda^{d(I_1) + d(I_2)} \prod_{j=1}^{l(I_1)} \Delta_{PTK}(c_{n_1}(I_{1j}), c_{n_2}(I_{2j})) \right) \]  \hspace{1cm} (5.4)

It follows that both larger trees and subtrees built on child subsequences containing gaps are penalized, where \( d(I_1) = I_{1l(I_1)} - I_{11} \) and \( d(I_2) = I_{2l(I_2)} - I_{21} \). In this way, both larger trees and child subsequences with gaps are penalized. PTK is more general than the STK as if we only consider the contribution of shared subsequences containing all children of nodes, we implement the STK kernel. An example of fragments that are considered by the \( \Delta_{PTK} \) function is shown in Figure 5.4. As for the previous STK kernel, each fragment is a dimension of the high-dimensional fragment space \( \mathcal{F} \). The main difference is that PTK is a richer space, as the production rules are not taken into account. This means that the sub-tree \((S \ (NP \ VP))\) generates the following novel fragments: \((S \ (NP \ VP))\) as the STK, but also \((S \ (NP))\) and \((S \ (VP))\).
5.1. Tree Kernels

Figure 5.4: Some tree fragments obtained for the sentence “Time flies like an arrow” with the PTK

5.1.3 The Smoothed Partial Tree Kernel

The novelty of SPTK is represented by the embedding of a similarity function $\sigma_\tau$ between nodes which are typed according to a syntactic category. It is general with respect to the SSTK as it depends on the position of the node pairs within the trees, i.e. non terminals nodes and leaves. Furthermore, the overall SPTK is neutral with respect to the target linguistic problems discussed in this Thesis. Obviously, the similarity function between nodes must be carefully designed in order to grant effectiveness in the target semantic processing task: in fact, the SPTK would enumerate and compare any possible node pairs, including non terminal nodes. From a linguistic perspective this is problematic as each node reflects a specific aspect of data and the comparison between nodes of different nature e.g syntactic nodes like NP or VP, and lexical nodes like apple or orange should be avoided. The similarity function $\sigma_\tau(n_1, n_1)$ between two nodes $n_1$ and $n_2$ must depend on the nodes’ type $\tau$. This would apply different criteria along different information, such as syntactic categories, pos-tags or lexical entries. An example of $\sigma_\tau$ is shown by Algorithm 1: given two nodes $n_1$ and $n_2$, it applies a different similarity for each node type. Types are described by $\tau$ and
are divided into: syntactic categories (i.e., \( \tau = \text{SYNT} \)), POS-Tag labels (i.e., \( \tau = \text{POS} \)) or a lexical (i.e., \( \tau = \text{LEX} \)) type. In this example we require a strong matching between non lexical nodes, i.e. assigning 0/1 similarity for \( \text{SYNT} \) and \( \text{POS} \) nodes. In Section 3 a lexical similarity function that is a valid kernel, i.e. Equation 3.1, has been introduced.

Here for \( \text{LEX} \) type the same kernel function, i.e. \( \sigma_{\text{LEX}} \) and it is applied between words sharing the same POS-Tag, thus words which belong to different shallow grammatical classes are never considered compatible, e.g., nouns with a verbs or adjectives.

As for the evaluation of PTK, the evaluation of the common SPTK rooted in nodes \( n_1 \) and \( n_2 \) requires the selection of the shared child subsets of the two nodes. Due to the importance of the order of the children, we can use subsequence kernels for their generation. More in detail, let \( \mathcal{F} = \{ f_1, f_2, \ldots, f_{|\mathcal{F}|} \} \) be the set of all possible PT fragment and let the indicator function \( I_i(n) \) be equal to 1 if the target \( f_i \) is rooted at node \( n \) and 0 otherwise, we define the SPTK as:

- If \( n_1 \) and \( n_2 \) are leaves then
  \[
  \Delta_{\text{SPTK}}(n_1, n_2) = \mu \lambda \sigma_{\tau}(n_1, n_2)
  \]

- Else
  \[
  \Delta_{\text{SPTK}}(n_1, n_2) = \mu \sigma_{\tau}(n_1, n_2) \times \left( \lambda^2 + \sum_{I_1, I_2, l(I_1) = l(I_2)} \lambda^{d(I_1) + d(I_2)} \prod_{j=1}^{l(I_1)} \Delta_{\text{SPTK}}(c_{n_1}(I_{1j}), c_{n_2}(I_{2j})) \right)
  \]

(5.5)

Here the formulation is similar to the PTK, \( c_{n_1} \) and \( c_{n_2} \) are the ordered child sequences of \( n_1 \) and \( n_2 \) respectively, while \( I_1 = \langle I_{11}, I_{12}, I_{13}, \ldots \rangle \) and \( I_2 = \langle I_{21}, I_{22}, I_{23}, \ldots \rangle \) are index sequences associated with the ordered child sequences such that \( c_{n_1}(I_{1j}) \) and \( c_{n_2}(I_{2j}) \) are the \( j \)-th children in the two sequences respectively. The function \( l(\cdot) \) returns the sequence length. As for PTK, two decay factors are employed: \( 0 < \mu \leq 1 \) for
5.1. Tree Kernels

Algorithm 1 $\sigma_\tau(n_1, n_2, lw)$

\[
\begin{align*}
\sigma_\tau &\leftarrow 0, \\
\text{if } \tau(n_1) = \tau(n_2) = \text{SYNT} \land \text{label}(n_1) = \text{label}(n_2) \text{ then } \\
\sigma_\tau &\leftarrow 1 \\
\text{end if} \\
\text{if } \tau(n_1) = \tau(n_2) = \text{POS} \land \text{label}(n_1) = \text{label}(n_2) \text{ then } \\
\sigma_\tau &\leftarrow 1 \\
\text{end if} \\
\text{if } \tau(n_1) = \tau(n_2) = \text{LEX} \land \text{pos}(n_1) = \text{pos}(n_2) \text{ then } \\
\sigma_\tau &\leftarrow \sigma_{\text{LEX}}(n_1, n_2) \times lw \\
\text{end if} \\
\text{return } \sigma_\tau
\end{align*}
\]

the height of the tree and $0 < \lambda \leq 1$ for the length of the child sequences. It follows that both larger trees and subtrees built on child subsequences that contain gaps are penalized depending on the exponent $d(I_1) = I_{1l(I_1)} - I_{11}$ and $d(I_2) = I_{2l(I_2)} - I_{21}$, i.e. the width of the production rule.

The lexical similarity function is therefore crucial in order to provide a meaningful kernel estimation. As discussed in the following chapters when focusing on empirical evaluations, this lexical kernel can be acquired from an existing lexicon or directly through Distributional modeling. Indeed, such general formulation also allows weighting differently each similarity function. For examples, in Algorithm 1 the contribution of the lexical information is amplified (or reduced) through a lexical weight (lw), that multiplies the similarity function between lexemes.

SPTK can encode generalized syntactic patterns from dependency or constituency structures. It measures the similarity between syntactic structures, which are partially similar and whose lexical nodes can be different but related. In [Croce et al. (2011a)] SPTK has been tested on a question classification (QC) task, achieving the state-of-the-art results. Here, the error analysis revealed that syntactic structures combined with distributional information about lexicals is needed to capture the question semantics.
In [Croce et al. (2012c)] SPTK has been used over structural representation of verbs based on syntactic dependencies to measure similarity between such representations: results on verb classification show a large improvement over the state-of-the-art for both VerbNet and FrameNet. Finally a SVM-SPTK system employed for the FrameNet based Semantic Role Labeling task is introduced in [Croce et al. (2011b)], achieving the state-of-the-art in almost all challenge tasks.

### 5.2 Towards a Compositional version of a SPTK

In order to describe the model of compositional tree kernel introduced in the previous section, a number of definitions are required. The input structure of a compositionally smoothed tree kernel, i.e. CSPTK, is a pair of trees whose nodes are enriched with lexical information needed for the recursive compositional matching foreseen by the adopted convolution model.

The syntactic structure of an individual sentence $s$ can be represented by a constituency-based parse tree, in which nodes are partitioned into three sets, called terminal (leaf), pre-terminal and non pre-terminal nodes, respectively.

- **Terminal nodes**, ($T$), i.e. the tree leaves that represent lexical information and part-of-speech of the words in $s$.
- **Pre-terminal nodes** ($PT$) are the direct ancestors of terminals and they have a unique son (i.e. the leaf they are linked with). The father of a pre-terminal node is strictly linked to only one node in $PT$.
- **Non Pre-terminal nodes** ($NPT$), i.e. nodes that are neither terminal nor pre-terminals.
5.2. Towards a Compositional version of a SPTK

Given the categories of a three node, we need to associate them with different types of information able to capture compositionality from a lexical as well as grammatical point of view.

5.2.1 Lexical syntactic marking of node for compositionality modeling

According to the definitions in Section 5.2 a generic terminal node (i.e. a leaf) \( n \in \mathcal{T} \) is noted as \( l_n : pos_n \), where \( l \) is the lemma of the token and \( pos \) its part-of-speech according to the Penn Treebank standard.

Otherwise, a generic pre-terminal node \( n \in \mathcal{PT} \) is marked through the part-of-speech \( pos \) of his unique corresponding leaf, i.e. \( pos_n \). Therefore, as seen in Figure 5.1 for a classical Costituency tree, the same labeling is adopted for terminal and pre-terminal nodes in a compositional constituency tree, i.e. that ones in Figure 5.2.

Finally, a more complex marking is required for generic non pre-terminal nodes \( n \in \mathcal{NPT} \). Non pre-terminal nodes in \( \mathcal{NPT} \) correspond to complex subtrees whose account of compositionality depends on at least the following types of syntactic as well as lexical information:

- **Grammatical Types.** Grammatical, or Tree, types \( GT \), describe the constituency information related to a sub-tree. An example of such types\(^\dagger\) are the nominal phrases (i.e. marked as \( np' \)s) or verb phrases (i.e. \( vp \)) as in Figure 5.2.

- **Lexical Information.** Non pre-terminal nodes express in general binary grammatical relations between a varying number of dominated subtrees (i.e. direct descendants). This can be expressed in terms of an head-modifier \((h, m)\) pair.

\(^\dagger\)A complete list of grammatical types, called tree types, is in Appendix A.
A complex sub-tree is thus characterized by the (structured) lexical information about the involved head ($l_h$) and modifier ($l_m$) lexicals. Notice that the grammatical head is the lexical element of a phrase that is essential to express the meaning of the corresponding phrase, and it is realized by one specific sub-tree rooted in the target non pre-terminal node. The modifier is one lexical, or phrasal element, the head is combined with in the sentence, e.g. an adjective for a noun. Modifiers thus include direct objects, indirect objects or predicative complements. The subtree corresponding to a complex non pre-terminal expresses information about its head $l_h$ and modifier $l_m$ through the 4-tuple

$$< l_h :: pos_h, l_m :: pos_m >$$

(5.6)

where $l$ and $pos$ correspond to the lemma and part-of-speech information as these are inherited upward the tree from the terminal nodes. Notice that when a complex non pre-terminal correspond to more than one binary grammatical relation only one among the represented relationships will be used to mark the root node. For this reason the 4-tuple $< l_h :: pos_h, l_m :: pos_m >$ is here called the minimal phrase corresponding to the targeted subtree.

- **Syntactic Relations.** A specific set of syntactical relations $SR$ is used to link the head to its modifier. It is also marked into root of the sub-tree. The relation is marked through a label, denoted by $rel_{h,m}$, expressing the grammatical information for $h$ and $m$, where $rel \in SR$. Examples of valid syntactic relations are: $np/vp$ for subject relationships or $vp,np$ for direct objects.

