Polysemy in Compositional Distributional Semantics

Siva Reddy

Department of Computer Science
University of York

A dissertation submitted for the degree
MSc by Research

January 2012

Abstract

Research in distributional semantics has made good progress in capturing individual word meanings using contextual frequencies obtained from a large corpus. While vocabulary of a language is limited, its generative power for combinatorial expressions is nonrestrictive, and so lexical semantic methods cannot be applied directly to phrasal or sentential semantics irrespective of the corpus size. Any distributional model that aims to describe a language adequately needs to address the issue of compositionality.

Very recently, a new field called Compositional Distributional Semantics (CDS) emerged, stretching the boundaries of distributional semantics from word level meaning representation to higher levels such phrasal and sentential semantic representations. CDS models deal with the task of composing the meaning of a phrase/sentence from the distributional meaning of its constituents.

Polysemy of words have been a major focus in distributional semantics. The challenges posed at lexical level make a transition to phrasal and higher levels, making polysemy a major threat to CDS models. In this thesis, we aim to build better CDS models by performing sense disambiguation. We test our hypothesis, *sense disambiguation benefits compositional models*, on different compositionality based evaluation tasks.

The evaluation of compositional models is an uncertain topic. Since we humans do not know the way we compose semantics of expressions, it is hard to prepare datasets for evaluation, thus making the evaluation of CDS models a challenging topic. In this thesis, we focus on evaluation methods for compositional models and develop a dataset with a novel annotation scheme.

Acknowledgements

I sincerely thank my supervisor Suresh Manandhar for pointing me towards this thesis problem, sharing many brainstorming discussions, giving enormous freedom and mainly for understanding and supporting my future goals. He is also an expert in badminton in which I didn't manage to beat him yet.

This thesis wouldn't have been possible without the unconditional support from my favourite collaborator Diana McCarthy. Her encouragement in every aspect of my life helped me face many challenges both professionally and personally. I dedicate this thesis to her. She is one among my role models.

I owe a lot to Matt Naylor, Shailesh Pandey, Suraj Pandey and Sonia Xavier for filling colours in my life at York. My memories remind me Matt's home grown strawberries and Irish songs played on his whistle, Sonia's tasty daal and her beautiful voice, Suraj's happy beers and tasty pork, and Shailesh's guitar and culinary skills. I will miss those good times.

I am grateful to Adam Kilgarriff for having enormous faith in me and supporting me in many ways especially for making me a family member of Sketch Engine development team, Ioannis Klapaftis for many interesting discussions ranging from Greece Politics to advanced Machine Learning algorithms and also for his contribution towards my thesis, Michael Banks for his welcoming and helpful nature, Kleanthis Malialis for sharing many thoughts on PhD life.

Great thanks to my examiners Mirella Lapata and Daniel Kudenko for accepting to review my thesis and timely submission of reviews. Coincidentally Mirella is going to be my PhD supervisor at Edinburgh with which I am very excited about.

I would like to thank my cheerful colleagues for making my life easier in the department: Santa Basnet, Burcu Can, Shiromani Ghimire, Azniah Ismail, Ali Karami, Tasawer Khan, Ioannis Korkontzelos, Shuguang Li, Nelson Lin, Ankur Patel and Ahmad Shahid. Special regards to my friends who were there to cheer me on phone: Bharat Ram Ambati, Phani Chaitanya, Abhilash Inumella, Janga, John, Koneru, Satish Pitchikala, PS, Avinesh PVS, Srikanth, Ravikiran Vadlapudi, Raghavendra Vanama, Vamsi, Sandeep YV, YSP.

Most importantly I adore my family, Amma, Pinni, Nanna, Babai, Mama, Ammamma, Nirosha, Anusha for their invaluable love and support all through my ups and downs although far. Finally my greatest applauds for Spandana who followed my heart, giving up her job, and relocating to many different countries. She is the one who always stood by my side.

Contents

List of Figures

List of Tables

Declaration

I hereby declare that I composed this thesis entirely myself and it describes my own research.

Siva Reddy
University of York

Portions of this thesis are based on the following papers:

1. **Siva Reddy**, Ioannis P. Klapaftis, Diana McCarthy, Suresh Manandhar. Dynamic and Static Prototype Vectors for Semantic Composition. In *Proceedings of The 5th International Joint Conference on Natural Language Processing 2011 (IJCNLP 2011), Chiang Mai, Thailand*
   [Best Paper Award]

2. **Siva Reddy**, Diana McCarthy, Suresh Manandhar. An Empirical Study on Compositionality in Compound Nouns In *Proceedings of The 5th International Joint Conference on Natural Language Processing 2011 (IJCNLP 2011), Chiang Mai, Thailand*

3. **Siva Reddy**, Diana McCarthy, Suresh Manandhar, Spandana Gella. Exemplar-based Word-Space Model for Compositionality Detection: Shared task system description. In *Proceedings of DISCo-2011 in conjunction with ACL-2011*, 2011
   [Our system was ranked 1^{st} in two evaluation categories and 2^{nd} in two other]

Chapter 1

Introduction

1.1 Compositional Semantics

How do humans comprehend utterances in natural language? How do we sum up the meaning of each component in an utterance and arrive at the right meaning? Can machines imitate this? While humans are highly competent in understanding multi word units like phrases, it still remains a herculean task for machines. Recent progress in semantic technologies like search engines has had an immense effect on human life, yet these technologies are scratching the surface of human language at word level. The impact of semantic technologies at higher levels like phrases is far beyond the reach of existing systems. Potential applications include, but are not limited to: intelligent search engines, automatic answer grading, bio-medical applications, question answering, textual entailment, and summarization.

Research in lexical semantics has made good progress in capturing individual word meanings. However, the same methods cannot be applied directly to model the semantics of phrases due to data sparsity. While vocabulary of a language is limited, its generative power for combinatorial expressions is nonrestrictive, and so lexical semantic methods fail to model phrasal or sentential semantics in-spite of how much ever data we use. Any model which aims capture language should be generative. But how do we make use of existing research in lexical semantics and advance further to phrasal or sentential semantics? The answer comes from *Compositional Semantics*.

Compositional semantics involves the study of the meaning of an expression in relation with the meaning of its parts. The *Principle of Compositionality* [Pelletier, 1994, page. 313] states that the meaning of an expression is a *function* of, and only of, the meaning of its parts and the way in which the parts are combined. While humans are gifted with this function, formalizing its true mathematical structure will be a miracle, which is the goal of compositional models.

1.2 Challenges

Many factors such as the polysemy of constituent words, the role of syntactic structure, semantic preferences of constituents, idiomatic and metaphoric usages, play an important role in molding the meaning of an expression. Cracking the way in which humans process these components to arrive at a meaning will be a major breakthrough in computational linguistics. Below, we discuss some of the challenges in brief.

1.2.1 Polysemy of Constituent Words

Polysemy of words have been a major focus of lexical semantics. The challenges posed at lexical level make a transition to phrasal and higher levels, making polysemy a major threat to compositional semantics. While the efforts of lexical sense disambiguation methods have not seen real benefits [Navigli, 2009], the effect of polysemy on compositional semantics is yet to be studied. Take an example phrase *bank balance*. In the WordNet [Fellbaum, 1998], *bank* and *balance* have 10 and 12 senses respectively. But, only one sense of *bank* and one sense of *balance* are relevant in the phrase *bank balance*, and choosing a correct sense for each constituent is critical for a good compositional model.

The questions, *how do you make use of sense disambiguation? Is sense disambiguation really useful for compositional models?*, still remains unanswered and have to be explored.

1.2.2 Syntactic Structure

The semantic interpretation of an expression changes with a change in its syntactic structure. For example, the semantics of phrases formed by the combinations of *house* and *rent* differ. The phrase *house rent* means *the rent to be paid for a house* whereas the phrase *rent house* means *a house which is available on periodic rental basis*. Syntactic structure guides the information flow in arriving at the correct interpretation of an expression. A good compositional model should take syntactic structure into account.

1.2.3 Semantic Preferences

Experimental studies on human sentence processing reveal that humans not only use lexical and syntactic information but also semantic preferences when processing a sentence [Padó *et al.*, 2009]. For example, given an uncompleted sentence such as *"Among all the fruits, John likes to eat a/an ___"*, a human processing model starts expecting a *fruit* in the blank, lets say *orange*. Perhaps the reasons for choosing *orange* is because the preferences of other words in the sentence expects an edible fruit and the properties of *orange* such as *taste, juice, pulp* makes it edible. Given that humans use semantic preferences in sentence processing, it is necessary for any good compositional model to take this information into consideration. Semantic preferences capture subtle properties beyond syntactic relations. For example, the semantic preferences of *laser* in the phrases *laser light* and *laser treatment* are different though the syntactic relation is the same (*modifier*). In *laser light*, the meaning gets transformed into *a specific type of light*, and in *laser treatment* the meaning becomes *a treatment using laser*. In each phrase, the properties picked up due to semantic preferences of words are completely different. Semantic preferences of words help to choose relevant properties of words required for composition.

1.2.4 Idiomatic and Metaphoric usages

As the name compositional in compositional models indicate, compositional models are designed to build semantics compositionally from the meaning of their parts. But compositional interpretation may not be possible with

idiomatic and metaphoric usages which are known to be non-compositional. In idiomatic expressions, the semantic interpretation of an expression is beyond the superficial meaning of its constituents e.g. *he was born with a silver spoon*, here the meaning of *silver spoon* is not literally meant but idiomatically meant to be *rich*. Most idioms can only be interpreted by knowing the meaning of the idiom beforehand. Metaphors on the other can be understood if one has enough cultural background e.g. *Juliet is the sunshine in Romeo's life*, here it is meant *Juliet means a lot to Romeo*. It is uncertain if compositional models are expected to model the semantics of non-compositional expressions. However, a good compositional model should be able to distinguish compositional meaning from non-compositional meaning.

1.2.5 Other Challenges

Metonymy is a phenomenon in which a foreign word stands on behalf of a target word, the foreign word representing the semantics of the target word, e.g. *everybody reads Shakespeare at school*, here *Shakespeare* stands for his *books* rather than himself [1]. Metonymy poses a major challenge to compositional semantics. In order to interpret metonymy, compositional models should make use of the clues from higher levels of semantic processing like discourse.

How do we formally represent semantics of words, phrases and text? Many frameworks exist for representing semantics. The most common ones in compositional semantics are formal semantics, and distributional semantics. Each framework has its own pros and cons. We will describe them in the coming sections. Depending on the framework we use, additional challenges creep in. We use distributional framework for all our compositional models, thus our research of interest is compositional distributional semantics.

How do we make use of information from all the above sources and compose the semantics of an expression? Each semantic framework has its own way of using the above information. Composition functions are the most common which take constituent words and structure as input arguments, and the resultant semantic composition of the expression as the output.

[1] I took this example from my discussions with Percy Liang

The evaluation of compositional models is an uncertain topic. Since we humans do not know the way we compose semantics of expressions, it is hard to prepare datasets for evaluation, thus making the evaluation of compositional models a challenging topic. Most evaluation methods are external application-based.

1.3 What is thesis about?

In this thesis, we pursue some of the challenges described above. We aim to explore the effect of polysemy in compositional models. Our hypothesis is that *sense disambiguation improves the performance of compositional models*. Our focus is also on evaluation methods for compositional models (Chapter 2). We create a compositionality dataset using Mechanical Turkers, and based on the dataset, we perform a study on the relation between constituent words and phrase compositionality, revealing interesting facts about compositionality in language (Chapter 3). We evaluate sense disambiguation-based composition models using three evaluation methods, two application based and one an internal evaluation. We show improvements in performance due to sense disambiguation over standard models which do not perform disambiguation, thus validating our initial hypothesis (Chapter 4, 5, 6). Finally we discuss interesting findings from our observations (Chapter 7).

1.4 Background

In this section we describe the background required to follow the upcoming chapters.

1.4.1 Representation of Semantics

In compositional semantics, two different frameworks have become popular for representing semantics - (1) formal semantics and (2) distributional semantics.

1.4.1.1 Formal Semantics

In formal semantics, semantics of an expression is represented in formal logic based on the grammatical structure while the meanings of words are symbolic with no rigid definition. According to Montague [1970] view of formal semantics, a human language can be modeled within a mathematically precise theory. The advantage of formal semantics is its generative power. A formal semantic model can be represented by a grammar which translates (parses) a given expression into formal logic. Formal semantic models are known for their wide coverage.

The semantic representation of the sentence *every man walks*, according to Montague [1973], is defined as $\forall u[man(u) \implies walk(u)]$. Some of the formal semantic methods include [Baldridge and Kruijff, 2002; Ge and Mooney, 2005; Copestake *et al.*, 2005; Kate and Mooney, 2007; Chen and Mooney, 2008; Liang *et al.*, 2011; Kwiatkowski *et al.*, 2011]. Language processing applications can make use of formal representation and reason on it.

The major drawback of formal representation is that it only deals with truth or falsity of meanings of an expression, but do not say anything about how to compare two different meanings. Formal semantic models deal more with syntax not worrying about lexical ambiguity. Since our focus is on lexical ambiguity, formal semantics is not our topic of interest in this thesis.

1.4.1.2 Distributional Semantics

Distributional hypothesis [Harris, 1954] states that *words that occur in similar contexts tend to have similar meanings*. Firth [1957] states it as *you shall know a word by the company it keeps*. Distributional hypothesis is the backbone of statistical semantics, also called as distributional semantics.

In distributional semantics, a word is represented by a distribution of its contexts. For a given word, the distribution of its contexts can be learned from the co-occurrence frequency of the contexts and the target word. Two words are said to be similar if they have similar distribution of contexts. For example, *house* and *flat* frequently occur with context words like *rent, bedroom, sale* etc, giving a clue to computational models that *house* and *flat* may be similar. Context of a word can be defined as its neighboring

words in a fixed size window, their part-of-speech categories or the syntactic information or the combinations of any of these.

Similar to the representation of a word, an expression can be also represented as a distribution of contexts. The goal of compositional distributional semantic models is to predict this distribution for an expression from the distributional representation of its constituents.

Ambiguity of words is well studied in distributional semantics. So we choose distributional framework to test our hypothesis. The most common implementation of distributional models are vector space models (described in the next section).

The main advantage of distributional models is their ability to give a quantitative assessment on the similarity between meanings. However, distributional models are not generative like formal semantic models. It is highly challenging to encode structure of an expression into a distributional representation. It is also challenging to decode the structure of an expression from its distributional meaning.

1.4.2 Vector space model of meaning

Vector Space Models (VSM) of distributional semantics [Turney and Pantel, 2010] have become a standard framework for representing a word's meaning. Typically these methods [Schütze, 1998; Pado and Lapata, 2007; Erk and Padó, 2008] utilize a bag-of-words model or syntactic dependencies such as subject/verb, object/verb relations, so as to extract the features which serve as the dimensions of the vector space. Each word is then represented as a vector of the extracted features, where the frequency of co-occurrence of the word with each feature is used to calculate the vector component associated with that feature. Phrases can also be represented as vectors by treating a phrasal unit to like a single word. Figure 1.1 provides a sample vector space representation of a phrase and its constituents assuming bag-of-words model.

Our VSM settings: The lemmatised context words along with their part of speech category around a target word in a window of size 100 are treated as its co-occurrences, e.g. evidence-n, fire-v etc. Concordances of words from

	vector dimensions				
	evidence-n	memo-n	health-n	pistol-n	fire-v
$\overrightarrow{\textbf{smoking}}$	⟨ 9	2	181	4	37 ⟩
$\overrightarrow{\textbf{gun}}$	⟨ 10	3	5	98	270 ⟩
$\overrightarrow{\textbf{smoking gun}}$	⟨ 83	33	6	0	6 ⟩

Figure 1.1: Co-occurrence vectors of *smoking gun* and its constituents

ukWaC corpus [Ferraresi *et al.*, 2008] are used to compute co-occurrence frequencies of context words. The top 10000 frequent content words in ukWaC corpus (along with their part-of-speech category) are only treated as co-occurrences i.e. the vector dimensions. Vector of a word is built from its concordances in ukWaC. To measure the similarity between two vectors, we use cosine similarity (*sim*).

$$sim(\overrightarrow{\mathbf{v1}}, \overrightarrow{\mathbf{v2}}) = \frac{\overrightarrow{\mathbf{v1}} \cdot \overrightarrow{\mathbf{v2}}}{||\overrightarrow{\mathbf{v1}}||\ ||\overrightarrow{\mathbf{v2}}||}$$

Following Mitchell and Lapata [2008], the context words in the vector are set to the ratio of probability of the context word given the target word to the overall probability of the context word[2].

1.4.3 Compositional Distributional Semantics

Compositional Distributional Semantics (CDS) models deal with the task of composing the meaning of a phrase/sentence from the distributional meaning of its constituents and the structure. These models define composition function ($\oplus$), which takes constituent word vectors and structure as input, and gives the resultant semantic composition as output. Below, we discuss some of the composition functions.

