# EXPLOITING LINGUISTIC KNOWLEDGE IN LEXICAL AND COMPOSITIONAL SEMANTIC MODELS

by

Tong Wang

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

# Abstract

Exploiting Linguistic Knowledge in Lexical and Compositional Semantic Models

Tong Wang

Doctor of Philosophy

Graduate Department of Computer Science

University of Toronto

2016

A fundamental principle in distributional semantic models is to use similarity in linguistic environment as a proxy for similarity in meaning. Known as the *distributional hypothesis*, the principle has been successfully applied to many areas in natural language processing. As a proxy, however, it also suffers from critical limitations and exceptions, many of which are the result of the overly simplistic definitions of the linguistic environment. In this thesis, I hypothesize that the quality of distributional models can be improved by carefully incorporating linguistic knowledge into the definition of linguistic environment.

The hypothesis is validated in the context of three closely related semantic relations, namely near-synonymy, lexical similarity, and polysemy. On the lexical level, the definition of linguistic environment is examined under three different distributional frameworks including lexical co-occurrence, syntactic and lexicographic dependency, and taxonomic structures. Firstly, combining kernel methods and lexical level co-occurrence with matrix factorization is shown to be highly effective in capturing the fine-grained nuances among near-synonyms. Secondly, syntactic and lexicographic information is shown to result in notable improvement in lexical embedding learning when evaluated in lexical similarity benchmarks. Thirdly, for taxonomy-based measures of lexical similarity, the intuitions for using structural features such as depth and density are examined and challenged, and the refined definitions are shown to improve correlation between the features and human judgements of similarity as well as performances of similarity measures using these features.

On the compositional level, distributional models of multi-word contexts are also shown to benefit from incorporating syntactic and lexicographic knowledge. Analytically, the use of syntactic teacher-forcing is motivated by derivations of full gradients in long short-term memory units in recurrent neural networks. Empirically, syntactic knowledge helps achieve statistically significant improvement in language modelling and state-of-the-art accuracy in near-synonym lexical choice. Finally, a compositional similarity function is developed to measure the similarity between two sets of random events. Application in polysemy with lexicographic knowledge produces state-of-the-art performance in unsupervised word sense disambiguation.

In loving memory of Jingli Wang.

# Acknowledgements

It's been a long journey, but nevertheless a fun ride. I thank my supervisor Professor Graeme Hirst for giving me the opportunity, the guidance, as well as the ultimate academic freedom to pursue my various research interests. I thank all the members of my supervisory committee, Professor Gerald Penn, Professor Ruslan Salakhutdinov, and Professor Suzanne Stevenson, for their help, support, and encouragement all along. I thank my external examiners Professor Ted Pedersen, Professor Frank Rudzicz, and Professor Sanja Fidler for their valuable time and for the insightful comments and discussions.

I'd also like to express my gratitude to Doctor Xiaodan Zhu and Doctor Afsaneh Fazly for reaching out and helping me at the most challenging times in my program, to Doctor Julian Brooke, Doctor Abdel-rahman Mohamed, and Nona Naderi for the collaborations and the thought-provoking discussions, and to all the fellow graduate students in the Computational Linguistics group for this memorable experience.

Finally, I'd like to thank my grandmother Fanglan Wu for inspiring me to persist when I was tempted to give up. I also thank my wife Yi Zhang for her unconditional support through the many stressful spans over the years. Without either of you, I could not have completed the degree. Thank you.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivations and Outline

Representing and modelling meaning in text is a difficult task but nonetheless a central goal in Natural Language Processing (NLP). A major part of the difficulty lies in the fact that the concept of meaning is highly abstract and complex, which is particularly challenging to represent or reason about computationally. Among the many existing attempts at semantic modelling, a particular class of models is based on the *distributional hypothesis* (DH, Harris, 1954), which states that meaning is at least in correlation with — if not completely defined by — the *linguistic environment*. Essentially, this hypothesis associates the abstract concept of meaning with the concrete and readily quantifiable distribution properties of linguistic elements — an association that constitutes the computational significance of the hypothesis.

The notion of linguistic environment is an integral part in the application of the DH, yet its effectiveness is often grossly undermined when a large number of distributional models rely heavily on *raw linguistic contexts*. For example, lexical-level contexts are often defined as unstructured windows of surrounding string tokens, and various forms of "centroids" derived from simple vector operations are still popular choices for composition functions. Despite their computational efficiency and ease of implementation, these models ignore even the most basic types of linguistic knowledge and thus bear little resemblance to human behaviour in natural language understanding.

Motivated by this observation, a central hypothesis in this thesis is that application of the DH can benefit both conceptually and empirically from incorporating linguistic knowledge into the definition of linguistic environment. Corresponding to the two major parts of the thesis, the hypothesis is applied and studied on both the lexical and the compositional level in the context of three closely-related lexical semantic relations that are central to lexical semantic research.

The following contributions have arisen in the process of validating the hypothesis. Firstly, novel approaches in semantic representation and modelling are proposed in a wide range of lexical and compositional semantic models. Secondly, a variety of linguistic knowledge in the distributional environment is explored and used by the proposed models including syntax, lexical taxonomy, and lexicography. Thirdly, notable empirical improvements are observed with regard to statistical and/or computational efficiency in the proposed models, as demonstrated by the improvements in model performance in several widely adopted NLP benchmarks.