According to the definitions above, every non pre-terminal $n \in \mathcal{NPT}$ is marked with the following triple:
5.2. Towards a Compositional version of a SPTK

\[ <gT, syntRel, lexInfo > \]  \hspace{1cm} (5.7)

where \( gT \in GT \) \( syntRel = rel_{h,m} \), with \( rel \in SR \) and \( lexInfo \) is of the form of the 4-tuple in 5.6.

![Compositional constituency tree diagram](image)

**Figure 5.5:** Marking of a Compositional constituency tree of the sentences *A man is riding a bicycle*, derived through the Charniak parser.

In the example of Figure 5.5, where the parse tree of the sentence

*A man is riding a bicycle*

is depicted, all leaves (i.e. terminals) are marked as described with their lexical information and part-of-speech\(^2\). In the same figure, pre-terminal nodes \( (n \in PT) \) are marked with the full part-of-speech (i.e. pos) of the terminal nodes they dominate (e.g. the vbg node dominating the word *ride*).

The full complex marking is devoted to generic non pre-terminal nodes, i.e. \( n \in NPT \). One such \( n \in NPT \) is represented as the triple \( <g, syntRel, lexInfo > \)

\(^2\)General POS tags are obtained from the PennTreebank standard by truncating at their first char (as in *ride :: v*).
where fields are separated by a #. For example, the verbal phrase *a man is (riding a bicycle)* in Figure 5.5 is marked as:

\[
\langle VP#VBN/G/\# < ride :: v, bicycle :: n > \rangle
\]

Given the triple in Eq. 5.7 it is possible to provide a similarity metrics among sentence pairs through the recursive comparison of the making of their subtrees: in particular the similarity estimation between two (sub)trees correspond to the estimation of how much their two triples are semantically equivalent. Equivalence between two nodes in a three is modeled through a measure of the corresponding lexical similarity between nodes that show the same grammatical type and relations. Given a relation \(rel_{h,m}\) in the two trees, i.e. the pair \(rel_{h_1,m_1}\) and \(rel_{h_2,m_2}\), a compositional account of \(rel\) depending on the distributional semantic model of \(h_1, h_2\) and \(m_1, m_2\), can be defined through a real valued score. Hereafter, we will refer to these function as the

\[
\sigma_{Comp}(h_1, m_1, h_2, m_2)
\]

metrics acting over pairs of nodes in two trees. As discussed in Chapter 4.2, a similarity function based on geometric projection into Support Subspace, for the pairs \(p_1 = (h_1, m_1)\) and \(p_2 = (h_2, m_2)\) could be adopted as in equation 4.17 thus

\[
\sigma_{Comp}(p_1, p_2) = \Phi_1^{(o)}(p_1, p_2) = \Phi_1^{(o)}(p_1, p_2) \circ \Phi_2^{(o)}(p_1, p_2)
\]

where a composition function \(\circ\) between the similarities over the left and right support subspaces is applied. Remind that the left and right composition operators \(\circ\) may differ from the overall composition operator \(\circ\) and several projection functions \(\Pi^k\) may be adopted. As discussed in Chapter 4.2 it is possible to define variants of equation 5.9 according to (1) the size of the support selection (i.e. the value of \(k\)), (2) the
5.2. Towards a Compositional version of a SPTK

symmetry of the similarity judgment (i.e. symmetric and asymmetric projection), (3) the symmetry of the support subspace (i.e. independent or unified spaces) and (4) the composition function (i.e. \( \diamond \)).

5.2.2 Special Marking of specific grammatical phenomena

In some particular cases the pair head \( h \) and modifier \( m \) and the syntactic relation \( rel_{h,m} \) implied by a node in the tree is not well defined either because it has a null semantic content or because it is not possible to model the modifier through an effective distributional model. This is the case of stopwords, i.e. articles or pronouns, that may not be helpful in a composition. In this case the representation of a bigram like “the bicycle” is reduced to the distributional representation of “bicycle” since the article does not play a relevant semantic role in the composition.

**Modifier Elicitation.** One specific case involve noun phrase where a NPT node is to be marked. There are some relations where the modifier does not carry any relevant lexical information, as in the case of the relation between a determiner and a noun (e.g. \(<a,bicycle>)\). In this case, the modifier of a non pre-terminal node is neglected and the syntactic relation of the node is marked through a null slot, i.e. \( \ast \), as it is shown in Figure 5.5

\[
\text{syntRel} = rel_{bicycle,a} = (NN, \ast).
\]

The resulting triple for such special NPT is

\[
\langle NP\#NN/\ast\# < bicycle :: n, a :: d > \rangle,
\]

where the determiner \( a \) is overwritten by the null marker \( \ast \). In the marking algorithm also auxiliary verbs are neglected too, as for the verb \( is \) in the example. In this cases the semantics of the head is not modified by the modifiers.
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

Missing Modifiers. Another special marking still applies to non pre-terminals in \( \mathcal{NPT} \). In Figure 5.6, the sentence “A man with glasses is talking” is represented. Notice that when a non pre-terminal node \( n \in \mathcal{NPT} \) has only one pre-terminal node \( n \in \mathcal{PT} \) as a child, its modifier on the syntactic relation is missing and no relation is expressed. However, in order to preserve a notation consistent with other \( \mathcal{NPT} \) nodes marked with the same grammatical type (e.g. modified noun phrases) the relation is still represented as a binary predicate with a missing slot, i.e. \( \text{syntRel} = (h, \ast) \). In the example, the non pre-terminal nodes directly linked with the verb \textit{to talk} is marked as

\[ \langle \text{VP}\#\text{VBG}/\ast\# < \text{talk} :: v > \rangle \]

Notice that missing modifiers corresponds to special lexical information so that \( \text{lexInfo} \) is mapped into \( \text{lexInfo} = \langle l_h \rangle \) that describes only the head \( h \), e.g. \( < \text{talk} :: v > \).

![Figure 5.6: Compositional Charniak constituency tree of the sentences A man with glasses is talking](image)

Marking prepositional relationships. Finally, prepositional phrases constitute another special case when marking non pre-terminal nodes \( \mathcal{NPT} \). In order to generalize the matching of prepositional phrases as complex modifiers, the lexical information of
5.2. Towards a Compositional version of a SPTK

Figure 5.7: Compositional enrichment of the constituency tree for the sentences *A little girl is playing a grand piano on the stage* from a tree derived by the Charniak parser

prepositions is expressed both in the grammatical type and in the syntactic relation. For example in Figure 5.6, the phrase *with glasses* is developed in the following triple:

\[ \langle \text{with} | PP, \text{rel}(NN, *\text{with}*) \rangle, < \text{glasses} :: n > \]

where the grammatical type and the syntactic relation specify the preposition used, i.e. \( (\text{with} | PP) \) and \( (NN, *\text{with}*) \). Otherwise the lexical information, by eliciting the modifier, model only the semantic of the head, i.e. \( < \text{glasses} :: n > \).

In Figure 5.7, a more complex example where all the above marking rules are applied, is shown. Notice how the labels on each \( NPT \) node reflect the essential information of their corresponding subtrees, whereby the root node represent the core meaning of the sentence, i.e. that there is a *girl* that *plays*.
5.3 Mark-Up Rules for Constituency Trees

While marking terminal $T$ nodes and pre-terminal $PT$ nodes is quiet simple, as discussed in the above section the marking of non pre-terminal $NPT$ nodes could be challenging: special notations and several grammatical and syntactical rules has been adopted. Mainly these rules differ depending on (1) the type of the child $NPT$ node, i.e. if they are all pre-terminal or not, and (2) the arity of the branches, i.e. if binary or higher. Main general rules and some example for each case are provided in the following section, while in Appendix A all the possibly cases are discussed.

5.3.1 Binary branches

$NPT$ nodes with binary branches are simpler to be marked, since a pair of nodes are always involved. One node will be marked as the head and the other one as the modifier, unless some special cases, e.g. when determiner or auxiliary verbs are involved. The treatment of binary trees whose binary branches only involves pre-terminal nodes depends exclusively on terminal nodes $n \in T$. Given two pre-terminal nodes $(p_1, p_2)$ that link directly to their correspctive leaves it follows that

$$(l_1 :: pos_1, l_2 :: pos_2) \leftarrow (p_1, p_2)$$

from which it is possible to derive the following rule

$$pos_2/pos_1[h = p_2, m = p_1] \leftarrow (p_1, p_2) \quad (5.10)$$

where $[h = p_2, m = p_1]$ states that in general the second leaf node is the head, and the first is the modifier. Moreover, the formalism in Equation 5.10 expresses the syntactic relation by using the words pos-tags, i.e. $pos_1$ and $pos_2$. Here the rule for the
syntactic relation is $pos_2/pos_1$. For example for the phrase *big pepper* the resulting compositionally marked tree is derived as shown in Figure 5.9(b), by applying 5.10

$$NN/JJ \ [h = p_{pepper}, \ m = p_{big}] \leftarrow (p_{big}, p_{pepper})$$

Since mark-up rules rely only on the pair *pos-tags* it is possible to relax the Equation 5.10, by starting only from the *pos-tag* informations, thus

$$pos_2/pos_1 \ [h = p_2, \ m = p_1] \leftarrow (pos_1, pos_2) \quad (5.11)$$

In this way it is possible to generalized the rule for the phrase *big pepper*, by relying only on the *pos-tags*, defining the heuristic for the Adj/Noun pairs, thus

$$NN/JJ \ [h = p_2, \ m = p_1] \leftarrow (JJ, NN)$$

that specifies for a Adj/Noun phrase the mark-up rule for which the head is the noun and the modifier the adjective, and the syntactic relation is $NN/JJ$. 

Figure 5.8: Marking the charniak constituency tree of the phrase (a) *big pepper* and (c) *a woman* in the resulting (b) and (d) respectively compositional charniak constituency tree

In this way it is possible to generalized the rule for the phrase *big pepper*, by relying only on the *pos-tags*, defining the heuristic for the Adj/Noun pairs, thus

$$NN/JJ \ [h = p_2, \ m = p_1] \leftarrow (JJ, NN)$$

that specifies for a Adj/Noun phrase the mark-up rule for which the head is the noun and the modifier the adjective, and the syntactic relation is $NN/JJ$. 

99
Moreover in Figure 5.8d is shown an example where a modifier elicitation, i.e. the determiner, is applied for the phrase *a woman*, the following rule is adopted

\[ NN/ * [h = p_2] \leftarrow (CD, NN) \]

The treatment of binary trees whose binary branches involves pre-terminal nodes or non pre-terminal nodes is less trivial. While for *PT* nodes this is derived strictly from the leaf, the main difference with respect the Equation 5.10 is that i) \( p_1 \), ii) \( p_2 \) or iii) both branches could be a *NPT* node, already marked with a couple (*head, modifier*). Since we distinguish three cases, here the node \( p \) could be a simple pre-terminal or a non pre-terminal node as depicted in Figure 5.9. The mark-up rule identifies for each node \( p \) a representative grammatical type \( gT_r \) and a representative lexical information \( p_r = < lex_r :: pos_r > \), thus

- if \( p \in NPT \), \( p_r \) is derived from the *head* of \( p \), while \( gT_r \) is the grammatical type of the head of \( p \), as in Figure 5.9a where for the phrase *slit pepper* it follows that \( p_r = < slit :: v > \) and \( gT_r = VP \),

- if \( p \in PT \), \( p_r \) and \( gT_r \) are derived from the lexical information and the *pos-tag* of the leaf directly linked with \( p_r \), as in Figure 5.9b where \( p = VBG \) and it follows that \( p_r = < slit :: v > \) and \( gT_r = VBG \),

- if one node doesn’t express significant information for compositionality (e.g. auxiliary verbs), than it is discarded and the target *NPT* node is marked as the remaining node, as in Figure 5.9c where \( p = AUX \) is discarded, i.e. “is slitting pepper”.