Mitchell and Lapata [2008] use simple addition and simple multiplication of constituent word vectors to compose phrasal semantics. For example, for the phrase *house hunting*

- Simple addition: $\oplus$(house hunting) = $a\ \overrightarrow{\textbf{house}} + b\ \overrightarrow{\textbf{hunting}}$ where a and b are scalars.

[2]This is similar to point-wise mutual information without logarithm

- Simple multiplication: $\oplus$(house hunting) = $\overrightarrow{\textbf{house}}$ $\overrightarrow{\textbf{hunting}}$ where $\oplus(\text{house hunting})_i = \overrightarrow{\textbf{house}}_i * \overrightarrow{\textbf{hunting}}_i$

The resulting composition does not take structure into account, e.g. $\overrightarrow{\textbf{house}}\oplus \overrightarrow{\textbf{hunting}}$ looks the same as $\overrightarrow{\textbf{hunting}} \oplus \overrightarrow{\textbf{house}}$ (if a=b).

The background should be enough by now to understand the rest of the thesis. Readers can skip to Chapter 2. **Interested readers on composition functions may proceed.**

Erk and Padó [2008] make the above model structure sensitive by using selectional preferences of constituents. Figure 1.2 displays the composition procedure. To compose the semantics of the phrase $\overrightarrow{\textbf{catch}} \leftarrow obj - \overrightarrow{\textbf{ball}}$, the semantic preference vector of *catch* formed by all its objects filter the contexts of $\overrightarrow{\textbf{ball}}$, and similarly the semantic preference vector of *ball* formed by all its inverse objects filter the contexts of $\overrightarrow{\textbf{catch}}$, and these filtered vectors are used for composition. In this setting, the composition of *house hunting* and *hunting house* differs.

Widdows [2008] use tensor product to account for word order. The composition of house hunting is defined as

$$\oplus(\text{house hunting}) = \sum_{i,j} \overrightarrow{\textbf{house}}_i * \overrightarrow{\textbf{house}}_j \quad [\overrightarrow{\textbf{house}}_{\textbf{i}} \times \overrightarrow{\textbf{hunting}}_{\textbf{i}}]$$

If the initial vector space is n-dimensional, the resultant vector space of $\oplus$(house hunting) is n^2 dimensions.

Guevara [2010] propose additive and multiplicative models which look slightly similar to [Mitchell and Lapata, 2008].

- Additive Model: $\oplus$(house hunting) = $\textbf{A}$ $\overrightarrow{\textbf{house}}$ + $\textbf{B}$ $\overrightarrow{\textbf{hunting}}$ where A and B are matrices.
- Multiplicative Model: $\oplus$(house hunting) = $\textbf{A}$ $\overrightarrow{\textbf{house}}$ $\overrightarrow{\textbf{hunting}}$, where A is a matrix.

The matrices account for structure making the composition word order sensitive.

Socher *et al.* [2011] proposed a similar model to Guevara [2010] where syn-

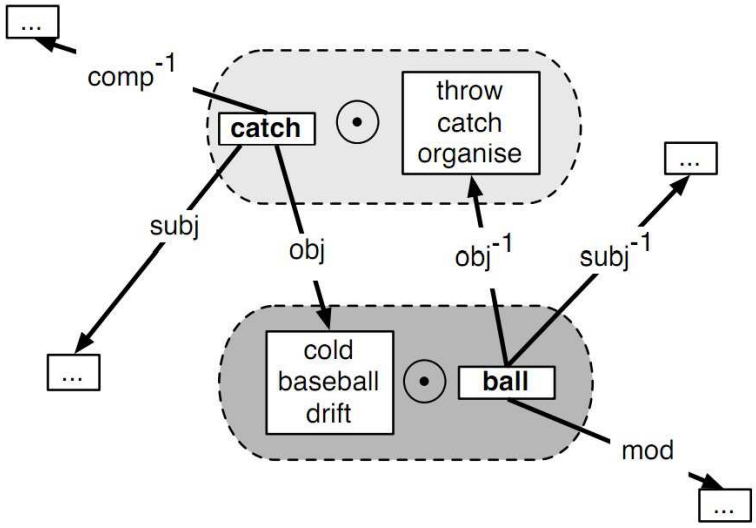

Figure 1.2: Composition using Structured vector space model. Courtesy: [Erk and Padó, 2008]

tactic relation between words is represented using a neural network (sigmoid-like function) which takes argument word vectors as input, and gives the resultant phrase composition vector as output.

Clark and Pulman [2007] aim to capture structure of a phrase/sentence by representing compositional meaning in a higher-order dimensional space using tensor product operation $\otimes$. Take an example sentence *"the boy ate a juicy orange"*. The structure can be represented as *boy* –subj→ *ate* ←obj– *orange* ←mod– *juicy*. Composition of this sentence is defined as:

$$\overrightarrow{\mathbf{boy}} \otimes \overrightarrow{\mathbf{subj}} \otimes (\overrightarrow{\mathbf{ate}} \otimes \overrightarrow{\mathbf{obj}} \otimes (\overrightarrow{\mathbf{juicy}} \otimes \overrightarrow{\mathbf{mod}} \otimes \overrightarrow{\mathbf{orange}}))$$

Sentences with different lengths are located in different higher-order dimensional spaces making it infeasible to measure the similarity between two unequal sentences. Dimensions of the space increase exponentially with increase in sentence length. The vector representation of a dependency relation is unclear. There is no experimental implementation of this work yet.

Grefenstette and Sadrzadeh [2011] assume that words belong to different type-based categories, and different categories exist in different dimensional spaces. The category of a word is decided by the number of adjoints

(arguments) it can take. The composition of a sentence results in a final vector which exists in sentential space. The vectors of verbs, adjectives and adverbs act as relational functions which modify the properties of noun vectors. For example, *"the boy ate a juicy orange"* results in a composition

$$\overrightarrow{\mathbf{ate}} \odot (\overrightarrow{\mathbf{boy}} \otimes (\overrightarrow{\mathbf{juicy}} \odot \overrightarrow{\mathbf{orange}}))$$

where $\odot$ is point-wise multiplication acting as a filtering operator and $\otimes$ is a tensor operator which is taking structure (order) into consideration. The above equation can be interpreted as: The selectional preferences of the relational words (here *ate*, and *juicy*) are filtering the noise in their corresponding arguments (nouns).

Let the category of a noun be n. Noun is assumed not to demand any arguments. The category of a relational word (verb, adjective, adverb) is decided by the number of arguments (adjoints) it takes and the category of the resultant output after combining with its arguments. For example, the adjoint of an adjective is noun (n^r) (located right) and the category after combining with noun is a noun n (adjective combines with a noun resulting in a noun(-phrase)). So the category is $adj = nn^r$. Similarly, lets say a verb can take at most two adjoints one on the left (subj n^l) and one on the right (object n^r), and the category of the output when the verb is combined with subject and object is a sentence s. So the category of verb is defined as $v = n^l s n^r$. Similarly category of an adverb is $adv = v^l v$. These categories decide the vector space in which corresponding words live.

All the resulting sentential vectors exist in the same space which is remarkable. However, the main limitations are

- As the number of adjoints of a word increase, the space in which it lives increases exponentially.
- It is always a prerequisite to have a verb to compute sentential semantics.
- The method is not designed for phrases.
- Similarity between words in different categories cannot be computed since they exist in different vector spaces e.g. the noun *running* and the verb *run*.

Chapter 2

Evaluation Methods

The aim of CDS models is to predict the semantics or the semantic behavior of the phrase from the constituents. But how do we say the predictions are correct? To date, the evaluation of CDS models is still a very uncertain issue. In this chapter, we give an overview of existing evaluation methods, their advantages and limitations.

There have been multiple proposals on evaluating CDS models. Most of them evaluate semantic behavior of the phrase rather than the evaluating the predicted semantics. Semantic behavior is evaluated on the basis of models' performance in reproducing human annotations on external tasks. Evaluating the predicted semantics require a comparison with the true semantics. The true semantics of phrases is not straightforward to capture, after all the goal of CDS is to predict this. However, distributional representation of the phrase obtained from a large corpus gives us an idea of its true semantics. CDS models are evaluated on their ability to reproduce this distributional representation observed from a corpus. This evaluation is considered to be an internal evaluation task.

External evaluation tasks include, but are not limited to, paraphrasing, compositionality detection, lexical substitution, summarization. Recently, a couple of these tasks have been integrated into a single shared task, and is being organized as a SemEval 2013 shared task [1].

In the followings sections, we describe three evaluation methods which are

[1] http://www.cs.york.ac.uk/semeval-2013/task5/

Annotator	N	N'	rating
4	phone call	committee meeting	2
25	phone call	committee meeting	7
11	football club	league match	6
11	health service	bus company	1
14	company director	assistant manager	7

Table 2.1: Evaluation dataset of [Mitchell and Lapata, 2010]

of particular interest to us.

2.1 Paraphrasing or Phrasal Similarity

Given a phrase, paraphrasing is the task of choosing alternative phrases which are similar to the given phrase. Paraphrasing datasets are prepared by human annotators. Humans generate paraphrases of a given phrase and also rank them based on the similarity with the given phrase. A good CDS model should correlate well with human rankings of paraphrases. The task can also be called as phrasal similarity task.

Mitchell and Lapata [2010] prepared a dataset which contains pairs of compound nouns and their similarity judgments. The dataset consists of 108 compound noun pairs with each pair having 7 annotations from different annotators who judge the pair for similarity in the range of score 1-7. A sample of 5 compound pairs is displayed in Table 2.1.

For each pair of the compound nouns, we take the mean value of all its human annotations as the final similarity judgment of the compound. Let N and N' be a pair. To evaluate a model, we calculate the cosine similarity between the composed vectors $\overrightarrow{\oplus(\mathbf{N})}$ and $\overrightarrow{\oplus(\mathbf{N'})}$ obtained from the composition, where $\oplus()$ denotes composition function. These similarity scores are correlated with human mean scores to judge the performance of a model. Higher the correlation, better is the CDS model.

Mitchell and Lapata [2008] also prepared a similar dataset for subject-verb phrases. Each phrase is paired with two landmark verbs, the synonyms of the reference verb in the phrase. The landmarks represent distinct word senses of the reference verb, one compatible with the reference phrase and the other incompatible e.g, for *The face glowed*, the landmarks *burned* and

	Noun	Reference	High	Low
The	fire	glowed	burned	beamed
The	face	glowed	beamed	burned
The	child	strayed	roamed	digressed
The	discussion	strayed	digressed	roamed
The	sales	slumped	declined	slouched
The	shoulders	slumped	slouched	declined

Table 2.2: Example Stimuli with High and Low similarity landmarks. Courtesy: Mitchell and Lapata [2008]

beamed are synonyms of *glowed* representing different senses of *glowed* while *beamed* is compatible with the reference phrase, *burned* is incompatible. A good CDS model should be able to compose the semantics of the phrase such that the phrasal vector is similar (closer) to the high-similarity landmark and different (farther) to the low-similarity landmark. Table 2.2 displays a sample from [Mitchell and Lapata, 2008] dataset.

In SemEval-2013, a shared task called *Identifying semantically similar phrases in context* is being organized based on idea of paraphrasing. For a given phrase, the participating systems should predict best similar phrases from very large corpora. Later, these phrases will be ranked by humans for phrase similarity. The evaluation method is kind-of reverse program to [Mitchell and Lapata, 2010].

The advantages of all the above methods in this evaluation are

- Since the final goal is only to predict or rank similar phrases of a given phrase, the evaluation method is independent of the dimensional space used by the CDS models.
- The evaluation method is easy to interpret and have many practical applications in natural language generation.

While the disadvantages are

- The evaluation is "external" since the actual composition task is not evaluated but evaluated on a different task.
- The evaluation method involves human annotations making the task expensive.

2.2 Compositionality Detection

A phrase is compositional if its meaning can be interpreted from the meaning of its constituents e.g. *swimming pool*. Not all phrases in a language are compositional. For example, the meaning of *couch potato* is hard to interpret from the meaning of *couch* and *potato*. Such phrases are non-compositional. Some phrases fall in-between compositional and non-compositional e.g. *rush hour*. The task of compositionality detection involves in identifying phrases which are compositional and non-compositional[2].

It is unclear if compositional models are expected to compose the semantics of non-compositional phrases. Pelletier [1994] presents arguments in favor of and against the notion of compositional models (compositionality principle) modeling the semantics of non-compositional phrases. Many existing methods [Schone and Jurafsky, 2001; Baldwin *et al.*, 2003; Giesbrecht, 2009] for compositionality detection assume compositional meaning from CDS models is completely different from non-compositional meaning. If a phrase is non-compositional, a good CDS model should compose the semantics of the phrase such that it is father from its actual meaning. If the phrase is compositional, the composition should lead to a meaning closer to the actual meaning. Based on this assumption, CDS models are evaluated on compositionality detection tasks.

In this evaluation, human annotate datasets with compositionality judgments. CDS models are evaluated based on their ability in reproducing human compositionality judgments of the annotated phrases. Recently Biemann and Giesbrecht [2011] organized a shared task based on the compositionality detection criteria.

There are many existing datasets marked with compositionality judgments. All the existing datasets are type-based evaluation datasets and are not context based evaluations. For example, *red carpet* have both compositional and non-compositional meaning. Type-based evaluation datasets are annotated only for the most frequent compositional behavior of the phrase (and therefore *red carpet* is non-compositional) and not context-dependent variation (In `The floor is covered with red carpet`, *red carpet* is com-

[2]For a deeper linguistic classification of phrases (multiwords), please refer to [Sag *et al.*, 2002]

positional). In SemEval 2013, a shared task is proposed on the idea of context-based evaluation of compositionality.

Existing type-based evaluation datasets either classify phrases into different classes or have scores demonstrating the degree of composition. Bannard *et al.* [2003] found moderate inter-annotator agreements in classifying the compounds into discrete classes, depicting the task is hard even for humans. Instead, McCarthy *et al.* [2003] suggests that compositionality exhibits a continuum, and created a dataset marked with compositionality scores rather than discrete classes.

In the next chapter, we discuss about existing type-based evaluation datasets and point out their limitations. We propose an annotation scheme different from the existing approaches and create a compositionality dataset for compound nouns. Our dataset is found to exhibit the continuum of compositionality.

The advantages of compositionality detection based evaluations are

- CDS models are evaluated both for compositionality and non-compositionality.
- The evaluation method is independent of the dimensional space used by the CDS models.
- The task may lead to creating better language understanding models.

However, the main disadvantages are

- The task is hard even for humans to classify phrases into compositional and non-compositional.
- The dataset is expensive to prepare.
- The evaluation is "external" since CDS models are evaluated on a task different to the actual task.

2.3 Similarity with Gold Phrasal Vectors (GPV metric)

In this evaluation metric, we evaluate compositional models by measuring the similarity between the distributional vector of a phrase built from the

corpus (*Gold Phrasal Vectors*) and the composed vector predicted by the models. A similar evaluation metric is proposed by Guevara [2011]. The evaluation is considered an *internal* evaluation metric since the evaluation is assessing the performance of CDS models on the actual semantic composition task and not an external task.

Given a set of n phrases, gold distributional vectors $\overrightarrow{\mathbf{G1}}$, $\overrightarrow{\mathbf{G2}}$ ... of the phrases are constructed using corpus instances of the phrases by treating the phrase as a single word unit, similar to building constituent word vectors. Let A and B be two CDS models. CDS model A is said to be better than B, if A's composed vectors $\overrightarrow{\mathbf{A1}}$, $\overrightarrow{\mathbf{A2}}$ ... of the given phrases are closer to $\overrightarrow{\mathbf{G1}}$, $\overrightarrow{\mathbf{G2}}$ ... than B's composed vectors $\overrightarrow{\mathbf{B1}}$, $\overrightarrow{\mathbf{B2}}$ ....

Let a_1, a_2 ..., where $a_i = sim(\overrightarrow{\mathbf{Ai}}, \overrightarrow{\mathbf{Gi}})$, be the cosine similarities of model A's composed vectors and gold vectors. Similarly, b_1, b_2 ..., where $b_i = sim(\overrightarrow{\mathbf{Bi}}, \overrightarrow{\mathbf{Gi}})$, be the cosine similarities of model B's composed vectors and gold vectors. Model A performance is measured by calculating its overall similarity defined as

$$GPV(A) = \frac{\sum_1^n a_i}{n}$$

Similarly Model B's overall similarity is defined as $GPV(B) = \frac{\sum_1^n b_i}{n}$. The model which gives higher overall similarity is a better compositional model than the one which gives lower similarity score. An ideal model should give an overall similarity score of 1.

The advantages of this model are

- The method does not require human annotated data, thus is less expensive and faster to create.
- "Internal" evaluation metric which evaluates the actual task.

However, the limitations are

- The method is badly affected by data sparsity as the length of the phrase increases.
- The method does not work for low frequency phrases even if the phrasal length is small.