Once \( gT_r \) and \( n_r \) for both branches are defined we have two nodes \((p_1, p_2)\) from
5.3. Mark-Up Rules for Constituency Trees

which it follows that

\[(p_r^1/gT_r^1, p_r^2/gT_r^2) \leftarrow (p_1, p_2)\]

It is possible now to modify Equation 5.10 thus

\[gT_r^2/gT_r^1 \ [h = p_r^2, m = p_r^1] \leftarrow (p_1, p_2)\]  \hspace{1cm} (5.12)

Since the rules relies only on the grammatical types involved, it is possible once again to simplify Equation 5.13 in

\[gT_r^2/gT_r^1 \ [h = p_r^2, m = p_r^1] \leftarrow (gT_r^1, gT_r^2)\]  \hspace{1cm} (5.13)

In Figure 5.9a, an example of marking a \(NPT\) node with \(NPT\) branches is shown. The representative nodes chosen \(p_1\) and \(p_2\) are the heads of the branches, i.e. \(\text{woman.n}\) and \(\text{slit.v}\), while the grammatical types are \(NP\) and \(VP\), thus

\[VP/NP \ [h = \text{slit.v}, m = \text{woman.n}] \leftarrow (NP, VP)\]

Figure 5.9: Marking the charniak constituency tree

(a) \hspace{1cm} (b) \hspace{1cm} (c)
from which follows the general rule for the $gT$ label like NounPhrase/VerbalPhrase
that is

$$VP/NP \ [h = p^2_r, m = p^1_r] \leftarrow (NP, VP)$$

Notice that if branches correspond to a $PT$ node and a $NPT$ node, the resulting
syntRel is an hybrid form mixing up pos-tag from the $PT$ node and grammatical
type from the $NPT$ node. This is the case depicted in Figure 5.9b where

$$VBG/NP \ [h = \text{slit}.v, m = \text{pepper}.n] \leftarrow (VBG, NP)$$

Finally in Figure 5.9c a modifier elicitation is shown, where the auxiliary is discarded
and the left branch label is used to mark the target $NPT$ node. Here the marking-up
rule that follow is

$$p^g_{Tr} \ [h = p^b_2, m = p^m_2] \leftarrow (elicit - gT_r, gT^2_r) \quad (5.14)$$

where the node $p_1$ is discarded and the node $p_2$ is the only source from which all the
information is copied in the father label. Thus in this particular example of Figure 5.9c
the information of $VBG$ node is reported in the father node

$$VBG/NP \ [h = \text{slit}.v, m = \text{pepper}.n] \leftarrow (AUX, VBG)$$

The complete list of marking rules for $NPT$ nodes with binary branches is reported
in Appendix A.

5.3.2 Flat or $n$-ary branches

Several $NPT$ nodes have more than 2 branches. These branches are here called flat.
As in the previous section, flat branches could be all pre-terminal nodes, otherwise they
could be in turn $NPT$ nodes. When flat branches are all $PT$ nodes, the marking task
5.3. Mark-Up Rules for Constituency Trees

is to identify head and modifier from leaves. Since each case of Equation 5.11 is listed in order of importance (i.e. from highest to lowest) in Appendix A, the first matching pair (head,modifier) that follows equation 5.11 is chosen to be representative for the target NPT node. An non-significant node like conjunction or determiner are here discarded.

![Constituency Tree Diagram](image)

(a)

![Constituency Tree Diagram](image)

(b)

Figure 5.10: Marking the charniak constituency tree

In Figure 5.10a, an example of marking a NPT node in case a flat branch consisting of only pre-terminal is met, is shown. Notice that the determiners are discarded, therefore the problem is equivalent to

$$NN/NN \ [h = woman.n, m = man.n] \leftarrow (NN, NN)$$

Moreover in Figure 5.10b branches are a mixture of PT and NPT nodes. In this case, once all non-significant nodes are discarded, the most important pair of nodes (from the complete list in Appendix A that satisfies Equation 5.11 is chosen: in the
example *put.v* is chosen as head and *towel.n* as modifier, reducing the problem to

\[ VBG/NNS \ [h = put.v, m = towel.n] \leftarrow (VBG, NNS) \]

By following these mark-up rules it is possible to represent syntactic composition in the so called compositionally parse trees. It could be interesting to efficiently and effectively measure similarity between this constituency structures. Thus the idea is to define a Kernel function able to measure similarity between these arboreal representation, enriched by compositional semantic. In Section 5.4 a Compositional Smoothed Partial Tree Kernel that relies on compositionally marked Charniak constituency tree will be thus defined. Furthermore, the CSPTK robustness is measured in experiments on the STS task already introduced Section 4.5

### 5.4 Extending the SPTK with compositional information: the Compositional SPTK

In this section we want to formulate a similarity function for the arboreal structure introduced in Section 5.2 in order to introduce a new Kernel, based on the SPTK formulation, enriched with the semantic composition of the marked-up head/modifier pairs introduced in Section 5.3. By starting from the SPTK formulation in Equation 5.5 and the marked-up versions of the three is depicted in Figure 5.7 the semantic smoothing algorithm \( \sigma_\tau \) turns into that described in Algorithm 2.

For the terminal nodes (i.e. LEX type) the same lexical kernel \( \sigma_{LEX} \) is applied between words sharing the same POS-Tag. Otherwise between pre-terminal nodes, a strong matching is required, assigning 0/1 similarity only if pre-terminal nodes share the same POS. The tricky part of Algorithm 2 is introduced with the similarity computation over non pre-terminal nodes. As introduced in Section 5.3, \( \mathcal{NP}T \) nodes in a
compositionally enriched constituency tree, are also marked with the triple composed by a grammatical type $gT$, a syntactic relation $\text{syntRel}$ and the lexical information $\text{lexInfo}$, thus a generic node $n \in NPT$ is marked up as in equation 5.7. According to Algorithm 2, in order to activate the similarity function between $NPT$ nodes, they must have $gT$ and $\text{syntRel}$ in common. If it is the case, $\text{lexInfo}$ pairs are checked and if their respective heads and modifiers share the corresponding $\text{pos}$ (i.e. head with head and modifier with modifier), finally a compositional similarity function is applied between the lexical pairs. The similarity function used between the $li$ pairs is the one derived from the DCS model introduced in Chapter 4, based on projecting lexical couple into subspaces, i.e. Equation 4.17 and reprised in Equation 5.9. Notice that as discussed in Section 5.2.2, modifier could be missing in lexical information pair, thus three different cases in which apply the DCS model are distinguished:

- Let call $li_x$ and $li_y$ the $\text{lexInfo}$ of two generic non pre-terminal nodes $x$ and $y$. If they have both heads and modifiers, i.e. $li_x = \langle h_x::\text{pos}_h, m_x::\text{pos}_m \rangle$ and $li_y = \langle h_y::\text{pos}_h, m_y::\text{pos}_m \rangle$, the similarity function of Equation 5.9 is applied as usual so

$$\sigma_{COMP}(\langle h_x, m_x \rangle, \langle h_y, m_y \rangle) = \phi_{COMP}(h_x, m_x, h_y, m_y)$$

Notice that the $\text{pos}$-tags of heads and modifiers must be the same.

- if a modifier is missing, e.g. $li_x = \langle h_x::\text{pos}_h \rangle$ and $li_y = \langle h_y::\text{pos}_h, m_y::\text{pos}_m \rangle$ the remaining modifier is applied to the single head node and the similarity function became

$$\sigma_{COMP} = \langle \langle h_x, m_y \rangle, \langle h_y, m_y \rangle \rangle$$
This gives us an “optimistic” estimator of similarity as the same modifier is assumed to restrict both pairs.

• finally if both the modifiers are missing, i.e. $li_x = <h_x::pos>$ and $li_y = <h_y::pos>$ the case is simplifying to the terminal nodes (i.e. LEX type) case, by using the lexical kernel $\sigma_{LEX}$ as follows

$$\sigma_{COMP} = ((h_x), (h_y)) = \sigma_{LEX}(n_x, n_y)$$

These three cases are shown in Algorithm 2. The first case follows the standard approach discussed in 4.2, while the third case is bringing back to a similarity case between two words, resolved by $\sigma_{LEX}(n_x, n_y)$. Finally in the second case only a modifier of a node $x$ is missing and the proposed approach forces the remaining modifier of node $y$ together with $h_x$ (i.e. the head with the missing modifier). In this way the forced pair $(h_x, m_y)$ and the pair $(h_y, m_y)$ projected and compared into their own subspaces, provide a measure of how the head $h_x$ is similar to $h_y$, with respect to the meaning that they evoke together with $m_y$. The more $h_x$ and $h_y$ could be juxtapose with $m_y$ to specify the same meaning, the more they receive an high score from $\sigma_{COMP} = ((h_x, m_y), (h_y, m_y))$.

Algorithm 2 could be modified depending on how the non pre-terminal similarity has to be strict on $gT$ and $sintRel$ and on how much is the weight of terminal and pre-terminal nodes. In particular four changes are suggested:

• **Coarse Grain Grammatical Marking.** By activating this setting, $gT$ and $sintRel$ are marked only by their own first character: e.g. the label

$$\langle VP#VBNG/NP# < li > \rangle$$
5.4. Extending the SPTK with compositional information: the Compositional SPTK

Algorithm 2 $\sigma_{T}(n_x, n_y, lw)$
Compositional estimation of the lexical contribution to semantic tree kernel

\[
\sigma_{T} \leftarrow 0,
\]

if
\[
\begin{align*}
&n_x = \langle lex_x::pos \rangle \quad \text{and} \\
&n_y = \langle lex_y::pos \rangle
\end{align*}
\]

\[
\sigma_{T} \leftarrow lw \cdot \sigma_{LEX}(n_1, n_2)
\]

end if

if
\[
\begin{align*}
&n_x = \text{pos} \quad \text{and} \\
&n_x = n_y
\end{align*}
\]

\[
\sigma_{T} \leftarrow 1
\]

end if

if
\[
\begin{align*}
&l_{x} = \langle gT, syntRel, \langle li_x \rangle \rangle \quad \text{and} \\
&\text{and} \\
&n_y = \langle gT, syntRel, \langle li_y \rangle \rangle
\end{align*}
\]

\[
\begin{align*}
&\text{if} \\
&\text{if} \\
&\text{if} \\
&\text{if}
\end{align*}
\]

\[
\begin{align*}
&l_{x} = \langle h_x::pos_h, m_x::pos_m \rangle \quad \text{and} \\
&l_{y} = \langle h_y::pos_h, m_y::pos_m \rangle \\
&\sigma_{T} \leftarrow \sigma_{COMP}((h_x, m_x), (h_y, m_y))
\end{align*}
\]

end if

end if

end if

end if

end if

end if

end if

\[
\text{end if}
\]

end if

if
\[
\begin{align*}
&l_{x} = \langle h_x::pos \rangle \quad \text{and} \\
&l_{y} = \langle h_y::pos \rangle \\
&\sigma_{T} \leftarrow \sigma_{COMP}((h_x), (h_y)) = \sigma_{LEX}(n_x, n_y)
\end{align*}
\]

end if

end if

return $\sigma_{T}$

becomes

\[
\langle V#V/N# < li > \rangle
\]

In this way a more shallow matching between $gT$ and $sintRel$ is activated in the if statement #3 in the Algorithm 2 for non pre-terminal nodes similarity. Differences between passive and active verbal phrases are avoided (i.e. $VBG$, verb gerund or present participle versus $VBN$, verb past participle). Notice
that the special prepositional phrase \( \langle \text{with}\,|\,PP, NN \ast \text{with}, < li > \rangle \) becomes \( \langle P, N, < li > \rangle \) in order to avoid also constraints triggered by the specific preposition involved.