- The predicted compositional vector should exist in the same space, thereby restricting the semantic composition functions that can be used.
- Method cannot be applied to phrases which do not occur in general language.
- It is unclear how to evaluate non-compositional phrases

2.4 Summary

In the above sections we discussed evaluation methods for CDS models and described three such methods in detail. There is no hard-and-fast rule in choosing an evaluation method. It mainly depends on the implementation of the CDS model and the task of interest. For example, GPV metric (Section 2.3) cannot be used if the composition vectors exist in different dimensional space than the gold vectors. In the coming chapters, we use the above mentioned evaluation methods to evaluate our CDS models.

In the next chapter we discuss the evaluation metric *Compositionality Detection* in detail. We propose a new annotation scheme for annotating compositionality judgments, describe an experimental setup for collecting annotations from many annotators, and evaluate computational methods for compositionality detection on our dataset.

Chapter 3

An Empirical Study on Compositionality

In the previous chapter we introduced *compositionality detection* evaluation method (Section 2.2). In this chapter, we collect and analyze the compositionality judgments for a range of compound nouns using Mechanical Turk to create a new compositionality detection dataset. Unlike existing compositionality datasets, our dataset has judgments on the contribution of constituent words as well as judgments for the phrase as a whole. We use this dataset to study the relation between the judgments at constituent level to that for the whole phrase. We introduce simple models of compositionality detection and evaluate them on our new dataset.

The past decade has seen interest in developing computational methods for compositionality detection [Lin, 1999; Schone and Jurafsky, 2001; Baldwin *et al.*, 2003; Bannard *et al.*, 2003; McCarthy *et al.*, 2003; Venkatapathy and Joshi, 2005; Katz and Giesbrecht, 2006; Sporleder and Li, 2009]. Recent developments in vector-based semantic composition functions [Mitchell and Lapata, 2008; Widdows, 2008] have also been applied to compositionality detection [Giesbrecht, 2009]. All these methods use constituent word semantics in contrast with the phrasal semantics to determine the compositionality of the phrase.

While the existing methods of compositionality detection use constituent word level semantics, the evaluation datasets are not particularly suitable

to study the contribution of each constituent word to the semantics of the phrase. Existing datasets [McCarthy *et al.*, 2003; Venkatapathy and Joshi, 2005; Katz and Giesbrecht, 2006; Biemann and Giesbrecht, 2011] only have the compositionality judgment of the whole expression without constituent word level judgment, or they have judgments on the constituents without judgments on the whole [Bannard *et al.*, 2003]. Our dataset allows us to examine the relationship between the two rather than assume the nature of it.

We collect judgments of the contribution of constituent nouns within noun-noun compounds (Section 3.1) alongside judgments of compositionality of the compound. We study the relation between the contribution of the parts with the compositionality of the whole (Section 3.2). We propose various constituent based models (Section 3.3.2) which are intuitive and related to existing models of compositionality detection (Section 3.3.1) and we evaluate these models in comparison to composition function based models. All the models discussed in this chapter are built using a distributional word-space model approach [Sahlgren, 2006].

3.1 Compositionality in Compound Nouns

In this section, we describe the experimental setup for the collecting compositionality judgments of English compound nouns. All the existing datasets focused either on verb-particle, verb-noun or adjective-noun phrases. Instead, we focus on *compound nouns* for which resources are relatively scarce. In this chapter, we only deal with compound nouns made up of two words separated by space.

3.1.1 Annotation setup

In the literature [Nunberg *et al.*, 1994; Baldwin *et al.*, 2003; Fazly *et al.*, 2009], compositionality is discussed in many terms including simple decomposable, semantically analyzable, idiosyncratically decomposable and non-decomposable. For practical NLP purposes, Bannard *et al.* [2003] adopt a straightforward definition of a compound being compositional if *"the overall semantics of the multi-word expression (here compound) can be composed*

from the simplex semantics of its parts, as described (explicitly or implicitly) in a finite lexicon". We adopt this definition and pose compositionality as a literality issue. *A compound is compositional if its meaning can be understood from the literal (simplex) meaning of its parts.* Similar views of compositionality as literality are found in [Lin, 1999; Katz and Giesbrecht, 2006]. In the past there have been arguments in favor/disfavor of compositionality as literality approach (e.g. see [Gibbs, 1989; Titone and Connine, 1999]). The idea of viewing compositionality as literality is also motivated from the shared task organized by Biemann and Giesbrecht [2011]. From here on, we use the terms compositionality and literality interchangeably.

We ask humans to score the compositionality of a phrase by asking them *how literal the phrase is.* Since we wish to see in our data the extent that the phrase is compositional, and to what extent that depends on the contribution in meaning of its parts, we also ask them *how literal the use of a component word is within the given phrase.*

For each compound noun, we create three separate tasks – one for each constituent's literality and one for the phrase compositionality. Tasks for the compound noun *"sacred cow"* are displayed in the Figure 3.1. The motivation behind using three separate tasks is to make the scoring mechanism for each task independent of the other tasks. This enables us to study the actual relation between the constituents and the compound scores without any bias to any particular annotator's way of arriving at the scores of the compound w.r.t. the constituents.

There are many factors to consider in eliciting compositionality judgments, such as ambiguity of the expression and individual variation of annotator in background knowledge. To control for this, we ask subjects if they can interpret the meaning of a compound noun from *only* the meaning of the component nouns where we also provide contextual information. All the possible definitions of a compound noun are chosen from WordNet [Fellbaum, 1998], Wiktionary or defined by ourselves if some of the definitions are absent. Five examples of each compound noun are randomly chosen from the ukWaC [Ferraresi *et al.*, 2008] corpus and the same set of examples are displayed to all the annotators. The annotators select the definition of the compound noun which occurs most frequently in the examples and then score the compound for literality based on the most frequent definition.

Phrase: *sacred cow*
Definitions:

1. a person unreasonably held to be immune to criticism
2. A cow which is worshipped

Examples:

1. we told our director , Kenneth Loach , that none of the *sacred cows* of television drama need stand in his way
2. Meles Zenawi said in an interview that there were no *sacred cows* in a war on corruption
3. many of the *sacred cows* will have to be sacrificed to fund digitization
4. you will find any number of *sacred cows* which are regarded as an intrinsic part of the teachings. Think of reincarnation, chakras, karma
5. TOTP has finally become the latest *sacred cow* to be slaughtered by the BBC

Instructions:

- Select the definition of *sacred cow* which occurs most of the times in the above examples. Ignore other definitions. Based on the definition chosen, score below tasks
- Scoring guidelines: Enter a number between 0 and 5
  - 0 means: Not to be understood literally at all
  - 5 means: To be understood very literally
  - Use values in between to grade your decision

Note: Each task below is dispalyed separately to different annotators.
Task1: Score of 0-5 for how literal is the phrase *sacred cow*
Task2: Score of 0-5 for how literal is the use of *sacred* in the phrase *sacred cow*
Task3: Score of 0-5 for how literal is the use of *cow* in the phrase *sacred cow*

Figure 3.1: Sample annotation tasks for *sacred cow*

We have two reasons for making the annotators read the examples, choose the most frequent definition and base literality judgments on the most frequent definition. The first reason is to provide a context to the decisions and reduce the impact of ambiguity. The second is that distributional models are greatly influenced by frequency and since we aim to work with distributional models for compositionality detection we base our findings on the most frequent sense of the compound noun. In this work we consider the compositionality of the noun-noun compound type without token based disambiguation which we leave for future work.

3.1.2 Compound noun dataset

We could not find any compound noun datasets publicly available which are marked for compositionality judgments. Korkontzelos and Manandhar [2009] prepared a related dataset for compound nouns but compositional-

ity scores were absent and their set contains only 38 compounds. There are datasets for verb-particle [McCarthy *et al.*, 2003], verb-noun judgments [Biemann and Giesbrecht, 2011; Venkatapathy and Joshi, 2005] and adjective-noun [Biemann and Giesbrecht, 2011]. Not only are these not the focus of our work, but also we wanted datasets with each constituent word's literality score. Bannard *et al.* [2003] obtained judgments on whether a verb-particle construction implies the verb or the particle or both. The judgments were binary and not on a scale and there was no judgment of compositionality of the whole construction. Ours is the first attempt to provide a dataset which have both scalar compositionality judgments of the phrase as well as the literality score for each component word.

We aimed for a dataset which would include compound nouns where: 1) both the component words are used literally, 2) the first word is used literally but not the second, 3) the second word is used literally but not the first and 4) both the words are used non-literally. Such a dataset would provide stronger evidence to study the relation between the constituents of the compound noun and its compositionality behaviour.

We used the following heuristics based on WordNet to classify compound nouns into 4 above classes.

1. Each of the component word exists either in the hypernymy hierarchy of the compound noun or in the definition(s) of the compound noun. e.g. *swimming pool* because *swimming* exists in the WordNet definition of *swimming pool* and *pool* exists in the hypernymy hierarchy of *swimming pool*
2. Only the first word exists either in the hypernymy hierarchy or in the definition(s) of the compound and not the second word. e.g. *night owl*
3. Only the second word exists either in the hypernymy hierarchy or in the definition(s) of the compound and not the first word. e.g. *zebra crossing*
4. Neither of the words exist either in hypernymy hierarchy or in the definition(s) of the compound noun. e.g. *smoking gun*

The intuition behind the heuristics is that if a component word is used literally in a compound, it would probably be used in the definition of the compound or may appear in the synset hierarchy of the compound. We

changed the constraints, for example decreasing/increasing the depth of the hypernymy hierarchy, and for each class we randomly picked 30 potential candidates by rough manual verification. There were fewer instances in the classes 2 and 4. In order to populate these classes, we selected additional compound nouns from Wiktionary by manually inspecting if they can fall in either class.

These heuristics were only used for obtaining our sample, they were *not* used for categorizing the compound nouns in our study. The compound nouns in all these temporary classes are merged and 90 compound words are selected which have at least 50 instances in the ukWaC corpus. These 90 compound words are chosen for the dataset.

3.1.3 Annotators

Snow *et al.* [2008] used Amazon mechanical turk (AMT) for annotating language processing tasks. They found that although an individual turker (annotator) performance was lower compared to an expert, as the number of turkers increases, the quality of the annotated data surpassed expert level quality. We used 30 turkers for annotating each single task and then retained the judgments with sufficient consensus as described in Section 3.1.4.

For each compound noun, 3 types of tasks are created as described above: a judgment on how literal the phrase is and a judgment on how literal each noun is within the compound. For 90 compound nouns, 270 independent tasks are therefore created. Each of these tasks is assigned to 30 annotators. A task is assigned randomly to an annotator by AMT so each annotator may work on only some of the tasks for a given compound.

3.1.4 Quality of the annotations

Recent studies[1] shows that AMT data is prone to spammers and outliers. We dealt with them in three ways. **a).** We designed a qualification test[2] which provides an annotator with basic training about literality, and they

[1] A study on AMT spammers `http://bit.ly/e1IPil`

[2] The qualification test details are provided with the dataset. Please refer to footnote 3.

No. of turkers participated	260	
No. of them qualified	151	
Turkers with $\rho <= 0$	21	
Turkers with $\rho >= 0.6$	81	
No. of annotations rejected	383	
Avg. submit time (sec) per task	30.4	
	highest ρ	avg. ρ
ρ for phrase compositionality	0.741	0.522
ρ for first word's literality	0.758	0.570
ρ for second word's literality	0.812	0.616
ρ for over all three task types	0.788	0.589

Table 3.1: Amazon Mechanical Turk statistics

can participate in the annotation task only if they pass the test. **b).** Once all the annotations (90 phrases * 3 tasks/phrase * 30 annotations/task = 8100 annotations) are completed, we calculated the average Spearman correlation score (ρ) of every annotator by correlating their annotation values with every other annotator and taking the average. We discarded the work of annotators whose ρ is negative and accepted all the work of annotators whose ρ is greater than 0.6. **c).** For the other annotators, we accepted their annotation for a task only if their annotation judgment is within the range of ± 1.5 from the task's mean. Table 3.1 displays AMT statistics. Overall, each annotator on average worked on 53 tasks randomly selected from the set of 270 tasks. This lowers the chance of bias in the data because of any particular annotator.

Spearman correlation scores ρ provide an estimate of annotator agreement. To know the difficulty level of the three types of tasks described in Section 3.1, ρ for each task type is also displayed in Table 3.1. It is evident that annotators agree more at word level than phrase level annotations. Thus, providing literality scores at component word level is an additional advantage of our dataset compared to the existing datasets on compositionality judgments.

For each compound, we also studied the distribution of scores around the mean by observing the standard deviation σ. All the compound nouns along with their mean and standard deviations are shown in Table 3.2.

Ideally, if all the annotators agree on a judgment for a given compound or a component word, the deviation should be low. Among the 90 compounds, 15 of them are found to have a deviation $> \pm 1.5$. These are displayed in Table

Compound	Word1	Word2	Phrase	Compound	Word1	Word2	Phrase
climate change	4.90±0.30	4.83±0.38	4.97±0.18	engine room	4.86±0.34	5.00±0.00	4.93±0.25
graduate student	4.70±0.46	5.00±0.00	4.90±0.30	swimming pool	4.80±0.40	4.90±0.30	4.87±0.34
speed limit	4.93±0.25	4.83±0.38	4.83±0.46	research project	4.90±0.30	4.53±0.96	4.82±0.38
application form	4.77±0.42	4.86±0.34	4.80±0.48	bank account	4.87±0.34	4.83±0.46	4.73±0.44
parking lot	4.83±0.37	4.77±0.50	4.70±0.64	credit card	4.67±0.54	4.90±0.30	4.67±0.70
ground floor	4.66±0.66	4.70±0.78	4.67±0.60	mailing list	4.67±0.54	4.93±0.25	4.67±0.47
call centre	4.73±0.44	4.41±0.72	4.66±0.66	video game	4.50±0.72	5.00±0.00	4.60±0.61
human being	4.86±0.34	4.33±1.14	4.59±0.72	interest rate	4.34±0.99	4.69±0.53	4.57±0.90
radio station	4.66±0.96	4.34±0.80	4.47±0.72	health insurance	4.53±0.88	4.83±0.58	4.40±1.17
law firm	4.72±0.52	3.89±1.50	4.40±0.76	public service	4.67±0.65	4.77±0.62	4.40±0.76
end user	3.87±1.12	4.87±0.34	4.25±0.87	car park	4.90±0.40	4.00±1.10	4.20±1.05
role model	3.55±1.22	4.00±1.03	4.11±1.07	head teacher	2.93±1.51	4.52±1.07	4.00±1.16
fashion plate	4.41±1.07	3.31±2.07	3.90±1.42	balance sheet	3.82±0.89	3.90±0.96	3.86±1.01
china clay	2.00±1.84	4.62±1.00	3.85±1.27	game plan	2.82±1.96	4.86±0.34	3.83±1.23
brick wall	3.16±2.20	3.53±1.86	3.79±1.75	web site	2.68±1.69	3.93±1.18	3.79±1.21
brass ring	3.73±1.95	3.87±1.98	3.72±1.84	case study	3.66±1.12	4.67±0.47	3.70±0.97
polo shirt	1.73±1.41	5.00±0.00	3.37±1.38	rush hour	3.11±1.37	2.86±1.36	3.33±1.27
search engine	4.62±0.96	2.25±1.70	3.32±1.16	cocktail dress	1.40±1.08	5.00±0.00	3.04±1.22
face value	1.39±1.11	4.64±0.81	3.04±0.88	chain reaction	2.41±1.16	4.52±0.72	2.93±1.14
cheat sheet	2.30±1.59	4.00±0.83	2.89±1.11	blame game	4.61±0.67	2.00±1.28	2.72±0.92
fine line	3.17±1.34	2.03±1.52	2.69±1.21	front runner	3.97±0.96	1.29±1.10	2.66±1.32
grandfather clock	0.43±0.78	5.00±0.00	2.64±1.32	lotus position	1.11±1.17	4.78±0.42	2.48±1.22
spelling bee	4.81±0.77	0.52±1.04	2.45±1.25	silver screen	1.41±1.57	3.23±1.45	2.38±1.63
smoking jacket	1.04±0.82	4.90±0.30	2.32±1.29	spinning jenny	4.67±0.54	0.41±0.77	2.28±1.08
number crunching	4.48±0.77	0.97±1.13	2.26±1.00	guilt trip	4.71±0.59	0.86±0.94	2.19±1.16
memory lane	4.75±0.51	0.71±0.80	2.17±1.04	crash course	0.96±0.94	4.23±0.92	2.14±1.27
rock bottom	0.74±0.89	3.80±1.08	2.14±1.19	think tank	3.96±1.06	0.47±0.62	2.04±1.13
night owl	4.47±0.88	0.50±0.82	1.93±1.27	panda car	0.50±0.56	4.66±1.15	1.81±1.07
diamond wedding	1.07±1.29	3.41±1.34	1.70±1.05	firing line	1.61±1.65	1.89±1.50	1.70±1.72
pecking order	0.78±0.92	3.89±1.40	1.69±0.88	lip service	2.03±1.25	1.75±1.40	1.62±1.06
cash cow	4.22±1.07	0.37±0.73	1.56±1.10	graveyard shift	0.38±0.61	4.50±0.72	1.52±1.17
sacred cow	1.93±1.65	0.96±1.72	1.52±1.52	silver spoon	1.59±1.47	1.44±1.77	1.52±1.45
flea market	0.38±0.81	4.71±0.84	1.52±1.13	eye candy	3.83±1.05	0.71±0.75	1.48±1.10
rocket science	0.64±0.97	1.55±1.40	1.43±1.35	couch potato	3.27±1.48	0.34±0.66	1.41±1.03
kangaroo court	0.17±0.37	4.43±1.02	1.37±1.05	snail mail	0.60±0.80	4.59±1.10	1.31±1.02
crocodile tears	0.19±0.47	3.79±1.05	1.25±1.09	cutting edge	0.88±1.19	1.73±1.63	1.25±1.18
zebra crossing	0.76±0.62	4.61±0.86	1.25±1.02	acid test	0.71±1.10	3.90±1.24	1.22±1.26
shrinking violet	2.28±1.44	0.23±0.56	1.07±1.01	sitting duck	1.48±1.48	0.41±0.67	0.96±1.04
rat race	0.25±0.51	2.04±1.32	0.86±0.99	swan song	0.38±0.61	1.11±1.14	0.83±0.91
gold mine	1.38±1.42	0.70±0.81	0.81±0.82	rat run	0.41±0.62	2.33±1.40	0.79±0.66
nest egg	0.79±0.98	0.50±0.87	0.78±0.87	agony aunt	1.86±1.22	0.43±0.56	0.76±0.86
snake oil	0.37±0.55	0.81±1.25	0.75±1.12	monkey business	0.67±1.01	1.85±1.30	0.72±0.69
smoking gun	0.71±0.75	1.00±0.94	0.71±0.84	silver bullet	0.52±1.00	0.55±1.10	0.67±1.15
melting pot	1.00±1.15	0.48±0.63	0.54±0.63	ivory tower	0.38±1.03	0.54±0.68	0.46±0.68
cloud nine	0.47±0.62	0.23±0.42	0.33±0.54	gravy train	0.30±0.46	0.45±0.77	0.31±0.59

Table 3.2: Compounds with their constituent and phrase level mean±deviation scores

brass ring	brick wall	cheat sheet	china clay
cutting edge	fashion plate	fine line	firing line
game plan	head teacher	sacred cow	silver screen
search engine	silver spoon	web site	

Table 3.3: Ambiguous Compounds with $\sigma > \pm 1.5$

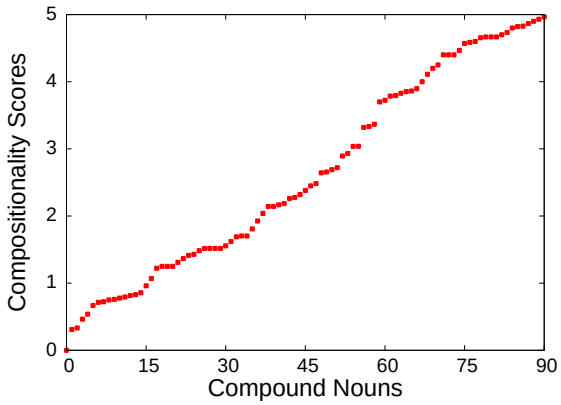

Figure 3.2: Mean values of phrase-level compositionality scores

3.3. We used this threshold to signify annotator disagreement. The reasons for annotator disagreement vary. From our analysis, some of the compounds are found to be compositionally ambiguous displaying both compositional and non-compositional nature at the same time. For e.g. *silver screen* in the example, *"Mike Myers talk about the improved technology used to bring Shrek 2 to the silver screen"* some think *silver screen* means *film industry* and others think in the meaning *cinema screen* which is actually silver in color. Some examples like *brass ring*, though compositionally not ambiguous, they exhibit equal chances of compositional and non-compositional usage in the corpus. This was evident when the random examples picked from the corpus are analyzed. For others such as *search engine* some think *engine* has only a little to do with *search engine* and the others disagree.

Overall, the inter annotator agreement (ρ) is high and the standard deviation of most tasks is low (except for a few exceptions). So we are confident that the dataset can be used as a reliable gold-standard with which we conduct experiments. The dataset is publicly available for download[3].

[3]Annotation guidelines, Mechanical Turk hits, qualification test, annotators demographic and educational background, and final annotations are downloadable from `http://sivareddy.in/downloads`

3.2 Analyzing the Human Judgments

By analyzing the mean values of phrase level annotations, we found that compounds displayed a varied level of compositionality. For some compounds annotators confirm that they can interpret the meaning of a compound from its component words and for some they do not. For others they grade in-between. Figure 3.2 displays the mean values of compositionality scores of all compounds. Compounds are arranged along the X-axis in increasing order of their score. The graph displays a *continuum of compositionality* [McCarthy *et al.*, 2003]. We note that our sample of compounds is selected to exhibit a range of compositionality.

3.2.1 Relation between the constituents and the phrase compositionality judgments

The dataset allows us to study the relation between constituent word level contributions to the phrase level compositionality scores.

Let $w1$ and $w2$ be the constituent words of the compound $w3$. Let $s1$, $s2$ and $s3$ be the mean literality scores of $w1$, $w2$ and $w3$ respectively. Using a 3-fold cross validation on the annotated data, we tried various function fittings f over the judgments $s1$, $s2$ and $s3$.

- ADD: $a.s1 + b.s2 = s3$
- MULT: $a.s1.s2 = s3$
- COMB: $a.s1 + b.s2 + c.s1.s2 = s3$
- WORD1: $a.s1 = s3$
- WORD2: $a.s2 = s3$

where a, b and c are coefficients.

We performed 3-fold cross validation to evaluate the above functions (two training samples and one testing sample at each iteration). The coefficients of the functions are estimated using least-square linear regression technique over the training samples. The average Spearman correlation scores (ρ) over testing samples are displayed in Table 3.4. The goodness of fit R^2 values when trained over the whole data are also displayed in Table 3.4.

Function f	ρ	$\mathbf{R^2}$
ADD	0.966	0.937
MULT	0.965	0.904
COMB	0.971	0.955
WORD1	0.767	0.609
WORD2	0.720	0.508

Table 3.4: Correlations between functions and phrase compositionality scores

Results (both ρ and R^2) clearly show that a relation exists between the constituent literality scores and the phrase compositionality. Existing compositionality approaches on noun-noun compounds such as [Baldwin *et al.*, 2003; Korkontzelos and Manandhar, 2009] use the semantics of only *one* of the constituent words (generally the head word) to determine the compositionality of the phrase. But the goodness of fit R^2 values show that the functions ADD, COMB and MULT which intuitively make use of *both* the constituent scores fit the data better than functions using only one of the constituents. Furthermore, COMB and ADD suggest that additive models are preferable to multiplicative. In this data, the first constituent word plays a slightly more important role than the second in determining compositionality.

Overall, this study suggests that *it is possible to estimate the phrase level compositionality scores given the constituent word level literality scores.* This motivates us to present constituent based models (Section 3.3.2) for compositionality score estimation of a compound. We begin the next section on computational models with a discussion of related work.

3.3 Computational Models

3.3.1 Related work

Most methods in compositionality detection can be classified into two types - those which make use of lexical fixedness and syntactic properties of the MWEs, and those which make use of the semantic similarities between the constituents and the MWE.

Non compositional MWEs are known to have lexical fixedness in which the

component words have high statistical association. Some of the methods which exploit this feature are [Lin, 1999; Pedersen, 2011]. This property does not hold always because institutionalized MWEs [Sag *et al.*, 2002] are known to have high association even though they are compositional, especially in the case of compound nouns. Another property of non-compositional MWEs is that they show syntactic rigidness which do not allow internal modifiers or morphological variations of the components, or variations that break typical selectional preferences. Methods like [Cook *et al.*, 2007; McCarthy *et al.*, 2007; Fazly *et al.*, 2009] exploit this property. This holds mostly for verbal idioms but not for compound nouns since the variations of any compound noun are highly limited.

Other methods like [Baldwin *et al.*, 2003; Sporleder and Li, 2009] are based on semantic similarities between the constituents and the MWE. Baldwin *et al.* [2003] use only the information of the semantic similarity between one of the constituents and the compound to determine the compositionality. Sporleder and Li [2009] determine the compositionality of verbal phrases in a given context (token-based disambiguation) based on the lexical chain similarities of the constituents and the context of the MWE. Bannard *et al.* [2003] and McCarthy *et al.* [2003] study the compositionality in verb particles and they found that methods based on the similarity between simplex parts (constituents) and the phrases are useful to study semantics of the phrases. These findings motivated our constituent based models along with the findings in Section 3.2.1.

In addition to the constituent based models (Section 3.3.2), there are composition function based vector models [Mitchell and Lapata, 2008; Widdows, 2008] which make use of the semantics of the constituents in a different manner. These models are described in Section 3.3.3 and are evaluated in comparison with the constituent-based models.

All our models discussed in this thesis are based on the vector space model described in Section 1.4.2.

3.3.2 Constituent based models

Given a compound word $w3$ with the constituents $w1$ and $w2$, constituent based models determine the compositionality score $s3$ of the compound by

first determining the literality scores $s1$ and $s2$ of $w1$ and $w2$ respectively (Section 3.3.2.1) and then using one of the functions f (described in Section 3.2.1), the compositionality score $s3$ is estimated using $s3 = f(s1, s2)$ (Section 3.3.2.2).

3.3.2.1 Literality scores of the constituents

If a constituent word is used literally in a given compound it is highly likely that the compound and the constituent share common co-occurrences. For example, the compound *swimming pool* has the co-occurrences *water*, *fun* and *indoor* which are also commonly found with the constituents *swimming* and *pool*. In the compound *smoking gun* (see Figure 1.1), the co-occurrences of the constituents and the compound differ which show that either *smoking* or *gun* are not meant literally in the compound.

We define the literality of a word in a given compound as the similarity between the compound and the constituent co-occurrence vectors i.e. if the number of common co-occurrences are numerous then the constituent is more likely to be meant literally in the compound.

Let $\overrightarrow{\mathbf{v1}}$, $\overrightarrow{\mathbf{v2}}$ and $\overrightarrow{\mathbf{v3}}$ be the co-occurrence vectors of $w1$, $w2$ and $w3$. The literality scores $s1$ and $s2$ of $w1$ and $w2$ in the compound $w3$ are defined as

$$s1 = sim(\overrightarrow{\mathbf{v1}}, \overrightarrow{\mathbf{v3}})$$
$$s2 = sim(\overrightarrow{\mathbf{v2}}, \overrightarrow{\mathbf{v3}})$$

where sim is the cosine similarity between the vectors.

3.3.2.2 Compositionality of the compound

Given the literality scores $s1$ and $s2$ of the constituents, we can now compute the compositionality score $s3$ of the compound $w3$ using any of the functions f defined in Section 3.2.1.

$$s3 = f(s1, s2)$$

3.3.3 Composition function based models

In these models [Schone and Jurafsky, 2001; Katz and Giesbrecht, 2006; Giesbrecht, 2009] of compositionality detection, firstly a vector for the compound is composed from its constituents using a compositionality function $\oplus$. Then the similarity between the composed vector and true co-occurrence vector of the compound is measured to determine the compositionality: the higher the similarity, the higher the compositionality of the compound. Guevara [2011] observed that additive models performed well for building composition vectors of phrases from their parts whereas Mitchell and Lapata [2008] found in favor of multiplicative models. We experiment using both the compositionality functions simple addition[4] and simple multiplication, which are the most widely used composition functions, known for their simplicity and good performance.

Vector $\overrightarrow{\mathbf{v1}} \oplus \overrightarrow{\mathbf{v2}}$ for a compound w_3 is composed from its constituent word vectors $\overrightarrow{\mathbf{v1}}$ and $\overrightarrow{\mathbf{v2}}$ using simple addition $a\overrightarrow{\mathbf{v1}}+b\overrightarrow{\mathbf{v2}}$ and simple multiplication $\overrightarrow{\mathbf{v1}}\overrightarrow{\mathbf{v2}}$ where the i^{th} element of $\overrightarrow{\mathbf{v1}} \oplus \overrightarrow{\mathbf{v2}}$ is defined as

$$\begin{aligned} (a\overrightarrow{\mathbf{v1}} + b\overrightarrow{\mathbf{v2}})_i &= a\ \overrightarrow{\mathbf{v1}}_i\ +\ b\ \overrightarrow{\mathbf{v2}}_i \\ \overrightarrow{\mathbf{v1}}\overrightarrow{\mathbf{v2}}_i &= \overrightarrow{\mathbf{v1}}_i\ .\ \overrightarrow{\mathbf{v2}}_i \end{aligned}$$

The compositionality score of the compound is then measured using $s3 = sim(\overrightarrow{\mathbf{v1}} \oplus \overrightarrow{\mathbf{v2}}, \overrightarrow{\mathbf{v3}})$ where $\overrightarrow{\mathbf{v3}}$ is the co-occurrence vector of the compound built from the corpus. For more details of these models please refer to [Mitchell and Lapata, 2008; Giesbrecht, 2009].

3.3.4 Evaluation

We evaluated all the models on the dataset developed in Section 3.1. Since our dataset has constituent level contributions along with phrase compositionality judgments, we evaluated the constituent based models against both the literality scores of the constituents (Section 3.3.2.1) and the phrase

[4]Please note that simple additive model [Mitchell and Lapata, 2008] is different from the additive model described in [Guevara, 2011]. In [Mitchell and Lapata, 2008] the coefficients are real numbers whereas in [Guevara, 2011] they are matrices.

	first constituent	second constituent
s1	0.616	–
s2	–	0.707

Table 3.5: Constituent level correlations

Model	ρ	R^2
Constituent Based Models		
ADD	0.686	0.613
MULT	0.670	0.428
COMB	0.682	0.615
WORD1	0.669	0.548
WORD2	0.515	0.410
Compositionality Function Based Models		
$a\overrightarrow{\mathbf{v1}} + b\overrightarrow{\mathbf{v2}}$	0.714	0.620
$\overrightarrow{\mathbf{v1}}\overrightarrow{\mathbf{v2}}$	0.650	0.501
RAND	0.002	0.000

Table 3.6: Phrase level correlations of compositionality scores

level judgments (Section 3.3.2.2). The composition function models are evaluated only on phrase level scores following [McCarthy *et al.*, 2003; Venkatapathy and Joshi, 2005; Biemann and Giesbrecht, 2011]: higher correlation scores indicate better compositionality predictions.

Constituent based models evaluation

Spearman's ρ correlations of s1 and s2 with the human constituent level judgments are shown in Table 3.5. We observed that the predictions for the second constituent are more accurate than those for the first constituent. Perhaps these constitute an easier set of nouns for modelling but we need to investigate this further.

For the phrase compositionality evaluation we did a 3-fold cross validation. The parameters of the functions f (Section 3.3.2.2) are predicted by least square linear regression over the training samples and optimum values are selected. The average Spearman correlation scores of phrase compositionality scores with human judgments on the testing samples are displayed in Table 3.6. The goodness of fit R^2 values when trained over the whole dataset are also displayed.

It is clear that models ADD and COMB which use both the constituents are better predictors of phrase compositionality compared to the single word based predictors WORD1 and WORD2. Both ADD and COMB are competitive in terms of both the correlations (accuracy) and goodness of fit values. The model MULT shows good correlation but the goodness of fit is lower. First constituent (model WORD1 i.e. $sim(\overrightarrow{\mathbf{v1}}, \overrightarrow{\mathbf{v3}})$) was found to be a better predictor of phrase compositionality than the second (WORD2) following the behaviour of the mechanical turkers as in Table 3.4.

Composition function based models evaluation

These models are evaluated for phrase compositionality scores. As with the constituent based models, for estimating the model parameters a and b of the composition function based models, we did a 3-fold cross validation. The best results of additive model on the training samples are found at a=0.60 and b=0.40. Average Spearman correlation scores of both addition and multiplication models over the testing samples are displayed in Table 3.6. The goodness of fit R^2 values when trained over the whole dataset are also displayed.

Vector addition has a clear upper hand over multiplication in terms of both accuracy and goodness of fit for phrase compositionality prediction.

Winner between constituent and composition function based models

For phrase compositionality prediction (Table 3.6), both constituent based and compositionality function based models are found to be competitive, though compositionality function based models perform slightly better. The reason could be because while constituent based models use contextual information of each constituent *independently*, composition function models make use of collective evidence from the contexts of both the constituents *simultaneously*.

The notion, "*contexts salient to both the constituents are important for composition*", is fundamental behind our idea of *Dynamic Prototypes* discussed in Chapter 4.

All the model results, both constituent and composition function based models, when compared with random baseline model (RAND in Table 3.6) which assigns a random compositionality score to a compound, are highly significant.