- **Depth penalty.** Depth penalty is a score penalty inversely proportional to the root distance of the nodes that are compared in the *CSPTK* algorithm. Every time a similarity score between two nodes \( n_x \) and \( n_y \) has to be computed, node distances from the root, i.e. \( d_x \) and \( d_y \), is calculated and the final similarity score is multiplying by the depth weight thus calculated:

\[
\rho_{x,y} = \frac{1}{d_x} \cdot \frac{1}{d_y}
\]

The closer are the nodes and the larger is the distance of nodes from the root, the higher their similarity score, weighted by \( \rho \).

- **Smoothing on Grammatical Types.** This setting concerns similarity between non pre-terminal nodes. In the standard case, as depicted in the Algorithm 2, similarity is activated only iff \( gT \) and \( sintRel \) of the two nodes are identical. This setting introduces a smoothing factor in case that only the \( gT \)s of the nodes are the same. The smoothing factor is set to \( \gamma = 1/3 \).

- **Only non pre-terminal.** By activating this setting, similarity between terminal and pre-terminal nodes has not been taken into account, i.e. similarity on the *if* statements (1) and (2) on Algorithm 2 are set to 0.

Algorithm 3 shows how Algorithm 2 change in the case that all the setting variations introduced above are activated. Notice that as the coarse grain grammatical
5.4. Extending the SPTK with compositional information: the Compositional SPTK

Algorithm 3 $\sigma_T(n_x, n_y, lw)$

\[
\begin{align*}
\sigma_T &\leftarrow 0, \\
SF &\leftarrow 1/3, \\
\text{if} &\quad n_x \in T \text{ and } \\
&\quad n_y \in T \text{ then} \\
&\quad \text{return } \sigma_T \\
\text{end if} \\
\text{if} &\quad n_x \in PT \text{ and } \\
&\quad n_y \in PT \text{ then} \\
&\quad \text{return } \sigma_T \\
\text{end if} \\
&\quad n_x = \langle gT, syntRel_x, <li_x> \rangle \quad \text{and} \\
&\quad n_y = \langle gT, syntRel_y, <li_y> \rangle \text{ then} \\
&\quad \text{return } SF \\
\text{end if} \\
&\quad n_x = \langle gT, syntRel, <li_x> \rangle \quad \text{and} \\
&\quad n_y = \langle gT, syntRel, <li_y> \rangle \text{ then} \\
&\quad \text{if} \\
&\quad \text{if} \\
&\quad \quad li_x = \langle h_x::pos, m_x::pos_m > \quad \text{and} \\
&\quad \quad li_y = \langle h_y::pos, m_y::pos_m > \quad \text{then} \\
&\quad \quad \sigma_T \leftarrow \sigma_{COMP}(\langle h_x, m_x \rangle, \langle h_y, m_y \rangle) \cdot DP \\
&\quad \text{end if} \\
&\quad \text{if} \\
&\quad \quad li_x = \langle h_x::pos > \quad \text{and} \\
&\quad \quad li_y = \langle h_y::pos > \quad \text{then} \\
&\quad \quad \sigma_T \leftarrow \sigma_{COMP}(\langle h_x, m_y \rangle, \langle h_y, m_y \rangle) \cdot DP \\
&\quad \text{end if} \\
&\quad \text{if} \\
&\quad \quad li_x = \langle h_x::pos > \quad \text{and} \\
&\quad \quad li_y = \langle h_y::pos > \quad \text{then} \\
&\quad \quad \sigma_T \leftarrow \sigma_{COMP}(\langle h_x, m_y \rangle, \langle h_y, m_y \rangle) = \sigma_{LEX}(n_x, n_y) \cdot DP \\
&\quad \text{end if} \\
&\quad \text{return } \sigma_T \\
\end{align*}
\]

marking setting affect the node labeling, is not visible neither in the Algorithm 2 nor in the Algorithm 3.

109
5.5 Experimenting the CSPTK for Semantic Textual Similarity

In this section the CSPTK model is used in a Semantic Textual Similarity (STS) task already described in Section 4.5. The UNITOR system participating to both tasks of the *SEM 2013 shared task on STS is validated according to its extension including the CSPTK feature. The aim of this section is to measure the CSPTK similarity function in the STS task, comparing its performance with the other Kernel function, i.e. PTK and SPTK in an unsupervised scenario. Moreover the CSPTK feature will be used, together with other syntactic, semantic and lexical features, to train a SV regressor, and its contribution will be measured.

In the **Core STS task** given two sentences, $s_1$ and $s_2$, participants are asked to provide a score reflecting the corresponding text similarity. It is essentially the same task proposed in [Agirre et al. (2012)]. The semantic relatedness between two sentences is first modeled in an unsupervised fashion by several similarity functions, each describing the analogy between the two texts according to a specific semantic perspective. We aim at capturing separately syntactic and lexical equivalences between sentences and exploiting either topical relatedness or paradigmatic similarity between individual words. Such information is then combined in a supervised schema in a scoring function $y = f(x)$ over individual measures $(x_1, \ldots, x_n) \equiv x$ that is learned from labeled data through SV regression: $y$ is the gold similarity score (provided by human annotators), while $x$ is the vector of the different individual score, provided by the chosen similarity functions.

The proposed approach has been implemented in a system that aims at providing high applicability and robustness. This objective is pursued by adopting four similarity
5.5. Experimenting the CSPTK for Semantic Textual Similarity

measures designed to avoid the risk of over-fitting over each specific dataset. Moreover, the approach does not require any manually coded resource (e.g. WordNet), but mainly exploits distributional analysis of unlabeled corpora. The experiment also aims to validate the additional CSPTK similarity function introduced in Chapter 5.4.

In Section 5.5.1 the employed similarity functions are described and the application of SV regression is presented, while the experimental setup is introduced in 5.5.2. Unsupervised results of the adopted features over both training and test datasets of the SemEval 2013 - Task 6, are shown in Section 5.5.3. Finally Section 5.5.4 discusses results obtained by combining STs features over SV regressor.

5.5.1 Similarity functions

Each STS depends on a variety of linguistic aspects in data, e.g. syntactic or lexical information. While their supervised combination can be derived through SV regression, different unsupervised estimators of STS exist.

**Lexical Overlap.** A basic similarity function is first employed as the *Lexical Overlap* (LO) between sentences. Given the sets $W_a$ and $W_b$ of words occurring in two generic texts $t_a$ and $t_b$, it is estimated as the *Jaccard Similarity* between the sets, i.e.

$$ LO = \frac{|W_a \cap W_b|}{|W_a \cup W_b|} \tag{5.15} $$

In order to reduce data sparseness, lemmatization is applied and each word is enriched with its POS to avoid the confusion between words from different grammatical classes.

**Compositional Distributional Semantics.** Other similarity functions are obtained in order to account for the syntactic composition of the lexical information involved in
the sentences. Basic Lexical information is obtained by a co-occurrence Word Space that is built as described in \cite{Sahlgren2006,CrocePrevitali2010}. Every word $w_1, \ldots, w_m$ appearing in a sentence of length $m$ is then projected as a vector in such space. A sentence can be thus represented, neglecting its syntactic information, by applying a additive linear combination, i.e. the so-called \textbf{SUM} operator, described as

$$s = \sum_{i=1}^{m} w_i$$  \hspace{1cm} (5.16)

where $s$ is the vectorial surrogate of a sentence. The similarity function between two sentences is then the cosine similarity between their corresponding vectors.

A second function is obtained by applying a \textit{Syntactic Soft Cardinality} operator (SSC), in line with the approaches described in \cite{MitchellLapata2010b}, employed to account for semantic composition. In particular, the approach described in Chapter 4.5 has been applied.

\textbf{Convolution kernel-based similarity.} The similarity function is here the \textit{Smoothed Partial Tree Kernel} (SPTK) proposed in \cite{Croceetal2011c}. This convolution kernel described in Section 5.1.3 estimates the similarity between sentences, according to the syntactic and lexical information in both sentences. Syntactic representation of a sentence like “A man is riding a bicycle” is derived from the dependency parse tree.
It allows to define different tree structures over which the SPTK operates. A tree includes lexemes, where edges encode their dependencies, is generated: then, we add two leftmost children to each lexical node, encoding the grammatical function and the POS-Tag respectively: it is the so-called Lexical Centered Tree (LCT). An example for the sentence “A man is riding a bicycle” is shown in Figure 5.11. It provides a kernel function so that the SPTK similarity score ($\text{SPTK}_{\text{LCT}}$) is here obtained.

**Compositionally Smoothed Partial Tree Kernel.** Finally the similarity function here proposed is the *Compositional Smoothed Partial Tree Kernel* (CSPTK) introduced and discussed in Section 5.4, that estimates the similarity between sentences, according to the syntactic, lexical and compositional semantic informations. The structure on which the CSPTK operates is the compositionally enriched constituency parse tree introduced in Section 5.3 (e.g. Figure 5.5 refers to the sentence “A man is riding a bicycle”).

The similarity functions described above provide scores capturing different linguistic aspects and an effective way to combine such information is made available by Support Vector (SV) regression, described in [Smola and Schölkopf(2004)]. The idea is to learn a higher level model by weighting scores according to specific needs implicit in training data. Given similarity scores $x_i$ for the $i$-th sentence pair, the regressor learns a function $y_i = f(x_i)$, where $y_i$ is the score provided by human annotators. Moreover, since the linear combination of kernels is still a kernel, we can apply polynomial and RBF kernels [Shawe-Taylor and Cristianini(2004b)] to the regressor.

### 5.5.2 Experimental Setup

In all experiments, sentences are processed with the Stanford CoreNLP\footnote{http://nlp.stanford.edu/software/corenlp.shtml} for Part-of-speech tagging, lemmatization, and dependency and compositionally enriched parsing...
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

In order to estimate the basic lexical similarity function employed in the SUM, SSC and SPTK operators, a co-occurrence Word Space is acquired through the distributional analysis of the UkWaC corpus [Baroni et al. (2009)], a Web document collection made of about 2 billion tokens. The same setting of [Croce et al. (2012b)] has been adopted for the space acquisition. First, all words occurring more than 100 times (i.e. the targets) are represented through vectors. The original space dimensions are generated from the set of the 20,000 most frequent words (i.e. features) in the UkWaC corpus. One dimension describes the Pointwise Mutual Information score between one feature as it occurs on a left or right window of 3 tokens around a target. Left contexts of targets are treated differently from the right ones, in order to capture asymmetric syntactic behaviors (e.g., useful for verbs): 40,000 dimensional vectors are thus derived for each target. The Singular Value Decomposition is applied and the space dimensionality is reduced to \( k = 250 \). The same setup described in [Croce et al. (2012a)] is applied to estimate the SSC function, referring the Equation 4.18. The similarity between pairs of syntactically restricted word compound is evaluated through a Symmetric model: it selects the best 200 dimensions of the space, selected by maximizing the component-wise product of each compound as in Section 4.2 and combines the similarity scores measured in each couple subspace with the product function. The similarity score in each subspace is obtained by summing the cosine similarity of the corresponding projected words. The “soft cardinality” is estimated with the parameter \( p = 2 \).