3.4 Summary

In this chapter we examined the compositionality judgments of noun compounds and also the literality judgments of their constituent words. Our study reveals that both the constituent words play a major role in deciding the compositionality of the phrase. We showed that the functions which predict the compositionality using both the constituent literality scores have high correlations with compositionality judgments. Based on this evidence we proposed constituent based models for compositionality detection. We compared constituent based models with compositionality function based models. The additive compositionality function based model is the best performing of all, however the performance of constituent based models (ADD and COMB) is comparable.

All the 8100 annotations collected in this work are released publicly. We hope the dataset can reveal more insights into the compositionality in terms of the contribution from the constituents.

Future directions of this work include context based compositionality detection of phrases, and designing sophisticated constituent based models. Extending this study on other kinds of phrases such as adjective-noun, verb particle, verb-noun phrases may throw more light into our understanding of compositionality.

In this chapter we did not deal with the polysemy of constituents or phrase. For example in the phrase *bank account*, we represented *bank* using all its corpus instances without worrying about the polysemy of *bank*, and determined the compositionality of *bank account*. In the coming chapters we focus on the polysemy of constituent words, and aim to build better compositional models by performing sense disambiguation of constituents.

Chapter 4

Dynamic and Static Prototoypes

The goal of this chapter is to answer the question: *Does sense disambiguation improve semantic composition?* In the previous chapter, the semantic behavior of a phrase is predicted using the co-occurrence vectors of its constituent words. A co-occurrence vector of a constituent word is built by conflating all the corpus instances of the constituent, which essentially is equivalent to conflating all the senses of the constituent word. However, not all the senses of the constituents are relevant when composing the semantics of the compound.

For example, take the compound *house hunting*. Figure 4.1 displays the co-occurrence vectors of the words *house* and *hunting* built from all the corpus instances of *house* and *hunting* respectively. Using the composition functions $a\ \overrightarrow{\mathbf{v1}} + b\ \overrightarrow{\mathbf{v2}}$ and $\overrightarrow{\mathbf{v1}}\overrightarrow{\mathbf{v2}}$ defined in the previous chapter (Section 3.3.3), the meaning of *house hunting* can be composed from the vectors of *house* and *hunting*. Figure 4.2 displays the composed vectors of *house*

	vector dimensions					
	animal	buy	apartment	price	rent	kill
$\overrightarrow{\mathbf{house}}$	⟨ 30	60	90	55	45	10 ⟩
$\overrightarrow{\mathbf{hunting}}$	⟨ 90	15	12	20	33	90 ⟩

Figure 4.1: A hypothetical vector space model.

	vector dimensions					
	animal	buy	apartment	price	rent	kill
$\overrightarrow{\textbf{house}}$	⟨ 30	60	90	55	45	10 ⟩
$\overrightarrow{\textbf{hunting}}$	⟨ 90	15	12	20	33	90 ⟩
$\overrightarrow{\textbf{house}} + \overrightarrow{\textbf{hunting}}$	⟨ ***120***	75	102	75	78	***100*** ⟩
$\overrightarrow{\textbf{house}}\ \overrightarrow{\textbf{hunting}}$	⟨ ***2700***	900	1080	1100	1485	***900*** ⟩

Figure 4.2: Composition using simple addition and simple multiplication operators

hunting. As can be observed from figure 4.2, the resulting composed vectors do not reflect the correct meaning of the compound *house hunting* due to the presence of irrelevant co-occurrences such as *animal* or *kill*. These co-occurrences are relevant to one sense of *hunting*, i.e. (*the activity of hunting animals*), but not to the sense of *hunting* meant in *house hunting*, i.e. *the activity of looking thoroughly*. Given that *hunting* has been associated with a single prototype (vector) by conflating all of its senses, the application of any composition function $\oplus$ is likely to include irrelevant co-occurrences in $\overrightarrow{\textbf{house}} \oplus \overrightarrow{\textbf{hunting}}$.

A potential solution to this problem would involve the following steps:

1. build separate prototype vectors for each of the senses of *house* and *hunting*
2. select the relevant prototype vectors of *house* and *hunting* and then perform the semantic composition.

In this chapter we present two methods (section 4.2) for carrying out the above steps on noun-noun compounds. The first one (section 4.2.1) applies Word Sense Induction (WSI) to identity different senses (also called static multi prototypes) of the constituent words of a compound noun and then applies composition by choosing the relevant senses. The second method (section 4.2.2) does not identify a fixed set of senses. Instead, it represents each constituent by a prototype vector which is built dynamically (also called as a dynamic prototype) by activating only those contexts considered to be relevant to the constituent in the presence of the other constituent, and then performs the composition on the dynamic prototypes. For performing composition, we use vector composition functions.

Our evaluation (section 4.4) on a task for rating similarity between noun-noun compound pairs shows: (1) sense disambiguation of constituents improves semantic composition and (2) dynamic prototypes are better than static multi prototypes for semantic composition.

4.1 Related work

Relevant to our work is the work of Erk and Padó [2008] who utilize a structured vector space model. The prototype vector of a constituent word is initially built, and later refined by removing irrelevant co-occurrences with the help of the selectional preferences of other constituents. The refined vectors are then used for the semantic composition of the compound noun. The results are encouraging showing that polysemy is a problem in vector space models. Our approach differs to theirs in the way we represent meaning - we experiment with static multi prototypes and dynamic prototypes. Our vector space model is based on simple bag-of-words which does not require selectional preferences for sense disambiguation and can be applied to resource-poor languages.

There are several other researchers who tried to address polysemy for improving the performance of different tasks but not particularly to the task of semantic composition. Some of them are Navigli and Crisafulli [2010] for web search results clustering, Klapaftis and Manandhar [2010b] for taxonomy learning, Reisinger and Mooney [2010] for word similarity and Korkontzelos and Manandhar [2009] for compositionality detection. In all cases, the reported results demonstrate that handling polysemy lead to improved performance of the corresponding tasks. This motivates our research for handling polysemy for the task of semantic composition using two different methods described in the next section.

4.2 Sense Prototype Vectors

In this section we describe two approaches for building sense specific prototype vectors of constituent words in a noun-noun compound. The first approach performs WSI to build static multi prototype vectors. The other

builds a single dynamic prototype vector for each constituent by activating only the relevant exemplars of the constituent with respect to the other constituent. An exemplar is defined as a corpus instance of a target word.

These sense specific prototype vectors are then used for semantic composition. Let N be a compound noun with constituents n and n'. Our aim is to select the relevant senses of n and n'.

4.2.1 Static Multi Prototypes Based Sense Selection

In the first stage (section 4.2.1.1), a WSI method is applied to both n and n'. The outcome of this stage is a set of clusters (senses). Each of these clusters is associated with a prototype vector taking the centroid of the cluster. Following Reisinger and Mooney [2010] we use the terminology *multi prototype vectors* in the meaning of *sense clusters*. Let $S(n)$ (resp. $S(n')$) be the set of prototypes of n, where each $s_i^n \in S(n)$ denotes the i^{th} sense of the noun n. Since these prototypes of constituents are static and do not change when the compound changes we refer to them as *static multi prototypes*.

In the next stage (section 4.2.1.2), we calculate all the pairwise similarities between the clusters of n and n', so as to select a pair of clusters with the highest similarity. The selected clusters are then combined using a composition function, to produce a single vector representing the semantics of the target compound noun N.

4.2.1.1 Graph-based WSI

Word Sense Induction is the task of identifying the senses of a target word in a given text. We apply a graph-based sense induction method, which creates a graph of target word instances and then clusters that graph to induce the senses. We follow the work of Klapaftis and Manandhar [2010a] for creating the graph and apply *Chinese Whispers* (CW) [Biemann, 2006], a linear graph clustering method that automatically identifies the number of clusters.

Figure 4.3 provides a running example of the different stages of the WSI

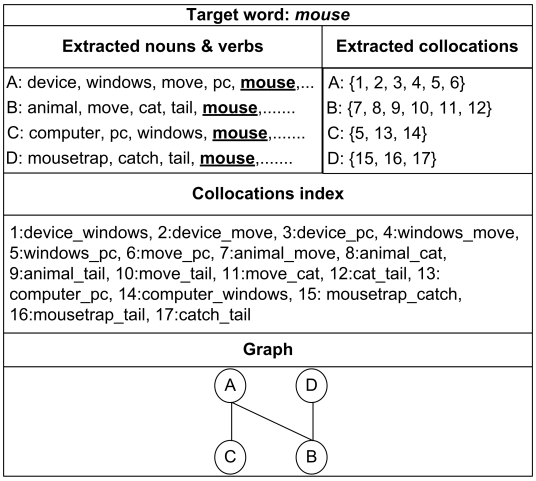

Figure 4.3: Running example of WSI

method. In the example, the target word *mouse* appears with the *electronic device* sense in the contexts A, C, and with the *animal* sense in the contexts B and D.

Corpus preprocessing: Let bc denote the base corpus consisting of the contexts containing the target word tw. In our work, a context is defined by a set of words in a window of size 100 around the target.

The aim of this stage is to capture words contextually related to tw. In the first step, the target word is removed from bc and part-of-speech tagging is applied to each context. Only nouns and verbs are kept and lemmatised. In the next step, the distribution of each word in the base corpus is compared to the distribution of the same noun in a reference corpus using the log-likelihood ratio (G^2) [Dunning, 1993]. Words that have a G^2 below a pre-specified threshold (parameter p_1) are removed from each context of the base corpus. The result of this stage is shown in the upper left part of Figure 4.3.

Graph creation & clustering: Each context $c_i \in bc$ is represented as a vertex in a graph G. Edges between the vertices of the graph are drawn based on their similarity, defined in Equation 4.1, where $sm_{cl}(c_i, c_j)$ is the *collocational weight* of contexts c_i, c_j and $sm_{wd}(c_i, c_j)$ is their bag-of-words weight. If the edge weight $W(c_i, c_j)$ is above a prespecified threshold (parameter p_3), then an edge is drawn between the corresponding vertices in

the graph.

$$W(c_i, c_j) = \frac{1}{2}(sm_{cl}(c_i, c_j) + sm_{wd}(c_i, c_j)) \tag{4.1}$$

Collocational weight: The limited polysemy of collocations is exploited to compute the similarity between contexts c_i and c_j. In this setting, a collocation is a juxtaposition of two words within the same context. Given a context c_i, a total of $\binom{N}{2}$ collocations are generated by combining each word with any other word in the context. Each collocation is weighted using the log-likelihood ratio (G^2) [Dunning, 1993] and is filtered out if the G^2 is below a prespecified threshold (parameter p_2). At the end of this process, each context c_i of tw is associated with a set of collocations (g_i) as shown in the upper right part of Figure 4.3 . Given two contexts c_i and c_j, the Jaccard coefficient is used to calculate the similarity between the collocational sets, i.e. $sm_{cl}(c_i, c_j) = \frac{|g_i \cap g_j|}{|g_i \cup g_j|}$.

Bag-of-words weight: Estimating context similarity using collocations may provide reliable estimates regarding the existence of an edge in the graph, however, it also suffers from data sparsity. For this reason, a bag-of-words model is also employed. Specifically, each context c_i is associated with a set of words (g_i) selected in the corpus preprocessing stage. The upper left part of Figure 4.3 shows the words associated with each context of our example. Given two contexts c_i and c_j, the bag-of-words weight is defined to be the Jaccard coefficient of the corresponding word sets, i.e. $sm_{wd}(c_i, c_j) = \frac{|g_i \cap g_j|}{|g_i \cup g_j|}$.

Finally, the collocational weight and bag-of-words weight are averaged to derive the edge weight between two contexts as defined in Equation 4.1. The resulting graph of our running example is shown on the bottom of Figure 4.3. This graph is the input to CW clustering algorithm. Initially, CW assigns all vertices to different classes. Each vertex i is processed for an x number of iterations and inherits the strongest class in its local neighborhood LN in an update step. LN is defined as the set of vertices which share a direct connection with vertex i. During the update step for a vertex i: each class C_k receives a score equal to the sum of the weights of edges (i, j), where j has been assigned class C_k. The maximum score determines the strongest class. In case of multiple strongest classes, one is chosen randomly. Classes

Parameter	Range
G^2 word threshold (p_1)	15,25,35,45
G^2 collocation threshold (p_2)	10,15,20
Edge similarity threshold (p_3)	0.05,0.09,0.13

Table 4.1: WSI parameter values.

are updated immediately, which means that a node can inherit classes from its LN that were introduced in the same iteration.

Experimental setting The parameters of the WSI method were fine-tuned on the nouns of the SemEval-2007 word sense induction task [Agirre and Soroa, 2007] under the second evaluation setting of that task, i.e. supervised (WSD) evaluation. We tried various parameter combinations shown in Table 4.1. Specifically, we selected the parameter combination p_1=15, p_2=10, p_3= 0.05 that maximized the performance in this evaluation. We use ukWaC [Ferraresi *et al.*, 2008] corpus to retrieve all the instances of the target words.

4.2.1.2 Cluster selection

The application of WSI on the nouns $n \in N$ and $n' \in N$ results in two sets of clusters (senses) $S(n)$ and $S(n')$. Cluster $S(n)$ is a set of contexts of the word n. Each context is represented as an exemplar $\overrightarrow{\mathbf{e}}$, a vector specific to the context. Only the 10000 most frequent words in the ukWaC (along with their part-of-speech category) are treated as the valid co-occurrences i.e. the dimensionality of the vector space is 10000. For example, the exemplar of *hunting* in the context *"the-x purpose-n of-i autumn-n hunting-n be-v in-i part-n to-x cull-v the-x number-n of-i young-j autumn-n fox-n"* is $\langle$ purpose-n:1, autumn-n:2, part-n:1, cull-v, number-n:1, young-j:1, fox-n:1 $\rangle$

For every cluster s_i^n in $S(n)$ we construct a prototype vector $\overrightarrow{\mathbf{v^{s_i^n}}}$ by taking the centroid of all the exemplars in the cluster. Following Mitchell and Lapata [2008], the context words in the prototype vector are set to the ratio of probability of the context word given the target word to the overall probability of the context word[1].

The next step is to choose the relevant sense of each constituent for a given compound. We assume that the meaning of a compound noun can be ap-

[1]This is similar to pointwise mutual information without logarithm

proximated by identifying the most similar senses of each of its constituent nouns. Accordingly all the pairwise similarities between the $\overrightarrow{\mathbf{v}^{\mathbf{s_i^n}}}$ and $\overrightarrow{\mathbf{v}^{\mathbf{s_i^{n'}}}}$ are calculated using cosine similarity and the pair with maximum similarity is chosen for composition.

followed by huge tarpon that like to use the	**light**	of your torch to help them hunt. At the
the Christmas trade this year or the	**lights**	will be off, probably for ever. The Merrymen
embrace better health - but doing so in the	**light**	of real and trusted information about the
present your organisation in a professional	**light**	and in a way our all our clients value.
continues to be significant, together with other	**light**	industries such as electrical engineering
and near-infrared light, along with red	**light**	emitted by hydrogen atoms and green light

Figure 4.4: Six random sentences of *light* from ukWaC

4.2.2 Dynamic Prototype Based Sense Selection

Kilgarriff [1997] argues that representing a word with a fixed set of senses is not a good way of modelling word senses. Instead word senses should be defined according to a given context. We propose a dynamic way of building word senses for the constituents of a given compound.

We use an exemplar-based approach to build the dynamic sense of a constituent with the help of other constituent. In exemplar-based modelling [Erk and Padó, 2010; Smith and Medin, 1981], each word is represented by all its exemplars without conflating them into a single vector. Depending upon the purpose, only relevant exemplars of the target word are activated. Exemplar-based models are more powerful than just prototype based ones because they retain specific instance information. As described in the previous section, an exemplar is a vector that represents a single instance of a given word in the corpus.

Let E_n be the set of exemplars of the word n. Given a compound N with constituents n and n', we remove irrelevant exemplars in E_n creating a refined set $E_n^{n'} \subset E_n$ with the help of the other constituent word n'. The prototype vector $\overrightarrow{\mathbf{n}^{\mathbf{n'}}}$ of n is then built from the centroid of the refined exemplar set $E_n^{n'}$. The vector $\overrightarrow{\mathbf{n}^{\mathbf{n'}}}$ represents the relevant prototype vector (sense) of n in the presence of the other constituent word n'. Unlike the static prototypes defined in the previous section, the prototype vectors of n and n' are built dynamically based on the given compound. Therefore, we refer to them as dynamic prototype vectors.

4.2.2.1 Building Dynamic Prototypes

We demonstrate our method of building dynamic prototypes with an example. Let us take the compound *traffic light*. Let $\overrightarrow{\textbf{Traffic}}$, $\overrightarrow{\textbf{Light}}$ and $\overrightarrow{\textbf{TrafficLight}}$ denote the prototype vectors of *traffic*, *light* and *traffic light* respectively. Word *light* occurs in many contexts such as quantum theory, optics, lamps and spiritual theory. In ukWaC, *light* occurs with 316,126 exemplars. Figure 4.4 displays 6 random sentences of *light* from ukWaC. None of these exemplars are related to the target compound *traffic light*. When a prototype vector of *light* is built from all its exemplars, irrelevant exemplars add noise increasing the semantic differences between *traffic light* and *light* and thereby increasing the semantic differences between $\overrightarrow{\textbf{TrafficLight}}$ and $\overrightarrow{\textbf{Traffic}} \oplus \overrightarrow{\textbf{Light}}$. This is not desirable. The cosine similarity sim($\overrightarrow{\textbf{Light}}$, $\overrightarrow{\textbf{TrafficLight}}$) is found to be 0.27.