The estimation of the semantically Smoothed Partial Tree Kernel (SPTK) is made available by an extended version of SVM-LightTK software[^4] (Moschitti(2006b)) im-

[^4]: http://disi.unitn.it/moschitti/Tree-Kernel.htm
5.5. Experimenting the CSPTK for Semantic Textual Similarity

Implementing the smooth matching between tree nodes. Similarity between lexical nodes is estimated as the cosine similarity in the co-occurrence Word Space described above, as in [Croce et al. (2011c)]. Finally, SVM-LightTK is employed for the SV regression learning to combine specific similarity functions.

5.5.3 Evaluating STs features in an unsupervised scenario

A first result in order to measure the contribution of the Kernel-based operators is pursued in a unsupervised scenario.

In Tables 5.1 and 5.2 results of Pearson Correlations between the human scores and PTK, SPTK and CSPTK similarity functions are shown and employed over the Training and the Test sets of the *SEM 2013 shared task respectively. They consists of 9 datasets, partially already introduced in Section 4.5. For the SPTK, we selected the parameters $\lambda = 0.1$, $\mu = 0.1$ and $\text{lexical}_\text{weight} = 100$ that provided best results in [Croce et al. (2012b)]. Otherwise for CSPTK we selected $\lambda = 0.4$, $\mu = 0.4$ and $\text{lexical}_\text{weight} = 10$ and the coarse grain marking, depth penalty and smoothing on $gT$ filters. In the Training set, MSRvid includes 1,500 short text pairs, from the Microsoft Research Video Description Corpus (MSR-Video); MSRpar is made of 1,500 sentence pairs from Microsoft Research Paraphrase Corpus; SMTeuroparl contains about 1,200 sentences of WMT2008 development dataset, derived from the Europarl corpus and it is made of syntactically complex sentences; surprise.OnWn comprised 750 pairs of glosses from OntoNotes 4.0 [Hovy et al. (2006)] and WordNet 3.1 senses; surprise.SMTnews contains 399 pairs of news conversation sentence pairs from WMT. In the Test set Headlines compares 750 sentences of news headlines, while FNWN consists in 189 sentence pairs taken from FrameNet [Baker et al. (1998)].
and WordNet. Additional 561 and 750 sentences from OntoNotes and WordNet, i.e. OnWN, and from WMT, i.e. SMTnews, have been employed in the Test set respectively.

PTK and SPTK functions are both applied to the Lexical Centered Tree and to the Constituency Tree representations, labeled respectively with lct and ct. CSPTK model consist in: i) lexical mark-up as a form of lexical compositional caching that generates the input Compositionally enriched Constituency Tree representation (i.e. cct) as introduced and discussed in Section 5.4 and ii) the matching function among the subtrees.

<table>
<thead>
<tr>
<th>Models</th>
<th>MSRvid</th>
<th>MSRPar</th>
<th>Training Dataset</th>
<th>Test Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>surprise.OnWN</td>
<td>surprise.SMTNews</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PTK_{lct}</td>
<td>.62</td>
<td>.56</td>
<td>.49</td>
<td>.63</td>
</tr>
<tr>
<td>SPTK_{lct}</td>
<td>.54</td>
<td>.55</td>
<td>.54</td>
<td>.6</td>
</tr>
<tr>
<td>PTK_{ct}</td>
<td>.12</td>
<td>.26</td>
<td>.45</td>
<td>.49</td>
</tr>
<tr>
<td>SPTK_{ct}</td>
<td>.18</td>
<td>.28</td>
<td>.45</td>
<td>.55</td>
</tr>
<tr>
<td>CSPTK_{MLE}</td>
<td>.50</td>
<td>.26</td>
<td>.43</td>
<td>.29</td>
</tr>
<tr>
<td>CSPTK_{SupSub}</td>
<td>.65</td>
<td>.32</td>
<td>.5</td>
<td>.59</td>
</tr>
</tbody>
</table>

Table 5.1: Unsupervised results of Pearson correlation for Kernel-based features adopted in SEMEVAL Task 6 over the training dataset

<table>
<thead>
<tr>
<th>Models</th>
<th>FNWN</th>
<th>Headlines</th>
<th>OnWN</th>
<th>SMTNews</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PTK_{lct}</td>
<td>.37</td>
<td>.66</td>
<td>.48</td>
<td>.34</td>
</tr>
<tr>
<td>SPTK_{lct}</td>
<td>.32</td>
<td>.60</td>
<td>.51</td>
<td>.32</td>
</tr>
<tr>
<td>PTK_{ct}</td>
<td>.15</td>
<td>.40</td>
<td>.04</td>
<td>.28</td>
</tr>
<tr>
<td>SPTK_{ct}</td>
<td>.19</td>
<td>.49</td>
<td>.24</td>
<td>.31</td>
</tr>
<tr>
<td>CSPTK_{MLE}_{cct}</td>
<td>.20</td>
<td>.52</td>
<td>.35</td>
<td>.32</td>
</tr>
<tr>
<td>CSPTK_{SupSub}</td>
<td>.21</td>
<td>.52</td>
<td>.37</td>
<td>.33</td>
</tr>
</tbody>
</table>

Table 5.2: Unsupervised results of Pearson correlation for Kernel-based features adopted in SEMEVAL Task 6 over the test dataset
5.5. Experimenting the CSPTK for Semantic Textual Similarity

Rows 3 and 4 in Tables 5.1 and 5.2 show Pearson results of PTK_{ct} and SPTK_{ct} functions applied over a constituency tree, while the last two rows show the CSPTK_{cct} results over the compositionally enriched arboreal structure. Two DC models have been exploited in the recursive formulation of Algorithm 2: the Mitchell and Lapata additive operator (i.e. \textit{ML}) and the Support Subspace based (i.e. \textit{SupSub}).

The first two rows are referring to the employee of PTK and SPTK functions over a dependency-based tree, i.e. lct. This representation seems to influence results that are inline, e.g. MSRvid, surprise.SMTNews, or overperform, e.g. MSRPar, the CSPTK function. Anyhow notice that only in OnWN and in SMTEuroparl, the SPTK_{lct} performs better than the PTK_{lct}, while starting from a constituency structure CSPTK_{cct} over performs always the referring PTK and SPTK functions.

By comparing results of the second couple of rows, i.e. PTK_{ct} and SPTK_{ct}, it seems that the introduction of lexical semantics leads to a slight increase of Spearman Correlation from PTK to SPTK operators. Moreover, exploiting compositional semantics together with the CSPTK function, implies a performance boost strongly related to the DCS model employed. The additive function does not seem robust within the CSPTK, by performing always worse than the Support Subspace model and in some cases even worse in comparison with SPTK. On the other hand the boost between SPTK_{ct} and CSPTK\textsubscript{SupSub} \textsubscript{cct} seems steady and sometimes very remarkable, e.g. switching from \(0.18\) to \(0.65\) in MSRvid and from \(0.24\) to \(0.37\) in OnWN.

The above difference is mainly due to the increasing sensitivity of PTK, SPTK and CSPTK to the incrementally rich lexical information. This is especially evident in sentence pairs with very similar syntactic structure. For example in the MSRvid dataset, a sentence pair is given by \textit{The man are playing soccer} and \textit{A man is riding a motor-}
cycle, that are strictly syntactically correlated. In fact, PTK provide a similarity score of 0.647 between the two sentences as differences between tree structures is confined to the leaves. By scoring 0.461, SPTK introduces an improvement as the distributional similarity (function $\sigma$ in Eq. 5.5) that acts as a smoothing factor between leaves better discriminates uncorrelated words, like motorcycle and soccer. However, ambiguous words such as verbs ride and play are still promoting a similarity that is locally misleading. Notice that both PTK and SPTK receive a strong contribution in the recursive computation of the kernels by the left branching of the tree, as the subject is the same, i.e. man. Compositional information about direct objects (soccer vs. motorcycle) is better propagated by the CSPTK$^{SupSub}$ operator. Its final scores for the pair is 0.36, as semantic differences between the sentences are emphasized. Even if grammatical types strongly contribute to the final score (as in PTK or SPTK), now the DCS computation over intermediate nodes (i.e. the VPs (ride::v, motorcycle::n) and (play::v, soccer::n)) is faced with less ambiguity with corresponding lower scores. This better reflects the semantics of the syntax-semantic interaction. It seems that a context free use of word vectors (such as in simple additive models) is a too weak modeling of compositionality.

In Tables 5.3 and 5.4 Pearson results of the remaining lexical features are shown. Notice how LO and SUM, functions strongly based on lexicon, perform best in almost all the dataset. Thus the validity and robustness of CSPTK is demonstrated over the tree Kernel-based function, but maybe in this specific task the role of the lexicon is crucial while the syntax seems not to be so essential, especially for dataset with sophisticated and long sentences, e.g. MSRPar.
5.5. Experimenting the CSPTK for Semantic Textual Similarity

<table>
<thead>
<tr>
<th>Models</th>
<th>MSR Vid</th>
<th>MSR Par</th>
<th>SMTEuroparl</th>
<th>surprise.OnWN</th>
<th>surprise.SMTNews</th>
</tr>
</thead>
<tbody>
<tr>
<td>LO</td>
<td>.75</td>
<td>.59</td>
<td>.46</td>
<td>.65</td>
<td>.40</td>
</tr>
<tr>
<td>SUM</td>
<td>.73</td>
<td>.25</td>
<td>.57</td>
<td>.68</td>
<td>.49</td>
</tr>
<tr>
<td>SSC</td>
<td>.39</td>
<td>.59</td>
<td>.52</td>
<td>.62</td>
<td>.40</td>
</tr>
</tbody>
</table>

Table 5.3: Unsupervised results of Pearson correlation for lexical features adopted in SEMEVAL Task 6 over the training dataset

<table>
<thead>
<tr>
<th>Models</th>
<th>FNWN</th>
<th>Headlines</th>
<th>OnWN</th>
<th>SMTNews</th>
</tr>
</thead>
<tbody>
<tr>
<td>LO</td>
<td>.36</td>
<td>.68</td>
<td>.5</td>
<td>.33</td>
</tr>
<tr>
<td>SUM</td>
<td>.40</td>
<td>.56</td>
<td>.64</td>
<td>.26</td>
</tr>
<tr>
<td>SSC</td>
<td>.38</td>
<td>.59</td>
<td>.33</td>
<td>.33</td>
</tr>
</tbody>
</table>

Table 5.4: Unsupervised results of Pearson correlation for lexical features adopted in SEMEVAL Task 6 over the test dataset

5.5.4 Combining STSs with SV Regression

The similarity functions described above provide scores capturing different linguistic aspects and an effective way to combine such information is made available by Support Vector (SV) regression, described in [Smola and Schölkopf(2004)]. The idea is to learn a higher level model by weighting scores according to specific needs implicit in training data. Given similarity scores \( x_i \) for the \( i \)-th sentence pair, the regressor learns a function \( y_i = f(x_i) \), where \( y_i \) is the score provided by human annotators. Moreover, since the combination of kernel is still a kernel, we can apply polynomial and RBF kernels [Shawe-Taylor and Cristianini(2004b)] to the regressor.

In the Core STS task, text similarity score is measured by the regressor: Each sentence pair from all datasets is scored according to 14 rates derived from each similarity function described in Section 5.5.3 derived from the different functions introduced in
Section 5.5.1

Five scores are derived by applying the LO operator over lemmatized words extracted by the sentences according to the filters ALL_v, ALL_n, ALL_j, ALL_r and ALL_nu,jr, that take into account only verbs, nouns, adjective, adverbs and all the previous respectively. A second set of 5 features is derived by the application of the SUM operator over the same selection of words. The SPTK is then applied to estimate the similarity between the LCT structures derived from the dependency parse trees of sentences. Then, the SPTK is applied to derive an additional score without considering any specific similarity functions between lexical nodes; in this setting, the SPTK can be considered as a traditional Partial Tree Kernel [Moschitti(2006b)], in order to capture a more strict syntactical similarity between texts. Another score is generated by applying the SSC operator, whilst the last score relies on CSPTK model introduced in Chapter 5.4. Thus, each sentence pair from all datasets is modeled according to a 14 dimensional feature space.

We avoid to determine the best combination of similarity functions, demanding this selection to the regressor itself. In [Croce et al.(2012b)] this strategy allowed to achieve the same results of the configuration where best features are manually selected.

The main difference between each run is the dataset employed in the training phase and the employed kernel within the regressor. Without any specific information about the test datasets, a strategy to prevent the regressor to over-fit training material has been applied. We decided to use a training dataset that achieved the best results over datasets radically different from the training material in the STS challenge of Semeval 2012. In particular, for the FNWN and OnWN datasets, we arbitrarily selected the training material achieving best results over the 2012 surprise.OnWN; for the headlines and SMT
5.5. Experimenting the CSPTK for Semantic Textual Similarity

datasets we maximized performance training over surprise.SMTnews. In Run\(_1\) the
SVM regressor is trained using dataset combinations providing best results according
to the above criteria: MSRpar, MSRvid, SMTeuroparl and surprise.OnWN are em-
ployed against FNWN and OnWN; MSRpar, SMTeuroparl and surprise.SMTnews are
employed against headline and SMT. A linear kernel is applied when training the re-
gressor. In Run\(_2\), differently from the previous one, the SVM regressor is trained using
all examples from the training datasets. A linear kernel is applied when training the re-
gressor. Finally, in Run\(_3\) the same training dataset selection schema of Run\(_1\) is applied
and a gaussian kernel is employed in the regressor. In Run\(_1^*\) we thus optimized the sys-
tem by manually selecting the training material that does provides best performance
on the test dataset: MSRvid, SMTeuroparl and surprise.OnWN are employed against
OnWN; surprise.OnWN against FNWN, SMTeuroparl against headlines; SMTeuroparl
and surprise.SMTnews against SMT.

General outcome for Run\(_1\), Run\(_2\), Run\(_3\) and Run\(_1^*\) in term of Pearson Correlation
are shown in Table 5.5. For these runs the complexity of the representation has been
reduced to a five dimensional feature space: LO and SUM without any specific filter,
SPTK\(_{lct}\), PTK\(_{lct}\) and SSC. Run\(_1^*\) shows the best performance and this system reach
the 19\(^{th}\) position in the STS Task of *SEM 2013. By starting from Run\(_1^*\), CSPTK
similarity function has been employed as a new feature for the SV-regressor.

In Table 5.6 features are added and removed from the optimized Run\(_1^*\) setting. In
particular in the first row, results of the regressor with the addition of the CSPTK\(_{cct}\)
feature are shown. In the second line SPTK\(_{lct}\) is neglected, whilst the last row shows
results using CSPTK\(_{cct}\) feature but not the SPTK\(_{lct}\). Thus, in Table 5.6 by starting
from the best system, i.e. Run\(_1^*\), CSPTK\(_{cct}\) feature has been introduced with or without
the contribution of the \( \text{SPTK}_{lc} \) feature. There is not a significantly changing in results between the three different settings. It seems that syntax (i.e. the arboreal representation), lexical semantic (i.e. the smoothing factor on the leaves), and compositional semantic (i.e. the DCS model applied on the non pre-terminal nodes of the compositionally enriched constituency tree) introduced by the \( \text{SPTK}_{lc} \) and \( \text{CSPTK}_{cct} \) does not affect significantly the system each other. Anyhow with respect to Run\(^*_1\), the \( \text{CSPTK}_{cct} \) feature introduces an improvement on OnWN and SMTNews datasets and it is in line with Headlines results, whilst best results on FNWN are achieved by neglecting both \( \text{CSPTK}_{cct} \) and \( \text{SPTK}_{lc} \) features.

<table>
<thead>
<tr>
<th></th>
<th>Headlines</th>
<th>OnWN</th>
<th>FNWN</th>
<th>SMTNews</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run(_1)</td>
<td>.63</td>
<td>.57</td>
<td>.35</td>
<td>.32</td>
</tr>
<tr>
<td>Run(_2)</td>
<td>.65</td>
<td>.56</td>
<td>.35</td>
<td>.31</td>
</tr>
<tr>
<td>Run(_3)</td>
<td>.60</td>
<td>.54</td>
<td>.32</td>
<td>.31</td>
</tr>
<tr>
<td>Run(_1^*)</td>
<td>.67</td>
<td>.63</td>
<td>.45</td>
<td>.34</td>
</tr>
</tbody>
</table>

Table 5.5: Results over the Core STS task

<table>
<thead>
<tr>
<th></th>
<th>Headlines</th>
<th>OnWN</th>
<th>FNWN</th>
<th>SMTNews</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run(<em>1^*)+CSPTK(</em>{cct})</td>
<td>.67</td>
<td>.64</td>
<td>.44</td>
<td>.34</td>
</tr>
<tr>
<td>Run(<em>1^*)-SPTK(</em>{lc})</td>
<td>.67</td>
<td>.64</td>
<td>.46</td>
<td>.34</td>
</tr>
<tr>
<td>Run(<em>1^*)-SPTK(</em>{lc})+CSPTK(_{cct})</td>
<td>.67</td>
<td>.65</td>
<td>.42</td>
<td>.35</td>
</tr>
</tbody>
</table>

Table 5.6: Results over the Core STS task

### 5.6 CSPTK for the Question Classification Task

The typical architecture of a QA system foresees three main phases: question processing, document retrieval and answer extraction [Kwok et al. (2001)]. Question processing is usually centered around the so called Question Classification task. It maps a
question into one of \( k \) predefined answer classes, thus posing constraints on the search space of possible answers. Typical class examples characterize the focus of the question and separate questions regarding persons or organizations (e.g., Who killed JFK?) from questions asking for definitions (e.g., What is the light?) or modalities (e.g., How fast does boiling water cool?)

For these experiments, the coarse grain question taxonomy defined in [Li and Roth(2002)] has been employed. Figure 5.7 shows question classes, organized in two levels: 6 coarse-grained classes (like ENTITY or HUMAN) and 50 fine-grained sub-classes (e.g. Plant, Food as subclasses of ENTITY). In [Li and Roth(2002)], an annotated corpus of questions labeled according to these classes has been also defined: it is composed by a training set of 5,452 questions and a test set of 500 questions. It represents the most used benchmark dataset for this task.\(^5\) Examples of the

<table>
<thead>
<tr>
<th>ABBR</th>
<th>description, expansion</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESC</td>
<td>definition, description, manner, reason</td>
</tr>
<tr>
<td>ENTY</td>
<td>animal, body, color, creation, currency, disease, medical, event, food, instrument, language, letter, other, plant, product, religion, sport, substance, symbol, technique, term, vehicle, word</td>
</tr>
<tr>
<td>HUM</td>
<td>description, group, individual, title</td>
</tr>
<tr>
<td>LOC</td>
<td>city, country, mountain, other, state</td>
</tr>
<tr>
<td>NUM</td>
<td>code, count, date, distance, money, order, other, percent, period, speed, temperature, size, weight</td>
</tr>
</tbody>
</table>

Table 5.7: The coarse and fine-grained question classes

\(^5\)http://cogcomp.cs.illinois.edu/Data/QA/QC/
respectively, and they provide more details on the type of expected numeric values in the answer. Most accurate QC systems apply supervised machine learning techniques, e.g., SVMs [Zhang and Lee(2003)] [Moschitti et al.(2007)] [Tomás and Giuliano(2009)] or the SNoW model [Li and Roth(2002)], where questions are encoded using a variety of lexical, syntactic and semantic features.

This task is very interesting to apply SPTK and CSPTK Kernel functions, as it has been shown in [Li and Roth(2002)] that the questions’ syntactic structure contributes remarkably to the classification accuracy. Some of the best models for Question Classification in fact exploit the benefits of grammatical information through Tree Kernels, as in [Zhang and Lee(2003)] [Moschitti et al.(2007)] [Moschitti et al.(2011)]. At the same time, the lexical information is crucial to discriminate examples showing the same syntactic patterns, e.g. “What is the weight . . .” (i.e. a weight) and “What is the length . . .” (i.e. a distance). In this cases, a kernel-based learning algorithm needs to acquire a proper lexical generalization related, for example, to the notion of distance to correctly classify questions, such as “What is the diameter . . .”, “What is the width . . .” or “What is the distance . . .”. A traditional Tree Kernel only provides a hard matching between lexical nodes and all those words must be in the resulting models. In real world scenarios, it may not be sufficient as questions are short sentences exposed to data sparseness and the training examples are in general not representative of every possible word use. In [Tomás and Giuliano(2009)], a learning approach with composite kernels is applied to incorporate semantic information and to extend a bag-of-words question representation, i.e. without an explicit representation of syntactic information. In particular, the authors employ latent semantic kernels [Cristianini et al.(2002)] to obtain a generalized similarity function between questions from Wikipedia, that im-
proved previous work.

5.6.1 General experimental setup

Thanks to structural kernel similarity, a QC task can be easily modeled by representing questions, i.e., the classification objects, with their parse trees. For generating constituency trees, the Charniak parser [Charniak(2000)] is used. In this experiment, sentences are processed with the Stanford CoreNLP

resulting trees are used to mark compositional information discussed in Section 5.3. As a smoothing factor, a coarse grain grammatical marking is here adopted, thus $gT$ and $rel_{h,m}$ are marked only by their own first character. In order to estimate the lexical similarity function exploited by SPTK and CSPTK kernels operators, a co-occurrence Word Space is acquired through the distributional analysis of the UkWaC corpus [Baroni et al. (2009)]. First, all words occurring more than 100 times (i.e. the targets) are represented through vectors. The original space dimensions are generated from the set of the 20,000 most frequent words (i.e. features) in the UkWaC corpus. One dimension describes the Pointwise Mutual Information score between one feature as it occurs on a left or right window of 3 tokens around a target. Left contexts of targets are treated differently from the right ones, in order to capture asymmetric syntactic behaviors (e.g., useful for verbs): 40,000 dimensional vectors are thus derived for each target. The Singular Value Decomposition is applied and the space dimensionality is reduced to $k = 250$. Similarity between lexical nodes is estimated as the cosine similarity in the co-occurrence Word Space, as in [Croce et al. (2011c)].

In this test, questions are represented by simple constituency tree, i.e. $ct$, and by their compositionally enriched counterparts, i.e. $cct$. The first representation is used
Chapter 5. Distributional Compositional Semantics in Tree Kernel Based Structured Learning

over PTK and SPTK functions, the latter over the CSPTK with the Support Subspace variant.

The aim of the experiment is to analyze the role of lexical similarity embedded in the compositionally enriched constituency trees over the CSPTK operator, by using the Support Subspace DCS model. The similarity function $\sigma_\tau(n_1, n_2)$ between tree nodes is here employed according to the Algorithm 3.

In this Question Classification (QC) task, a binary classifier for each class is defined: questions belonging to a class are considered positive example for the corresponding classifier, while the remaining instances are all negative ones. It means that 6 classifiers are employed for the coarse-grained settings. The resulting classifier is defined according to a multi-classification problem, which we model through the One-Vs-All scheme (OVA) [Rifkin and Klautau (2004)], by selecting the category associated with the maximum SVM margin, that reflects the higher classification confidence. For learning our models we employed LIBSVM after computing the entire Gram Matrix. The quality of such classification is measured in terms of the accuracy score, i.e. the percentage of test questions correctly classified.