We aim to remove irrelevant exemplars of *light* with the help of the other constituent word *traffic* and then build a prototype vector of *light* which is related to the compound *traffic light*. Our intuition and motivation for exemplar removal is that it is beneficiary to choose only the exemplars of *light* which have context words related to *traffic* since the exemplars of *traffic light* will have context words related to both *traffic* and *light*. For example car, road, transport will generally be found within the contexts of all the words *traffic*, *light* and *traffic light*.

We rank each exemplar of *light* with the help of collocations of *traffic*. Collocations of *traffic* are defined as the context words which frequently occur with *traffic*, e.g. car, road etc. The exemplar of *light* representing the sentence "*Cameras capture **cars** running red **lights** ...*" will be ranked higher than the one which does not have context words related to *traffic*. We use Sketch Engine[2] [Kilgarriff *et al.*, 2004] to retrieve the collocations of *traffic* from ukWaC. Sketch Engine computes the collocations using Dice metric [Dice, 1945]. We build a collocation vector $\textbf{Traffic}^{\textbf{colloc}}$ from the collocations of *traffic*.

We also rank each exemplar of *light* using the distributionally similar words to *traffic* i.e. words which are similar to *traffic* e.g. transport, flow etc. These distributionally similar words helps to reduce the impact of data sparseness

[2]Sketch Engine `http://www.sketchengine.co.uk`

and helps prioritize the contexts of *light* which are semantically related to *traffic*. Sketch Engine is again used to retrieve distributionally similar words of *traffic* from ukWaC. Sketch Engine ranks similar words using the method of Rychlý and Kilgarriff [2007]. We build the vector $\mathbf{Traffic^{similar}}$ which consists of the similar words of *traffic*.

Every exemplar **e** from the exemplar set E_{light} [3] is finally ranked by

$$sim(\mathbf{e}, \mathbf{Traffic^{colloc}}) + sim(\mathbf{e}, \mathbf{Traffic^{similar}})$$

We choose the top n% of the ranked exemplars in E_{light} to construct a refined exemplar set $E_{\text{light}}^{\text{traffic}}$. A prototype vector $\overrightarrow{\mathbf{Light^{Traffic}}}$ is then built by taking the centroid of $E_{\text{light}}^{\text{traffic}}$. $\overrightarrow{\mathbf{Light^{Traffic}}}$ denotes the sense of *light* in the presence of *traffic*. Since sense of *light* is built dynamically based on the given compound (here *traffic light*), we define $\overrightarrow{\mathbf{Light^{Traffic}}}$ as the dynamic prototype vector. The similarity sim($\overrightarrow{\mathbf{Light^{Traffic}}}$, $\overrightarrow{\mathbf{TrafficLight}}$) is found to be 0.47 which is higher than the initial similarity 0.27 of $\overrightarrow{\mathbf{Light}}$ and $\overrightarrow{\mathbf{TrafficLight}}$. This shows that our new prototype vector of *light* is closer to the meaning of *traffic light*.

Similarly we build the dynamic prototype vector $\overrightarrow{\mathbf{Traffic^{Light}}}$ of *traffic* with the help of *light*. The dynamic prototypes $\overrightarrow{\mathbf{Traffic^{Light}}}$ and $\overrightarrow{\mathbf{Light^{Traffic}}}$ are used for semantic composition to construct $\overrightarrow{\mathbf{Traffic^{Light}}} \oplus \overrightarrow{\mathbf{Light^{Traffic}}}$

4.3 Composition functions

Given a compound, we perform composition using the sense based prototypes selected in the above section. We use the composition functions simple addition (ADDITION) and simple multiplication (MULTIPLICATION) described in Section 3.3.3.

[3] In E_{light}, we do not include the sentences which have the compound noun *traffic light* occurring in them.

$$\begin{aligned}
&\text{ADDITION:} && \overrightarrow{\oplus(\mathbf{N})} = a\ \overrightarrow{\mathbf{n}} + b\ \overrightarrow{\mathbf{n}'} \\
& && \text{i.e. } \overrightarrow{\oplus(\mathbf{N})}_i = a\ \overrightarrow{\mathbf{n}}_i + b\ \overrightarrow{\mathbf{n}}'_i \\
&\text{MULTIPLICATION:} && \overrightarrow{\oplus(\mathbf{N})} = \overrightarrow{\mathbf{nn}'} \\
& && \text{i.e. } \overrightarrow{\oplus(\mathbf{N})}_i = \overrightarrow{\mathbf{n}}_i\ .\ \overrightarrow{\mathbf{n}}'_i
\end{aligned} \tag{4.2}$$

where a and b are real numbers.

For the function ADDITION, we use equal weights for both constituent words i.e. $a = b = 1$. For the function MULTLIPLICATION there are no parameters.

4.4 Evaluation

In this section we present the evaluation dataset, evaluation scheme and the models evaluated. We use the evaluation scheme *Phrasal Similarity* (Section 2.1) described in Chapter 2. For convenience, we present a brief description of the dataset and evaluation scheme again.

4.4.1 Dataset

Mitchell and Lapata [2010] prepared a dataset[4] which contains pairs of compound nouns and their similarity judgments. The dataset consists of 108 compound noun pairs with each pair having 7 annotations from different annotators who judge the pair for similarity. A sample of 5 compound pairs is displayed in Figure 4.5.

4.4.2 Evaluation Scheme

For each pair of the compound nouns, the mean value of all its annotations is taken to be the final similarity judgment of the compound.

Let N and N' be a pair. To evaluate a model, we calculate the cosine similarity between the composed vectors $\overrightarrow{\oplus(\mathbf{N})}$ and $\overrightarrow{\oplus(\mathbf{N}')}$ obtained from

[4]We would like to thank Jeff Mitchell and Mirella Lapata for sharing the dataset.

Annotator	N	N'	rating
4	phone call	committee meeting	2
25	phone call	committee meeting	7
11	football club	league match	6
11	health service	bus company	1
14	company director	assistant manager	7

Figure 4.5: Evaluation dataset of [Mitchell and Lapata, 2010]

the composition on sense based prototypes generated by the model. These similarity scores are correlated with human mean scores to judge the performance of the model.

4.4.3 Models Evaluated

We evaluate all the models w.r.t. the composition functions ADDITION and MULTIPLICATION.

Static Single Prototypes: This model does not perform any sense disambiguation and is similar to the method described in [Mitchell and Lapata, 2008]. The prototype vector of each constituent formed by conflating all its instances is used to compose the vector of the compound.

Static Multi Prototypes: In the method described in section 4.2.1, word sense induction produces a large number of clusters i.e. static multi prototypes. We tried various parameters like choosing the target prototype of a constituent only from the top 5 or 10 large clusters.

Static Multi Prototypes with Guided Selection: This is similar to Static Multi Prototypes model except in the way we choose the relevant prototype for each constituent. In section 4.2.1.2 we described an unsupervised way of prototype selection from multi prototypes. Unlike there, here we choose the constituent prototype (sense) which has the highest similarity to the prototype vector of the compound. This is a guided way of sense selection since we are using the compound prototype vector which is built from the compound's corpus instances. The performance of this model gives us an idea of the upper boundary of multi prototype models for semantic composition.

Dynamic Prototypes: In the method described in section 4.2.2, the dynamic prototype of a constituent is produced from the top n% exemplars of the ranked exemplar set of the constituent. We tried various percent activation (n%) values - 2%, 5%, 10%, 20%, 50%, 80%.

Compound Prototype: We directly use the corpus instances of a compound to build the prototype vector of the compound. This method does not involve any composition. Ideally, one expects this model to give the best performance.

4.5 Results and Discussion

All the above models are evaluated on the dataset described in section 4.4.1. Table 4.2 displays the Spearman correlations of all these models with the human annotations (mean values).

The results of Static Single Prototypes model are consistent with the previous findings of Mitchell and Lapata [2010], in which MULTIPLICATION performed better than ADDITION.

All the parameter settings of Dynamic Prototypes outperformed Static Single Prototypes. This shows that selecting the relevant sense prototypes of the constituents improve semantic composition. We also observe that the highest correlation is achieved by including just the top 2% exemplars for each constituent. It seems that as the sample of exemplars increases, noise increases as well, and this results in a worse performance.

The comparison between Static Single Prototypes and Static Multi Prototypes shows that the former performs significantly better than the latter. This is not according to our expectation. The possible reason for poor performance could be because of the sense selection process (section 4.2.1.2) which might have failed to choose the relevant sense of each constituent word.

However, Static Multi Prototypes with Guided Sense Selection still fail to perform better than Static Single Prototypes. Therefore, we can conclude that the lower performance of Static Multi Prototypes cannot be attributed to the sense selection process only. Despite that, the applied graph clustering

Parameter Description	ADDITION	MULTIPLICATION
Static Single Prototypes		
	0.5173	0.6104
Static Multi Prototypes		
Top 5 clusters	0.1171	0.4150
Top 10 clusters	0.0663	0.2655
Static Multi Prototypes with Guided Selection		
Top 5 clusters	0.2290	0.4187
Top 10 clusters	0.2710	0.4140
Dynamic Prototypes		
Top 2 % exemplars	0.6261	**0.6552**
Top 5 % exemplars	0.6326	0.6478
Top 10 % exemplars	**0.6402**	0.6515
Top 20 % exemplars	0.6273	0.6359
Top 50 % exemplars	0.5948	0.6340
Top 80 % exemplars	0.5612	0.6355
Compound Prototype		
	0.4152	

Table 4.2: Spearman Correlations of Model predictions with Human Predictions

method results in the generation of a very large number of clusters, some of which refer to the same word usage with subtle differences. Hence, our future work focuses on a selection process that chooses multiple relevant clusters of a constituent word. Additionally, our ongoing work suggests that the use of verbs as features in the graph creation process (section 4.2.1.1) causes the inclusion of noisy edges and results in worse clustering.

Our evaluation also shows that Dynamic Prototypes provide a better semantic composition than Static Multi Prototypes. The main reason for this result stems from the fact that Dynamic Prototypes explicitly identify the relevant usages of a constituent word with respect to the other constituent and vice versa, without having to deal with a set of issues that affect the performance of Static Multi Prototypes such as the clustering and the sense selection process.

The performance of Compound Prototype is lower than the compositional models. The reason could be due to the data sparsity. Data sparsity is known to be a major problem for modelling the meaning of compounds. In a way, the results are encouraging for compositional models.

In all these models, the composition function MULTIPLICATION gave a better performance than ADDITION.

4.6 Summary

This chapter presented two methods for dealing with polysemy when modeling the semantics of a noun-noun compound. The first one represents senses by creating static multi prototype vectors, while the second represents context-specific sense of a word by generating a dynamic prototype vector. Our experimental results show that: (1) sense disambiguation improves semantic composition, and (2) dynamic prototypes are a better representation of senses than static multi prototypes for the task of semantic composition.

Future direction of this work include using all or some of the multiple static prototypes similar to [Reisinger and Mooney, 2010] rather than selecting a single prototype for composition. This gives a better idea if at all static prototypes are useful for composition. It will also be interesting to test Dynamic prototypes on the traditional word sense disambiguation tasks. Dynamic prototypes present a different mechanism for sense representation unlike traditional methods.

In the coming chapters, we will present additional experiments with Dynamic prototypes on other evaluation tasks presented in Chapter 2. In the conclusions chapter, we present a bigger picture on the possible reasons why dynamic prototypes perform better than static prototypes.

Chapter 5

Compositionality Detection with Dynamic Prototypes

In the previous chapter, we pointed out that polysemy is a problem in CDS models, and showed *Dynamic Prototypes* improve the performance of phrasal similarity based composition task by performing sense disambiguation. In this chapter, we will discuss the problems due to polysemy in compositionality detection methods, and experiment with dynamic prototypes for improving compositionality detection. We participated with dynamic prototype-based systems in the ACL 2011 shared task on compositionality detection [Biemann and Giesbrecht, 2011]. Our systems were ranked the best in two evaluation criteria and the second best in two other evaluation criteria. The organizers Biemann and Giesbrecht [2011] claimed our system as the most robust among all the participating systems.

5.1 Problems due to polysemy in Compositionality Detection

Distributional methods of compositionality detection make use of both *distributional hypothesis* [Harris, 1954] and *the principle of compositionality* [Partee, 1995, page. 313] to detect compositionality. The idea is described in the following paragraphs.

The distributional hypothesis (DH) states that words that occur in similar contexts tend to have similar meanings. Vector space model (VSM) described in Section 1.4.2 is based on this hypothesis that the similarity between two meanings is the *closeness* (proximity) between the vectors. A compound word can be represented as a co-occurrence vector using all the corpus instances where the compound occurs. This is one way of representing the meaning of a compound.

The other way of representing the meaning of a compound is based on the principle of semantic compositionality (PSC). PSC states that the meaning of a compound word is a function of, and only of, the meaning of its parts and the way in which the parts are combined. The composition functions described in Section 4.3 are based on this principle. If the meaning of a part is represented in a VSM using the distributional hypothesis, then the principle can be applied to compose the distributional behaviour of a compound word from its parts without actually using the corpus instances of the compound. We refer to this as a PSC-based vector. So a PSC-based is composed of component DH-based vectors.

Both of these two mechanisms are capable of determining the meaning vector of a compound word. For a given compound, if a DH-based vector and a PSC-based vector of the compound are projected into an identical space, one would expect the vectors to occupy the same location i.e. both the vectors should be nearly the same. However the principle of semantic compositionality does not hold for non-compositional compounds, which is actually what the existing VSMs of compositionality detection exploit [Giesbrecht, 2009; Katz and Giesbrecht, 2006; Schone and Jurafsky, 2001]. The DH-based and PSC-based vectors are expected to have high similarity when a compound is compositional and low similarity for non-compositional compounds.

All these methods represent a word by a single prototype vector conflating all its corpus instances. These prototype-based vectors do not distinguish the instances according to the senses of a target word. Since most compounds are less ambiguous than single words, there is less need for distinguishing instances in a DH-based prototype vector of a compound. However the constituent words of the compound are more ambiguous. When DH-based vectors of the constituent words are used for composing the PSC-based vector of the compound, the resulting vector may contain instances and therefore

contexts, that are not relevant for the given compound. These noisy contexts effect the similarity between the PSC-based vector and the DH-based vector of the compound. Basing compositionality judgments on a such a noisy similarity value is not reliable.

In this chapter, we address this problem of polysemy of constituent words by building compositionality detection models using the *Dynamic Prototypes* proposed in the previous chapter. We have evaluated our models on the validation data released in the shared task [Biemann and Giesbrecht, 2011]. Based on the validation results, we have chosen three systems for public evaluation and participated in the shared task [Biemann and Giesbrecht, 2011].

5.2 Related Work

Let *w1w2* be a compound with constituent words w1 and w2. $\overrightarrow{\mathbf{w1}}$, $\overrightarrow{\mathbf{w2}}$ and $\overrightarrow{\mathbf{w3}}$ be the co-occurrence vectors of *w1*, *w2* and *w1w2* respectively built from a corpus. *sim* denotes cosine similarity. As described above, most distributional models for compositionality detection measure the similarity between the distributional vector $\overrightarrow{\mathbf{w1w2}}$ of the compound and the composed vector $\overrightarrow{\mathbf{w1}} \oplus \overrightarrow{\mathbf{w2}}$, where $\oplus$ denotes a compositionality function. If the similarity is high, the compound is treated as compositional or else non-compositional.

Giesbrecht [2009]; Katz and Giesbrecht [2006]; Schone and Jurafsky [2001] obtained the compositionality vector of *w1w2* using vector addition $a\ \overrightarrow{\mathbf{w1}} + b\ \overrightarrow{\mathbf{w2}}$ (refer equation 4.2). In this approach, if $sim(\overrightarrow{\mathbf{w1}} \oplus \overrightarrow{\mathbf{w2}}, \overrightarrow{\mathbf{w1w2}}) > \gamma$, the compound is classified as compositional, where γ is a threshold for deciding compositionality. Global values of a and b were chosen by optimizing the performance on the development set. It was found that no single threshold value γ held for all compounds. Changing the threshold alters performance arbitrarily. This might be due to the polysemous nature of the constituent words which makes the composed vector $\overrightarrow{\mathbf{w1}} \oplus \overrightarrow{\mathbf{w2}}$ filled with noisy contexts and thus making the judgment unpredictable.

In the above model, if a=0 and b=1, the resulting model is similar to that of Baldwin *et al.* [2003]. They also observe similar behaviour of the threshold γ. We try to address this problem by addressing the polysemy in VSMs using

dynamic prototypes.