5.6.2 Results

The outcome of the different kernels using (i) PTK and SPTK applied to $ct$ and (ii) CSPTK with the Support Subspace variant applied to $cct$ for the coarse-grained QC is reported in Table 5.8.

Results over all the test set show the benefit of the introduction of lexical semantics in kernel operators, i.e. SPTK and CSPTK. Notice that a large number of questions has a really simple syntactic structure: as a consequence the interrogative form of the

7http://www.csie.ntu.edu.tw/cjlin/libsvm/
5.6. CSPTK for the Question Classification Task

<table>
<thead>
<tr>
<th></th>
<th>Entire Test Set</th>
<th>RER</th>
<th>Subset of Test Set</th>
<th>RER</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTK&lt;sub&gt;ct&lt;/sub&gt;</td>
<td>.90</td>
<td>-</td>
<td>.79</td>
<td>-</td>
</tr>
<tr>
<td>SPTK&lt;sub&gt;ct&lt;/sub&gt;</td>
<td>.92</td>
<td>20%</td>
<td>.81</td>
<td>9.5%</td>
</tr>
<tr>
<td>CSPTK&lt;sub&gt;ct&lt;/sub&gt;</td>
<td>.92</td>
<td>20%</td>
<td>.86</td>
<td>33%</td>
</tr>
</tbody>
</table>

Table 5.8: Results in the Question Classification task

A sentence is very simple and very few compositional phenomena are observed. It is understandable how compositional semantic phenomena in syntactically simple questions as *What is an atom?* or *What is epilepsy?* are not exploited. Therefore, it is reasonable to think that SPTK and CSPTK results are here inline due to the fact that in the majority of the questions syntactic structure and lexical semantic information are enough for the task. The compositionally marked *cct* tree is here of no advantage and compositional similarity does not seem to add information. This is clear because the enriched tree in a sentence like *What is epilepsy?* does not carry any additional compositional information over non pre-terminal nodes. In order to consider the impact of the CSPTK operator it is necessary to deal with more complex questions like *What imaginary line is halfway between the North and South Poles?* or *What is the average speed of the horses at the Kentucky Derby?* whose syntactic trees are more structured. Thus, it could be interesting to test the kernel operators over a subset of 84 questions with more than 8 tokens, where the *cct* representation could exploit valid compositional pairs and the CSPTK function could correctly apply a DCS model between compounds.

The fourth column of Table 5.8 shows the Accuracy measures over this subset. Here the compositionality modeling of compounds seem crucial for the classification and the performance differences are more marked: accuracy measures of CSPTK overperform the other kernel operators. Notice how the Reduced Error Rate (i.e. RER) could be
directly attributable to the ability of the CSPTK to exploit the compositional semantics and the explicit compound representation in \( cct \), since the higher is the complexity of the syntactic structures of questions, the larger is the impact on the RER with respect to the other Kernel operators.
The problem of how to manage and model semantic compositionality is the main aim of this Thesis. We have seen that distributional analysis of lexicon has been used as a surrogate of its semantic representation, following the philosophy of [Wittgenstein(1953), Harris(1954), Firth(1961)] largely cited in Chapter 2. In Chapter 3, distributional semantic models [Schütze(1998a)] for lexical representation, that rely on distributional hypothesis [Landauer and Dumais(1997)] have been introduced. It has been stated how Topical and Word-Space models reflect syntagmatic and paradigmatic relations between lexicon entities. These models lack for a representation of multiple words expressions (i.e. phrases or sentences), since they simple model words in isolation. Semantic space models, such as LSA, represent lexical information in metric spaces where individual words are represented according to the distributional analysis of their co-occurrences over a large corpus; in order to govern the recursive interpretation of sentences or phrases, a compositional semantics mechanism acting over individual vectors is needed.

Thus, in order to express a computationally representation of the meaning of a complex structure, a Distributional Compositional Semantic (DCS) Model is presented in Chapter 4, where a family of brand new Distributional Compositional Models (DCM)
has been defined. Follow the Frege’s principle in [Frege(1960)], this Thesis focuses the attention on how to combine distributional lexical representations into a structured representation of the meaning of phrases or sentences. We have seen how state-of-the-art distributional methods that try to account for compositionality are still controversial, whereas general algebraic operators over lexical vectors [Mitchell and Lapata(2010a)] or regression-based models [Grefenstette and Sadrzadeh(2011), Guevara(2010), Baroni and Zamparelli(2010)] have been proposed. In this Thesis a novel distributional model for semantic composition is proposed, by focusing on the geometry of latent semantic spaces.

First of all in Section 4.2 the semantics of syntactic bigrams is modeled in lexically-driven subspaces. The assumption of the model proposed is that, in a composition, words influences each other interpretation. This influence is expressed via the operation of vector projection into a so called Support Subspace, i.e. a subset of the original features. A subspace local to a bigram phrase can be found such that only the features specific to its meaning are selected. This emphasize the role of common features that constraint in “parallel” the interpretation of the involved lexical meanings and better capture phrase-specific aspects. The proposed projection-based method of DCS has been evaluated in Section 4.3 over a well known dataset and task introduced in [Mitchell and Lapata(2010a)], in which grammatical bigram similarity has to be rated. Compositional similarity scores are correspondingly derived by selecting the most important components for a specific word pair in a relation, by projecting bigrams in the Support Subspace. Thus, bigrams meanings are captured locally to the relevant specific context evoked by the underlying lexical pair. The model seems very effective
for the syntactic structures of VO, NN and AdjN and achieves the same results than the average human interannotator agreement, by outperforming most previous results.

In a second evaluation stage, the generalization capability of the employed operators has been investigated and the robustness of Support Subspaces based model has been confirmed in a cross-linguistic scenario, i.e. in the English and Italian Language. It’s fair to expect that grammatical bigrams belonging to the same semantic classes must be related each other and the DCS model might to hold them together in clusters. This kind of generalization capability is crucial to use this class of operators in a QA scenario by enabling to rank results according to the complex representations of the question. In Section 4.4 a manually developed dataset consisting in a multilingual (i.e. English and Italian) VO word pairs divided into 3 different semantic classes is presented and used for the second experiment. This evaluation aims to measure how the proposed compositional operators group together semantically related word pairs, i.e. those belonging to the same class, and separate the unrelated pairs. The results suggest that the DCS model based on Support Subspace seems to reduce clusters’ ambiguity, since it is always able to increase the gap in the average similarity between the infra and extra-class pairs.

A further task is investigated in Section 4.5. As a measure of semantic similarity to support complex textual inferences, our DCS model is used in the Semantic Textual Similarity task presented in [Agirre et al.(2012)]. An SV regressor is trained combining different similarity functions, including the DCS Support Subspace metric, in order to learn a function to score sentences. This system participated to the SemEval chal-
Chapter 6. Conclusions

challenge, i.e. task 6, ranking around the 12 and 13 system positions (out of 89 systems) and providing a highly portable method as it does not depend on any manually built resource (e.g. WordNet) nor controlled, e.g. aligned, corpus.

The final step of this Thesis has been to scale towards more complex phenomena, by using the DCS model to account for the meaning of an entire sentence. Kernel functions are here pursued as a metric within a space of complex linguistic structures. DCS model introduced in Section 4.2 is applied to linguistic structures dominated by the non-terminal nodes of a grammar, called Compositionally enriched Constituency parse tree (see Section 5.2). In this way the lexical and grammatical descriptions have been enriched by spreading compositional lexical information of the leaves all over the tree nodes, in form of head-modifier lexical pairs.

Starting from arboreal Kernel functions [Collins and Duffy (2001)] and their smoothed extension proposed in [Croce et al. (2011c)], the idea pursued here is to spread semantic information all over parse trees. To achieve this, a mark-up technique for parse trees is presented in Section 5.3 and such that arboreal representation is used in a new kernel metric called Compositional Smoothed Partial Tree Kernel (CSPTK). In Section 5.4 this compositional metric, that is a valid kernel to account similarity between sentences, has been investigated. First empirical results of the CSPTK in a STS task has been presented in Section 5.5 demonstrating the robustness and the generalization capability of the metric firstly in a unsupervised scenario, secondly using it to train a SV regressor. In Section 5.6 first empirical results of the CSPTK in QC task demonstrate the robustness and the generalization capability of the proposed kernel.

Future investigation is needed in the adoption of the same compositionality per-
spective over alternative syntactic representations (e.g. dependency graphs), where the
compositional representations of the head/modifier compound are even more explicit.
Furthermore, a comparative analysis of the DCS operator employed within the CSPTK
formulation against other compositional operators (e.g. dilation) is required. Further
experiments for assessing the methodology are also foreseen against other NLP-tasks,
e.g. verb classification or semantic role labeling.
In this appendix the formalism by which nodes, lexicals and trees are grammatically and lexically linked within the Compositionally enriched Smoothed Partial Tree Kernel (CSPTK) discussed in Section 5.2 is described.

Let’s take a sentence such as *I make room for my friends*: the resulting parse tree is reported in the following Figure A.1.

It is possible to outline three different types of node that play different roles in the CSPTK formulation:

- **Terminal nodes** are the leaves and they belong to the set $\mathcal{T}$. Starting from the set $\mathcal{V}$ of the lexical vocabulary, a terminal node is a tuple of a lexical entry (in particular a lemma) $l \in \mathcal{V}$ and its lexical category $\text{pos} \in \text{POS}$, i.e. the set of the all possible lexical category depicted in Table A. In Figure A.1, friend::n is a terminal node that reflect the lemma friend and the part-of-speech of type noun shortened with n.

- **Pre-terminal nodes** are directly linked to the leaves and belong to the set $\mathcal{PT}$. A pre-terminal node is labeled with the $\text{pos}$ of the lemma of the respective terminal-node. For example the pre-terminal node directly linked to the leaf labeled friend:n is marked with the lexical category NNS, i.e. the $\text{pos}$ of the
Figure A.1: Charniak constituency tree of the sentences “I make room for my friends”

lemma friend as depicted in Figure A.1.

• **Non pre-terminal nodes** are the remaining nodes. These nodes have a Grammatical Type $\mathcal{GT}$, a Syntactic Relation $\mathcal{SR}$ and a lexical information. Grammatical Types, also called (sub-)tree types, are complex lexical category: while POS tags $\mathcal{POS}$ are referred to lemmas, elements in $\mathcal{GT}$ are referring to subtrees. The complete list of Grammatical Types is shown in Table A.