Bannard *et al.* [2003]; McCarthy *et al.* [2003] observed that methods based on distributional similarities between a phrase and its constituent words help when determining the compositionality behaviour of phrases. We therefore also use evidence from the similarities between each constituent word and the compound.

5.3 Dynamic Prototype-based Compositionality Detection Models

Our approach works as follows. Firstly, given a compound $w1w2$, we build its DH-based prototype vector $w1w2$ from all its corpus instances. Secondly, we build the dynamic prototypes $\overrightarrow{\mathbf{w1^{w2}}}$ and $\overrightarrow{\mathbf{w2^{w1}}}$, which represent sense specific prototypes relevant in the given context (for each word in a phrase, other constituent word is its context). Using these dynamic prototypes, we build the PSC-based composed vector of *w1w2* using composition functions. The vector similarities between DH and PSC based vectors are used for compositionality detection.

5.3.1 Vector Space Model

Our vector space model is similar to as described in section 1.4.2. The only difference is that instead of using 10000 top frequent content words, we used only top 2000 top frequent content words to make our models run faster. $\overrightarrow{\mathbf{w1w2}}$ is built in this vsm using all its corpus instances. In addition, the exemplars of *w1* and *w2* are also represented in this VSM. Using these exemplars, the dynamic prototypes $\overrightarrow{\mathbf{w1^{w2}}}$ and $\overrightarrow{\mathbf{w2^{w1}}}$ are built as described in section 4.2.2.1. These vectors are used for composition.

5.3.2 Building Compositional Vectors

We use the compositionality functions, simple addition and simple multiplication to build compositional vectors (Section 4.3). In model addition, $a\ \overrightarrow{\mathbf{w1}}\ +\ b\ \overrightarrow{\mathbf{w2}}$, all the previous approaches use static values of a and b.

Instead, we use dynamic weights computed from the participating vectors using $a = \frac{sim(\overrightarrow{\mathbf{w1w2}}, \overrightarrow{\mathbf{w1}})}{sim(\overrightarrow{\mathbf{w1w2}}, \overrightarrow{\mathbf{w1}}) + sim(\overrightarrow{\mathbf{w1w2}}, \overrightarrow{\mathbf{w2}})}$ and $b = 1 - a$. These weights differ from compound to compound.

5.3.3 Compositionality Judgment

To judge if a compound is compositional or non-compositional, previous approaches (see Section 5.2) base their judgment on a single similarity value. As discussed, we base our judgment based on the collective evidences from all the similarity values using a linear equation of the form

$$\begin{aligned}\alpha(\overrightarrow{\mathbf{w1^{w2}}}, \overrightarrow{\mathbf{w2^{w1}}}) = a_0 &+ a_1.sim(\overrightarrow{\mathbf{w1w2}}, \overrightarrow{\mathbf{w1^{w2}}}) \\ &+ a_2.sim(\overrightarrow{\mathbf{w1w2}}, \overrightarrow{\mathbf{w2^{w1}}}) \\ &+ a_3.sim(\overrightarrow{\mathbf{w1w2}}, a\ \overrightarrow{\mathbf{w1^{w2}}} + b\ \overrightarrow{\mathbf{w2^{w1}}}) \\ &+ a_4.sim(\overrightarrow{\mathbf{w1w2}}, \overrightarrow{\mathbf{w1^{w2}}}\overrightarrow{\mathbf{w2^{w1}}})\end{aligned} \tag{5.1}$$

where the value of α denotes the compositionality score. The range of α is in between 0-100. If $\alpha \leq 34$, the compound is treated as non-compositional, $34 < \alpha < 67$ as medium compositional and $\alpha \geq 67$ as highly compositional. The parameters a_i's are estimated using ordinary least square regression by training over the training data released in the shared task [Biemann and Giesbrecht, 2011]. For the three categories – adjective-noun, verb-object and subject-verb – the parameters are estimated separately.

Note that if $a_1 = a_2 = a_4 = 0$, the model bases its judgment only on addition. Similarly if $a_1 = a_2 = a_3 = 0$, the model bases its judgment only on multiplication.

We also experimented with combinations such as $\alpha(\overrightarrow{\mathbf{w1^{w2}}}, \overrightarrow{\mathbf{w2}})$ and $\alpha(\overrightarrow{\mathbf{w1}}, \overrightarrow{\mathbf{w2^{w1}}})$ i.e. using refined vector for one of the constituent word and the unrefined prototype vector for the other constituent word.

5.3.4 [Biemann and Giesbrecht, 2011] Shared Task Dataset

Biemann and Giesbrecht [2011] prepared a compositionality dataset using Mechanical Turk. The dataset contains 144 ADJ-NN, 74 V-SUBJ, 133 V-OBJ phrases marked with compositionality score. Each phrase is annotated by 4 Amazon Mechanical Turkers. Each turker is presented with 4-5 random sentences containing a target phrase and the annotator annotates the compositionality score in the range of 0-10. Scores from all the annotators are averaged for each phrase and the final score is normalized in the range of 0-100.

In addition, phrases are divided into three classes based on the compositionality score - high in compositional (score $>$ 75), medium in compositional (62 $>$ score $>$ 38) and low in compositional (25 $>$ score $>$ 0). These phrases are labeled with corresponding class labels, also called as coarse labels. There are 102 ADJ-NN, 56 V-SUBJ, 96 V-OBJ phrases which have scores in the defined score range. All other phrases are not classified and are not included in evaluation for coarse-grained labels.

The final dataset is divided into 40% training, 10% validation and 50% test datasets.

5.3.5 Selecting the best model

To participate in the shared task, we have selected the best performing model by evaluating the models on the validation data released in the shared task. Table 5.1 displays the results on the validation data. The average point difference (APD) is calculated by taking the average of the difference in a model's score α and the gold score annotated by humans, over all compounds. The lower the APD score, the better is the model. Table 5.1 also displays the overall accuracy of coarse grained labels – low, medium and high.

Best performance for verb(v)-object(o) compounds is found for the combination $\alpha(\overrightarrow{\mathbf{v^o}}, \overrightarrow{\mathbf{o^v}})$ of Equation 5.1. For subject(s)-verb(v) compounds, it is for $\alpha(\overrightarrow{\mathbf{s^v}}, \overrightarrow{\mathbf{v^s}})$ and $a_3 = a_4 = 0$. For adjective(j)-noun(n) compounds, it is $\alpha(\overrightarrow{\mathbf{j^n}}, \overrightarrow{\mathbf{n}})$. We are not certain of the reason for this difference, perhaps there may be less ambiguity of words within specific grammatical relationships

Model	**APD**	**Acc.**
Dyn-Best	13.09	88.0
Sta-Addn	15.42	76.0
Sta-Mult	17.52	80.0
Sta-Best	15.12	80.0

Table 5.1: Average Point Difference (APD) and Coarse Grained Accuracy (Acc.) of Compositionality Judgments on validation data

or it may be simply due to the actual compounds in those categories. In ADJ-NN, it may be the case that ADJ is not a good disambiguator of NN whereas NN is a good disambiguator of ADJ. We combined the outputs of these category-specific models to build the best model *Dyn-Best*.

For comparison, results of standard (static) models prototype addition *(Sta-Addn)* and prototype-multiplication *(Sta-Mult)* are also displayed in Table 5.1. *Sta-Addn* can be represented as $\alpha(\overrightarrow{\mathbf{w1}}, \overrightarrow{\mathbf{w2}})$ with $a_1 = a_2 = a_4 = 0$. *Sta-Mult* can be represented as $\alpha(\overrightarrow{\mathbf{w1}}, \overrightarrow{\mathbf{w2}})$ with $a_1 = a_2 = a_3 = 0$. *Sta-Best* is the best performing model in prototype-based modeling. It is found to be $\alpha(\overrightarrow{\mathbf{w1}}, \overrightarrow{\mathbf{w2}})$. (Note: Depending upon the compound type, some of the a_i's in *Sta-Best* may be 0).

Overall, dynamic prototype-based modeling excelled in both the evaluations, average point difference and coarse-grained label accuracies. The systems *Dyn-Best* and *Sta-Best* were submitted for public evaluation in the shared task. All the model parameters were estimated by regression on the task's training data separately for the 3 compound types as described in Section 5.3.3 in order to maximize performance.

5.4 Shared Task Results

Table 5.2 displays Spearman ρ and Kendalls τ correlation scores of all the models. TotPrd stands for the total number of predictions. Rand-Base is the baseline system which randomly assigns a compositionality score for a compound. Our model Dyn-Best was the best performing system compared to all other systems in this evaluation criteria. SharedTaskNextBest is the next best performing system apart from our models. Due to lemmatization errors in the test data, our models could only predict judgments for 169

	TotPrd	**Spearman ρ**	**Kendalls τ**
Rand-Base	174	0.02	0.02
Dyn-Best	169	**0.35**	**0.24**
Sta-Best	169	0.33	0.23
SharedTaskNextBest	174	0.33	0.23

Table 5.2: Correlation Scores

	All	**ADJ-NN**	**V-SUBJ**	**V-OBJ**
Rand-Base	32.82	34.57	29.83	32.34
Zero-Base	23.42	24.67	17.03	25.47
Dyn-Best	**16.51**	15.19	15.72	18.6
Sta-Best	16.79	14.62	18.89	18.31
SharedTaskBest	**16.19**	14.93	21.64	14.66

Table 5.3: Average Point Difference Scores

out of 174 compounds.

Table 5.3 displays average point difference scores. Zero-Base is a baseline system which assigns a score of 50 to all compounds. SharedTaskBest is the overall best performing system. Dyn-Best was ranked second best among all the systems. For ADJ-NN and V-SUBJ compounds, the best performing systems in the shared task are Sta-Best and Dyn-Best respectively. Our models did less well on V-OBJ compounds, and exploring the reasons for this will be our future work.

Table 5.4 displays coarse grained scores. As above, similar behaviour is observed for coarse grained accuracies. Most-Freq-Base is the baseline system which assigns the most frequent coarse-grained label for a compound based on its type (ADJ-NN, V-SUBJ, V-OBJ) as observed in training data. Most-Freq-Base outperforms all other systems.

	All	**ADJ-NN**	**V-SUBJ**	**V-OBJ**
Rand-Base	0.297	0.288	0.308	0.30
Zero-Base	0.356	0.288	0.654	0.25
Most-Freq-Base	0.593	0.673	0.346	0.65
Dyn-Best	**0.576**	0.692	0.5	0.475
Sta-Best	0.567	0.731	0.346	0.5
SharedTaskBest	**0.585**	0.654	0.385	0.625

Table 5.4: Coarse Grained Accuracy

5.5 Summary

In this chapter, we examined the effect of polysemy in compositional models of compositionality detection. We experimented with dynamic prototypes to eliminate noisy contexts which arrive due to polysemy. Overall, the performance of the dynamic prototype-based models of compositionality detection is found to be superior to standard (static) prototype-based models. This shows 1). polysemy is a problem for compositionality detection 2). Dynamic Prototypes perform better than their static counterparts.

In the next chapter, we will evaluate dynamic prototypes on our final evaluation metric, namely the GPV metric (Section 2.3).

Chapter 6

Dynamic Prototypes on an Internal Evaluation Task

In the previous chapters, we discussed the effect of polysemy in composition models, and demonstrated *dynamic prototypes* are better at dealing with polysemy by evaluating compositional models on external tasks such as phrasal similarity and compositionality detection. In this chapter, we will evaluate dynamic prototype based composition models on an internal evaluation task, the similarity with *Gold Phrasal Vectors* (*GPV metric*) introduced in section 2.3.

In the following sections we will describe our experimental setup, evaluation dataset, and present our evaluation results.

6.1 Experimental Setup

We use the vector space model described in section 1.4.2 in all our experiments described in this chapter. Let *w1w2* be a compound with constituent words *w1* and *w2*. The prototype vectors $\overrightarrow{\mathbf{w1}}$, $\overrightarrow{\mathbf{w2}}$, $\overrightarrow{\mathbf{w1w2}}$ are built using corpus instances of *w1*, *w2* and *w1w2* respectively. $\overrightarrow{\mathbf{w1w2}}$ is the gold phrasal vector. The dynamic prototypes $\overrightarrow{\mathbf{w1^{w2}}}$ and $\overrightarrow{\mathbf{w2^{w1}}}$ are also built as described in section 4.2.2.1.

Using the composition models, simple addition (ADDITION) and simple

multiplication (MULTIPLICATION), described in section 4.3, we construct the composed vectors $\overrightarrow{\oplus(\mathbf{w1w2})}$ of *w1w2*. We aim to test two models - a static prototype model (STATIC) and a dynamic prototype model (DYNAMIC).

In the model STATIC, we build composed vector $\overrightarrow{\oplus(\mathbf{w1w2})}$ using $\overrightarrow{\mathbf{w1}} \oplus \overrightarrow{\mathbf{w2}}$ from the prototypes $\overrightarrow{\mathbf{w1}}$ and $\overrightarrow{\mathbf{w2}}$, where $\oplus$ represents either the composition operator addition or multiplication. Similarly in the model DYNAMIC, we build $\overrightarrow{\oplus(\mathbf{w1w2})}$ using $\overrightarrow{\mathbf{w1^{w2}}} \oplus \overrightarrow{\mathbf{w2^{w1}}}$ from the dynamic prototypes $\overrightarrow{\mathbf{w1^{w2}}}$ and $\overrightarrow{\mathbf{w2^{w1}}}$.

For each model, we measure its GPV metric (Equation 6.1) over n phrases $P1, P2, P3 \ldots, Pn$. Higher the value, better is the compositional model.

$$\frac{\sum_1^n sim(\ \overrightarrow{\mathbf{Pi}},\ \overrightarrow{\oplus(\mathbf{Pi})}\ )}{n} \tag{6.1}$$

6.2 Dataset

It is unclear if compositional models are expected to compose the semantics of non-compositional and medium compositional phrases. Instead of dealing with all types of phrases, for this task, we only deal with compositional phrases to make credible observations from the results. In Chapter 3 we developed a new compound noun dataset annotated with compositionality scores in the range 0-5 (refer to Table 3.2). We only take the compounds whose phrasal composition score is greater than 3 and treat them as compositional compounds (all the compounds above and including *face value* in Table 3.2). The filtered dataset consists of 37 compound nouns. We evaluate above models on this filtered dataset.

6.3 Results and Discussion

Table 6.1 displays the evaluation results of models STATIC and DYNAMIC on the above dataset. In the model DYNAMIC, we have experimented different percent exemplar activations (refer Section 4.2.2.1 and Section 4.4.3).

	ADDITION	**MULTIPLICATION**
Model STATIC	0.4639	0.4697
Model DYNAMIC		
Top 2 % exemplars	0.4844	0.3451
Top 5 % exemplars	0.5162	0.3777
Top 10 % exemplars	0.5313	0.4005
Top 20 % exemplars	**0.5328**	0.4250
Top 50 % exemplars	0.5235	0.4439
Top 70 % exemplars	0.5108	0.4497

Table 6.1: GPV metric results

With respect to the composition function ADDITION, model DYNAMIC outperforms model STATIC with 14.85% additional improvement, showing that dynamic prototypes are better than static prototypes for semantic composition. The models perform their best at a= 0.45 and b= 0.55, where a and b are the scalars in the addition composition function $a\ \overrightarrow{\mathbf{w1}} + b\ \overrightarrow{\mathbf{w2}}$. In the model DYNAMIC, the best performance is achieved by selecting the top 20% exemplars and the next best being top 10%. This observation is slightly similar to as observed in Chapter 4 for the task of phrasal similarity. With increase in number of exemplars after a certain point ($\approx$ 20% activation), the performance starts decreasing, which means irrelevant exemplars start adding noisy contexts.

With respect to composition function MULTIPLICATION, the results are rather unexpected. STATIC model performs better than DYNAMIC. With increase in number of exemplars, the performance increases. We think the reason behind lower performance of DYNAMIC model is because of the cosine similarity metric. In the dynamic prototypes ($\overrightarrow{\mathbf{w1}^{\mathbf{w2}}}$ and $\overrightarrow{\mathbf{w2}^{\mathbf{w1}}}$) most vector components have zero frequency compared to static prototypes ($\overrightarrow{\mathbf{w1}}$ and $\overrightarrow{\mathbf{w2}}$). When an operator like vector multiplication is performed between two dynamic prototypes, many more components will become zero in the resultant composition vector $\overrightarrow{\mathbf{w1}^{\mathbf{w2}}}\overrightarrow{\mathbf{w2}^{\mathbf{w1}}}$. On the other hand, the co-occurrence vector of the compound $\overrightarrow{\mathbf{w1w2}}$ and the composed vector $\overrightarrow{\mathbf{w1}}\overrightarrow{\mathbf{w2}}$ have many non-zero components (due to noise). In the study of Weeds *et al.* [2004], cosine similarity is known to prefer collocates which have high frequencies (which equivalently means collocates having many non-zero vector components), which might be the reason of $\overrightarrow{\mathbf{w1w2}}$ preferring $\overrightarrow{\mathbf{w1}}\overrightarrow{\mathbf{w2}}$ over $\overrightarrow{\mathbf{w1}^{\mathbf{w2}}}\overrightarrow{\mathbf{w2}^{\mathbf{w1}}}$.

6.4 Summary

In this chapter, we experimented with dynamic prototypes in comparison with standard prototypes on an internal evaluation task of semantic composition. Overall, the performance of additive dynamic prototype-model outperformed all other models. Therefore, dynamic prototypes are the winner in this evaluation metric.

Chapter 7

Discussion

The dominant theme of the thesis has been to exlpore the hypothesis – *sense disambiguation benefits compositional distributional semantic (CDS) models.* Our experimental results (Chapters 4, 5 and 6) on three compositionality based evaluation tasks (Chapter 2) confirm that sense disambiguation improves the performance of CDS models.

We first started off with two simple compositional models (Chapter 3) – constituent-based (inspired by our study on human annotations) and composition function based models (inspired from existing research) – to assess the performance of CDS models without any sense disambiguation. Based on the relative performance between the two models, we have chosen composition function models as our main framework to test our hypothesis.

We have proposed two types of sense disambiguation methods (Chapter 4), static multi prototype-based and dynamic prototype-based . On the task of phrasal similarity, compositional models using dynamic prototypes are found to outperform standard (single) prototypes which do not perform any sense disambiguation. This strengthens our hypothesis that sense disambiguation leads to better composition. Surprisingly, the performance of static multi prototypes is found to be worse than standard prototypes, showing that traditional methods of sense disambiguation are not suitable for compositional models (as is the case with other tasks [Navigli, 2009]). In the later experiments, we only evaluated dynamic prototypes in comparison with standard prototypes, discarding static multi prototypes.

Our additional experiments on compositionality detection (Chapter 5) and an internal evaluation task (Chapter 6) reveal that dynamic prototypes are better than standard prototypes which do not perform any disambiguation, strengthening our hypothesis further.

We therefore conclude the following statements from this thesis.

- Sense disambiguation benefits compositional distributional models.
- Dynamic prototypes are better than static (multi and single) prototypes for compositional tasks.

In the following sections, based on the observations from our experiments, we discuss subtle aspects of dynamic prototypes in comparison with static prototypes, the composition function simple addition in comparison with simple multiplication, and external evaluation tasks in comparison with internal evaluation tasks.

7.1 Dynamic vs Static Prototypes

Why do Dynamic Prototypes perform better than Static Prototypes on different evaluation tasks? Perhaps it could be due to the difference in the features activated by different methods. A dynamic prototype of a word in a given context, activate all the fine grained features that are common between the word and the context along with the features of the word that co-occur with the commonly shared features. In the static single prototypes (standard prototypes) all the features of a word are used for composition and the composition model has to deal with noisy features by itself. While static multi prototypes try to deal with noise, they overdo it by throwing away fine grained features that might be relevant. In static multi prototypes, each prototype represents a spectrum of meaning (a cluster of instances) which in turn represents certain dominant features. When one of these prototypes is selected for composition, several other features, though important in the given context but are not represented by the cluster, are left behind. A better way would be to activate many clusters relevant to the given context. However, most common clustering algorithms are hard classification methods based on coarse-grained features whereas from dynamic prototypes we understand that composition models benefit from fine grained features.

Recently, Reisinger and Mooney [2011] experimented with soft clustering methods based on fine grained features and evaluated them on various distributional tasks. In this method, every pair of clusters may share a different subset of features and a cluster represents a fine grained usage of a word. In the future, it might be fruitful to perform composition using static multi prototypes built using soft clustering techniques combined with multiple prototype activation. It would also be interesting to study the performance of dynamic prototypes represented by coarse grained topics obtained from techniques like LDA [Blei *et al.*, 2003].

In the current work, we focused only on phrases with length two. Also, our methods are not highly sensitive to word order (though slightly sensitive in the implementation). A challenging direction of this work would be to build dynamic prototypes when the context size is greater than one, to account for word order, and to use structured vector space models. The option of building dynamic prototypes from standard prototypes directly without using exemplars can also be explored.

7.2 Simple Addition vs Simple Multiplication

In all our experiments, we have used simple addition and simple multiplication composition functions. While simple addition performed better on compositionality detection (Table 3.6) and internal evaluation (Table 6.1) tasks, simple multiplication performed better on phrasal similarity (Table 4.2). Similar results are observed by earlier researchers. Giesbrecht [2009] and Guevara [2011] observed addition to perform better than multiplication on compositionality detection and internal evaluation tasks respectively, whereas Mitchell and Lapata [2010] observed multiplication to perform better on phrasal similarity. It is hard to decide a winner among addition and multiplication. From our results, we observe that additive models are superior to multiplicative models if sense-based prototypes are used for composition, else multiplicative models are superior. The composition function multiplication acts as a disambiguation operator, and if sense based prototypes are used for composition, the operation becomes severe filtering out relevant features too.

We preferred to experiment with addition and multiplication over complex composition operators [Widdows, 2008] since addition and multiplication are found to competitive [Giesbrecht, 2009] and are also easy to interpret. Some composition functions inherently include noise removal (sense disambiguation) e.g. [Erk and Padó, 2008] use selectional preferences of words participating in the composition. With our approach of dynamic prototypes, composition functions have an advantage of not worrying much about polysemy of participants, and thus can concentrate on the main composition task.

7.3 External vs Internal Evaluation tasks

In this thesis, all our observations are based on three compositionality based evaluation tasks (Chapter 2), phrasal similarity and compositionality detection tasks being external, and GPV metric being internal evaluation task. While external tasks require human annotation, internal task requires large data. We think external tasks are more reliable than internal task since internal task requires gold phrasal vectors which are difficult to obtain. Gold phrasal vectors may not represent the true semantics of phrases (as is the case observed with compound prototype model in Section 4.4.3). As the length of the phrase increases, data becomes sparser and sparser making it infeasible to build gold phrasal vectors.

7.4 Conclusions

The main contribution of the thesis are the experiments to strengthen the hypothesis that *sense disambiguation benefits compositional distributional models*. We introduced the *Dynamic Prototypes* which represent context sensitive meaning of words. Dynamic prototypes are found to be better than conventional representation of word senses. Using dynamic prototypes we were able to build better systems for compositionality detection and phrasal similarity compared to the existing systems evaluated in this thesis. The other main contribution of our work is the compositionality dataset, which exhibits the continuum of compositionality without any bias to spe-

cific compositional classes. Our study on this dataset revealed interesting facts about the relation between the semantics of constituent words and phrases.

Until now, distributional models of compositional semantics focused on modeling phrasal semantics. Bigger challenges are posed at higher levels like sentential and document level semantics. We believe that sense disambiguation using dynamic representation of meaning is a fruitful direction to look at for building these systems.

Bibliography

[Agirre and Soroa, 2007] Eneko Agirre and Aitor Soroa. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 7–12, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[Baldridge and Kruijff, 2002] Jason Baldridge and Geert-Jan M. Kruijff. Coupling ccg and hybrid logic dependency semantics. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 319–326, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[Baldwin *et al.*, 2003] Timothy Baldwin, Colin Bannard, Takaaki Tanaka, and Dominic Widdows. An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment - Volume 18*, MWE '03, pages 89–96, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[Bannard *et al.*, 2003] Colin Bannard, Timothy Baldwin, and Alex Lascarides. A statistical approach to the semantics of verb-particles. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment - Volume 18*, MWE '03, pages 65–72, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[Biemann and Giesbrecht, 2011] Chris Biemann and Eugenie Giesbrecht. Distributional semantics and compositionality 2011: Shared task description and results. In *Proceedings of DISCo-2011 in conjunction with ACL 2011*, 2011.

[Biemann, 2006] Chris Biemann. Chinese Whispers - An Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. In *Proceedings of TextGraphs*, pages 73–80, New York, USA, 2006.

[Blei *et al.*, 2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[Chen and Mooney, 2008] David L. Chen and Raymond J. Mooney. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 128–135, New York, NY, USA, 2008. ACM.

[Clark and Pulman, 2007] Stephen Clark and Stephen Pulman. Combining symbolic and distributional models of meaning. In *Proceedings of the AAAI Spring Symposium on Quantum Interaction, Stanford, CA, 2007*, pages 52–55, 2007.

[Cook *et al.*, 2007] Paul Cook, Afsaneh Fazly, and Suzanne Stevenson. Pulling their weight: exploiting syntactic forms for the automatic identification of idiomatic expressions in context. In *Proceedings of the Workshop on a Broader Perspective on Multiword Expressions*, MWE '07, pages 41–48, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[Copestake *et al.*, 2005] Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan Sag. Minimal Recursion Semantics: An Introduction. *Research on Language and Computation*, 3(4):281–332–332, December 2005.

[Dice, 1945] Lee R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):pp. 297–302, 1945.

[Dunning, 1993] Ted Dunning. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[Erk and Padó, 2008] Katrin Erk and Sebastian Padó. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 897–906, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[Erk and Padó, 2010] Katrin Erk and Sebastian Padó. Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 92–97, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[Fazly *et al.*, 2009] Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. Unsupervised type and token identification of idiomatic expressions. *Comput. Linguist.*, 35:61–103, March 2009.

[Fellbaum, 1998] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

[Ferraresi *et al.*, 2008] Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. Introducing and evaluating ukWaC, a very large web-derived corpus of english. In *Proceedings of the WAC4 Workshop at LREC 2008*, Marrakesh, Morocco, 2008.

[Firth, 1957] John R. Firth. A Synopsis of Linguistic Theory, 1930-1955. *Studies in Linguistic Analysis*, pages 1–32, 1957.

[Ge and Mooney, 2005] Ruifang Ge and Raymond J. Mooney. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 9–16, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[Gibbs, 1989] Raymond W. Gibbs. Understanding and literal meaning. *Cognitive Science*, 13(2):243–251, 1989.

[Giesbrecht, 2009] Eugenie Giesbrecht. In search of semantic compositionality in vector spaces. In *Proceedings of the 17th International Conference on Conceptual Structures: Conceptual Structures: Leveraging Semantic Technologies*, ICCS '09, pages 173–184, Berlin, Heidelberg, 2009. Springer-Verlag.

[Grefenstette and Sadrzadeh, 2011] Edward Grefenstette and Mehrnoosh Sadrzadeh. Experimental support for a categorical compositional distributional model of meaning. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011.

[Guevara, 2010] Emiliano Guevara. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '10, pages 33–37, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[Guevara, 2011] Emiliano Raul Guevara. Computing semantic compositionality in distributional semantics. In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '2011, 2011.

[Harris, 1954] Zellig S. Harris. Distributional structure. *Word*, 10:146–162, 1954.

[Kate and Mooney, 2007] Rohit J. Kate and Raymond J. Mooney. Learning language semantics from ambiguous supervision. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 1*, pages 895–900. AAAI Press, 2007.

[Katz and Giesbrecht, 2006] Graham Katz and Eugenie Giesbrecht. Automatic identification of non-compositional multi-word expressions using latent semantic analysis. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, MWE '06, pages 12–19, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[Kilgarriff *et al.*, 2004] Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. The sketch engine. In *Proceedings of EURALEX 2004*, 2004.

[Kilgarriff, 1997] Adam Kilgarriff. I don't believe in word senses. In *Computers and the Humanities, 31(2):91-113.*, 1997.

[Klapaftis and Manandhar, 2010a] Ioannis Klapaftis and Suresh Manandhar. Word sense induction & disambiguation using hierarchical random graphs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 745–755, Cambridge, MA, October 2010. Association for Computational Linguistics.

[Klapaftis and Manandhar, 2010b] Ioannis P. Klapaftis and Suresh Manandhar. Taxonomy Learning Using Word Sense Induction. In *Proceedings*

of NAACL-HLT-2010, pages 82–90, Los Angeles, California, June 2010. ACL.

[Korkontzelos and Manandhar, 2009] Ioannis Korkontzelos and Suresh Manandhar. Detecting compositionality in multi-word expressions. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 65–68, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[Kwiatkowski *et al.*, 2011] Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1512–1523, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[Liang *et al.*, 2011] Percy Liang, Michael I. Jordan, and Dan Klein. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 590–599, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[Lin, 1999] Dekang Lin. Automatic identification of non-compositional phrases. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 317–324, Stroudsburg, PA, USA, 1999. Association for Computational Linguistics.

[McCarthy *et al.*, 2003] Diana McCarthy, Bill Keller, and John Carroll. Detecting a continuum of compositionality in phrasal verbs. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment - Volume 18*, MWE '03, pages 73–80, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[McCarthy *et al.*, 2007] Diana McCarthy, Sriram Venkatapathy, and Aravind K. Joshi. Detecting compositionality of verb-object combinations using selectional preferences. In *EMNLP-CoNLL*, pages 369–379, 2007.

[Mitchell and Lapata, 2008] Jeff Mitchell and Mirella Lapata. Vector-based Models of Semantic Composition. In *Proceedings of ACL-08: HLT*, pages

236–244, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[Mitchell and Lapata, 2010] Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 2010.

[Montague, 1970] R. Montague. Universal grammar. *Theoria*, 36(3):373–398, 1970.

[Montague, 1973] Richard Montague. The Proper Treatment of Quantification in Ordinary English. pages 221–242, 1973.

[Navigli and Crisafulli, 2010] Roberto Navigli and Giuseppe Crisafulli. Inducing word senses to improve web search result clustering. In *EMNLP*, pages 116–126, 2010.

[Navigli, 2009] Roberto Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41:10:1–10:69, February 2009.

[Nunberg *et al.*, 1994] Geoffrey Nunberg, Thomas Wasow, and Ivan A. Sag. Idioms. *Language*, 70(3):491–539, 1994.

[Pado and Lapata, 2007] Sebastian Pado and Mirella Lapata. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199, 2007.

[Padó *et al.*, 2009] Ulrike Padó, Matthew W. Crocker, and Frank Keller. A probabilistic model of semantic plausibility in sentence processing. *Cognitive Science*, 33(5):794–838, 2009.

[Partee, 1995] Barbara Partee. Lexical semantics and compositionality. *L. Gleitman and M. Liberman (eds.) Language, which is Volume 1 of D. Osherson (ed.) An Invitation to Cognitive Science (2nd Edition)*, pages 311–360, 1995.

[Pedersen, 2011] Ted Pedersen. Identifying collocations to measure compositionality : Shared task system description. In *Proceedings of DISCo-2011 in conjunction with ACL 2011*, 2011.

[Pelletier, 1994] Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13:11–24, 1994. 10.1007/BF00763644.

[Reisinger and Mooney, 2010] Joseph Reisinger and Raymond J. Mooney. Multi-prototype vector-space models of word meaning. In *HLT-NAACL*, pages 109–117, 2010.

[Reisinger and Mooney, 2011] Joseph Reisinger and Raymond Mooney. Cross-cutting models of lexical semantics. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1405–1415, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[Rychlý and Kilgarriff, 2007] Pavel Rychlý and Adam Kilgarriff. An efficient algorithm for building a distributional thesaurus (and other sketch engine developments). In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 41–44, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[Sag *et al.*, 2002] Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann A. Copestake, and Dan Flickinger. Multiword expressions: A pain in the neck for nlp. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '02, pages 1–15, London, UK, 2002. Springer-Verlag.

[Sahlgren, 2006] Magnus Sahlgren. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces.* PhD thesis, Stockholm University, 2006.

[Schone and Jurafsky, 2001] Patrick Schone and Daniel Jurafsky. Is knowledge-free induction of multiword unit dictionary headwords a solved problem? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '01, 2001.

[Schütze, 1998] Hinrich Schütze. Automatic Word Sense Discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

[Smith and Medin, 1981] Edward E. Smith and Douglas L. Medin. *Categories and concepts / Edward E. Smith and Douglas L. Medin.* Harvard University Press, Cambridge, Mass. :, 1981.

[Snow *et al.*, 2008] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[Socher *et al.*, 2011] Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2011.

[Sporleder and Li, 2009] Caroline Sporleder and Linlin Li. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 754–762, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[Titone and Connine, 1999] Debra A. Titone and Cynthia M. Connine. On the compositional and noncompositional nature of idiomatic expressions. *Journal of Pragmatics*, 31(12):1655 – 1674, 1999. Literal and Figurative Language.

[Turney and Pantel, 2010] Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *JOURNAL OF ARTIFICIAL INTELLIGENCE RESEARCH*, 37:141, 2010.

[Venkatapathy and Joshi, 2005] Sriram Venkatapathy and Aravind K. Joshi. Measuring the relative compositionality of verb-noun (v-n) collocations by integrating features. In *Proceedings of the joint conference on Human Language Technology and Empirical methods in Natural Language Processing*, pages 899–906, Vancouver, B.C., Canada, 2005.

[Weeds *et al.*, 2004] Julie Weeds, David Weir, and Diana McCarthy. Characterising measures of lexical distributional similarity. In *Proceedings of Coling 2004*, pages 1015–1021, Switzerland, 2004. COLING.

[Widdows, 2008] Dominic Widdows. Semantic vector products: Some initial investigations. In *Second AAAI Symposium on Quantum Interaction, Oxford*, March 2008.