Remember that the mark-up rules can start from i) two pre-terminal nodes, ii) two non-pre terminal nodes, or iii) one pre-terminal and one non-preterminal nodes. Let’s name these node pair $(p_1, p_2)$, it follows that

1. $(l_1 :: pos_1, l_2 :: pos_2) \leftarrow (p_1, p_2)
   \text{iff } p_1, p_2 \in \mathcal{PT}$

2. $(l_1 :: pos_1, < gT, syntRel, lexInfo >) \leftarrow (p_1, p_2)
   \text{iff } p_1, p_2 \in \mathcal{NP}\mathcal{T}$
<table>
<thead>
<tr>
<th>POS tag</th>
<th>Pre-terminal Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>Coordinating conjunction</td>
<td>Maybe you and/CC I could partner up</td>
</tr>
<tr>
<td>CD</td>
<td>Cardinal number</td>
<td>I’m 50/CD</td>
</tr>
<tr>
<td>DT</td>
<td>Determiner</td>
<td>This/DT is conjecture</td>
</tr>
<tr>
<td>EX</td>
<td>Existential there</td>
<td>There/EX’s got to be some other way</td>
</tr>
<tr>
<td>FW</td>
<td>Foreign word</td>
<td>Don Eladio esta/FW muerto/FW!</td>
</tr>
<tr>
<td>IN</td>
<td>Preposition or subordinating conjunction</td>
<td>What you said in/IN the desert, I get it</td>
</tr>
<tr>
<td>JJ</td>
<td>Adjective</td>
<td>Good/JJ job wearing the pants in the family!</td>
</tr>
<tr>
<td>JJR</td>
<td>Adjective, comparative</td>
<td>You need me more/JJR than I need you</td>
</tr>
<tr>
<td>JJS</td>
<td>Adjective, superlative</td>
<td>A higher purity means a greater/JJS yield</td>
</tr>
<tr>
<td>LS</td>
<td>List item marker</td>
<td>I/LS1-year old kid doing their killing for them</td>
</tr>
<tr>
<td>MD</td>
<td>Modal</td>
<td>Science is/MD a mystery</td>
</tr>
<tr>
<td>NN</td>
<td>Noun, singular or mass</td>
<td>Say my name/NN</td>
</tr>
<tr>
<td>NNS</td>
<td>Noun, plural</td>
<td>Even drug dealers/NNS need lawyers, right?</td>
</tr>
<tr>
<td>NNP</td>
<td>Proper noun, singular</td>
<td>I am not in danger, Skyler/NNP</td>
</tr>
<tr>
<td>NNPS</td>
<td>Proper noun, plural</td>
<td>Cartel/NNPS got Fring</td>
</tr>
<tr>
<td>PDT</td>
<td>Predeterminer</td>
<td>There is blame on both/PDT sides</td>
</tr>
<tr>
<td>POS</td>
<td>Possessive ending</td>
<td>The agent’s/POS name is Hank Schrader</td>
</tr>
<tr>
<td>PRP</td>
<td>Personal pronoun</td>
<td>I/PRP am the danger</td>
</tr>
<tr>
<td>PRP$</td>
<td>Possessive pronoun</td>
<td>The decision is not yours/PRP$ to make</td>
</tr>
<tr>
<td>RB</td>
<td>Adverb</td>
<td>I wanna talk to you about/RB Heisenberg</td>
</tr>
<tr>
<td>RBR</td>
<td>Adverb, comparative</td>
<td>I’m more/RBR of a humanities guy</td>
</tr>
<tr>
<td>RBS</td>
<td>Adverb, superlative</td>
<td>You’ve got the greatest/RBS meth cook</td>
</tr>
<tr>
<td>RP</td>
<td>Particle</td>
<td>He’s not/RP you</td>
</tr>
<tr>
<td>SYM</td>
<td>Symbol</td>
<td>To W.W./SYM My star, my perfect silence</td>
</tr>
<tr>
<td>TO</td>
<td>to</td>
<td>Listen to/TO me</td>
</tr>
<tr>
<td>UH</td>
<td>Interjection</td>
<td>Mr. White, you kicked its ass, yo/UH!</td>
</tr>
<tr>
<td>VB</td>
<td>Verb, base form</td>
<td>I won’t cook/VB meth anymore</td>
</tr>
<tr>
<td>VBD</td>
<td>Verb, past tense</td>
<td>So Hector Salamanca killed/VBD Fring</td>
</tr>
<tr>
<td>VBG</td>
<td>Verb, gerund or present participle</td>
<td>It’s all about accepting/VBG who you really are</td>
</tr>
<tr>
<td>VBN</td>
<td>Verb, past participle</td>
<td>There’s been/VBN a contamination</td>
</tr>
<tr>
<td>VBP</td>
<td>Verb, non-3rd person singular present</td>
<td>Take/VBP it out</td>
</tr>
<tr>
<td>VBZ</td>
<td>Verb, 3rd person singular present</td>
<td>Bolsa says/VBZ the DEA is off-limits</td>
</tr>
<tr>
<td>WDT</td>
<td>Wh-determiner</td>
<td>What/WDT’s going on?</td>
</tr>
<tr>
<td>WP</td>
<td>Wh-pronoun</td>
<td>Who/WP the hell are you?</td>
</tr>
<tr>
<td>WP$</td>
<td>Possessive wh-pronoun</td>
<td>Passed down? By whom/WP$?</td>
</tr>
<tr>
<td>WRB</td>
<td>Wh-adverb</td>
<td>How/WRB’d you find me?</td>
</tr>
</tbody>
</table>

Table A.1: Complete list of POS elements
<table>
<thead>
<tr>
<th>Constituency Type</th>
<th>Label</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adjective Phrase</td>
<td>ADJP</td>
<td>Only (the very best) of you</td>
</tr>
<tr>
<td>Adverb Phrase</td>
<td>ADVP</td>
<td>Shot in the face and died (instantly)</td>
</tr>
<tr>
<td>Conjunction Phrase</td>
<td>CONJP</td>
<td>Now thank me (and shake my hand)</td>
</tr>
<tr>
<td>Fragment</td>
<td>FRAG</td>
<td>(Not exactly)</td>
</tr>
<tr>
<td>Interjection</td>
<td>INTJ</td>
<td>(Oh yeah), I remember that little bitch!</td>
</tr>
<tr>
<td>List marker</td>
<td>LST</td>
<td>(-First -Second -Third)</td>
</tr>
<tr>
<td>Not a Constituent</td>
<td>NAC</td>
<td>(Five-time tour the France winner), is in Denver</td>
</tr>
<tr>
<td>Noun Phrase</td>
<td>NP</td>
<td>He was (a meth cook)</td>
</tr>
<tr>
<td>complex NPs</td>
<td>NX</td>
<td>The (smart dog) and the (rude cat)</td>
</tr>
<tr>
<td>Prepositional Phrase</td>
<td>PP</td>
<td>Fill your pockets and leave (in peace)</td>
</tr>
<tr>
<td>Parenthetical</td>
<td>PRN</td>
<td>Kind of (you know)</td>
</tr>
<tr>
<td>Particle</td>
<td>PRT</td>
<td>I figured it out</td>
</tr>
<tr>
<td>Quantifier Phrase</td>
<td>QP</td>
<td>(Two and a half months ago i went in Rome</td>
</tr>
<tr>
<td>Reduced Relative Clause</td>
<td>RRC</td>
<td>Paolo, (currently a students), is a rocker</td>
</tr>
<tr>
<td>Unlike Coordinated Phrase</td>
<td>UCF</td>
<td>I know (the answer and that you don’t know it)</td>
</tr>
<tr>
<td>Verbal Phrase</td>
<td>VP</td>
<td>You (will not kill) Walter White</td>
</tr>
<tr>
<td>Wh-adjective Phrase</td>
<td>WHADJP</td>
<td>(Which reminds me), I better get back to it</td>
</tr>
<tr>
<td>Wh-adverb Phrase</td>
<td>WHAVP</td>
<td>Jesse, (where are you?)</td>
</tr>
<tr>
<td>Wh-noun Phrase</td>
<td>WHNP</td>
<td>(Who the hell are you?)</td>
</tr>
<tr>
<td>Wh-prepositional Phrase</td>
<td>WHPP</td>
<td>(Whom did you send to the letter?)</td>
</tr>
<tr>
<td>Unknown</td>
<td>X</td>
<td>-</td>
</tr>
</tbody>
</table>

Table A.2: Complete list of $G T$ elements
List of Mark-up rules

<table>
<thead>
<tr>
<th>Rule</th>
<th>h = p₂, m = p₁</th>
<th>← (p₁, p₂)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN/JJ</td>
<td></td>
<td>(JJ, NN)</td>
</tr>
<tr>
<td>VBG/NN</td>
<td></td>
<td>(NN, VBG)</td>
</tr>
<tr>
<td>VBG/VBG</td>
<td></td>
<td>(VBG, VBG)</td>
</tr>
<tr>
<td>VBG/JJ</td>
<td></td>
<td>(JJ, VBG)</td>
</tr>
<tr>
<td>NN/NN</td>
<td></td>
<td>(NN, NN)</td>
</tr>
<tr>
<td>NN/JJ</td>
<td></td>
<td>(NN, JJ)</td>
</tr>
<tr>
<td>V/NN</td>
<td></td>
<td>(V, NN)</td>
</tr>
<tr>
<td>NN/NN</td>
<td></td>
<td>(CD, NN)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(NN, POS)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(DT, NN)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(P, NN)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(C, N)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(I, NN)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(T, NN)</td>
</tr>
<tr>
<td>NN/*</td>
<td></td>
<td>(N, C)</td>
</tr>
<tr>
<td>JJ/JJ</td>
<td></td>
<td>(JJ, JJ)</td>
</tr>
<tr>
<td>JJ/*</td>
<td></td>
<td>(JJ, I)</td>
</tr>
<tr>
<td>JJ/*</td>
<td></td>
<td>(I, JJ)</td>
</tr>
<tr>
<td>JJ/*</td>
<td></td>
<td>(JJ, R)</td>
</tr>
<tr>
<td>JJ/*</td>
<td></td>
<td>(D, JJ)</td>
</tr>
<tr>
<td>JJ/*</td>
<td></td>
<td>(JJ, JJ)</td>
</tr>
<tr>
<td>C/C</td>
<td></td>
<td>(C, C)</td>
</tr>
<tr>
<td>R/R</td>
<td></td>
<td>(R, R)</td>
</tr>
<tr>
<td>VBG/NP</td>
<td>h = slit.v, m = pepper.n</td>
<td>(AUX, VBG)</td>
</tr>
</tbody>
</table>

Table A.3: Complete list of Mark-Up rules

3. (<gT, syntRel, lexInfo>, l₂ :: pos₂) ← (p₁, p₂)

iff p₁ ∈ NPT and p₂ ∈ PT

It is possible to derive the following specific rules for each case depicted in Table A.


tional lexical semantics: Toward uniform representation paradigms for advanced

curse of dimensionality for local kernel machines. Technical report, Departement
de’Informatique et Recherche Operationnelle.

[Blackburn and Bos (2005a)] Blackburn, P. and Bos, J. (2005a). *Representation and
Inference for Natural Language. A First Course in Computational Semantics.* CSLI
Publications.

Inference for Natural Language. A First Course in Computational Semantics.* CSLI.

[Cancedda *et al.* (2003)] Cancedda, N., Gaussier, E., Goutte, C., and Renders, J. M.
1082.

cedings of NAACL’00.*

[Chierchia and McConnell-Ginet (2000)] Chierchia, G. and McConnell-Ginet, S.

*IRE Transactions on Information Theory, 2,* 113–124.


BIBLIOGRAPHY

to plato’s problem: The latent semantic analysis theory of acquisition, induction, and

ity reduction. Information science and statistics. Springer.

[Li and Roth(2002)] Li, X. and Roth, D. (2002). Learning question classifiers. In Pro-
ceedings of ACL ’02, COLING ’02, pages 1–7, Stroudsburg, PA, USA. Association
for Computational Linguistics.

Proceedings of COLING-ACL, Montreal, Canada.

In Proceedings of the 17th international conference on Computational linguistics
- Volume 2, COLING ’98, pages 768–774, Stroudsburg, PA, USA. Association for
Computational Linguistics.

[Lodhi et al.(2002)] Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., , and
Learning Research, pages 419–444.

associative priming in high-dimensional semantic space. In Proceedings of the 17th

of statistical natural language processing. MIT Press, Cambridge, MA, USA.


BIBLIOGRAPHY