More specifically, the thesis is structured as follows. For the remainder of this chapter, I will first define and motivate the scope of semantic relations that are central to the proposed research. The individual relations of choice will each be reviewed and collectively, their interrelations established and discussed. Secondly, I will define the scope of distributional semantic models used in the thesis and present computational challenges in both general distributional models and in the particular realm of semantic similarity, synonymy, and polysemy studies.

Specific models and evaluations are formulated on both the lexical (Part I) and the compositional (Part II) level. On the lexical level, I will first review the problem of dimension reduction in Chapter 2 and investigate lexical representation learning using matrix factorization techniques. In Chapter 3, I will show that when carefully combined, traditional methods such as *latent semantic analysis* (Landauer and Dumais, 1997) and *support vector machines* (Cortes and Vapnik, 1995) can achieve highly competitive performance in near-synonym lexical choice (Wang and Hirst, 2010). I will then focus on the problem of improving lexical representation learning with linguistic knowledge from dependency parsers and dictionaries (Chapter 4). Empirically, the linguistic information is shown to greatly reduce computational complexity and at the same time produce lexical embeddings with state-of-the-art performances in lexical similarity evaluations (Wang et al., 2015).

Apart from vector space models, lexical taxonomy is another important knowledge source in semantic similarity studies. For certain structural features in semantic taxonomies, I will show that there are crucial discrepancies between their empirical distribution and the intuitions that motivate the use of these features in existing WordNet-based similarity measures. Consequently, new definitions of depth and density are proposed in Chapter 5 to address these issues (Wang and Hirst, 2011). Empirically, the modifications are shown to significantly improve the correlation between these features and the human judgements of similarity as well as the performance of WordNet similarity models using these features (Wu and Palmer, 1994; Jiang and Conrath, 1997).

Beyond the lexical level, Part II focuses on developing and improving composition models by incorporating linguistic knowledge. Specifically, a simple phrasal composition model is shown in Chapter 7 to converge to an additive model when trained on an artificially created phrasal composition dataset. Despite the empirical motivations of the additive model, however, its limitations in the overly simplistic representations of the context are addressed by taking into account syntactic information such as word order and syntactic categories of the context words. In Chapter 8, a recurrent neural network model with long short-term memory (LSTM) units is employed to implement word order capacities with state recurrence. Gradient analysis across multiple time steps provides theoretical motivations for using syntactic information to improve the gating behaviour of LSTM units. Empirically, syntactic knowledge is shown to help the model achieve significant improvements in language modelling and state-of-the-art results in near-synonym lexical choice. Finally in Chapter 9, a compositional similarity measure is developed to measure the similarity between two sets of random events without explicitly relying on vector-space representations. With the help of lexicographic knowledge, the model is applied to the task of unsupervised word sense disambiguation (WSD) and achieves state-of-the-art performance in several WSD benchmarks (Wang and Hirst, 2014).

Part III concludes the thesis and proposes future directions for the major topics covered in the thesis, including embedding learning and semantic composition.

## 1.2 The Scope of Semantic Relations

The distributional models proposed in the thesis are built around three closely-related lexical semantic relations, namely synonymy, polysemy, and similarity. The motivations for the focus on these particular relations are as follows. Firstly, due to its relation to semantic identity, synonymy plays a central role in several fundamental aspects in semantics such as the definition of entities (Quine, 1969). It is also a highly nuanced relation that is, unlike many other tasks in NLP, challenging even for native speakers of a language. Polysemy, on the other hand, is a dual problem to synonymy in that it aims at distinguishing different senses of the same word form. In a broader sense, both relations border on semantic similarity and together, the three relations are of great importance to lexical and compositional semantics.

### 1.2.1 Synonymy and its Variants

By definition, synonymy is the semantic relation between words of the same meaning. It is commonly believed that absolute synonymy does not exist in a language, and even it does, the absolute sameness in meaning would make the relation intellectually and computationally uninteresting since it would entail only the trivial task of string substitution. Nonetheless, there exist several variants of synonymy that do allow semantic variation while maintaining a high degree of semantic similarity. The semantic variation — particularly under the assumption of correlation with variation in their linguistic contexts — makes it possible to build distributional models to study different aspects of the relation, while the high degree of similarity makes the problem computationally challenging.

The discussions in this section focus on two widely studied variants, namely *cognitive synonymy* and *near-synonymy*. In a simplified definition, cognitive synonymy is a relation between words of the same *denotational* meaning (Cruse, 1986). That is, substitution among cognitive synonyms does not alter the truth condition of an utterance. Consequently, unlike absolute synonymy, cognitive synonymy allows semantic variation along several semantic dimensions including selectional or collocational constraints, dialectal differences, registers, etc.

In practice, however, the requirement of denotational equivalence can still be quite restrictive. Near-synonymy is the result of relaxing this requirement. A good starting point to define and understand the concept of near-synonymy is an acceptability test proposed by Cruse (1986), which is based on the observation that the phrase *more exactly* usually collocates with pairs of words whose meanings differ by relatively minor semantic traits. For example, consider the acceptability of the following sentences:

(1) ? John *passed away* — or more exactly, he *died*.

(2) It was a *fog* — or more exactly, a *mist*.

(3) ? I have an *apple* — or more exactly, an *orange*.

Acceptability is high only for near-synonyms (i.e., Sentence (2)) but not for cognitive synonyms or non-synonyms. This is because the phrase *more exactly* is often used to cancel a minor semantic trait by means of correction, which fits the characteristics of near-synonymy but not the other types of semantic relations. Unfortunately, (as is commonly agreed upon) not much progress can be made towards a more precise definition of near-synonymy. Even the above acceptability test is a necessary but insufficient condition for identifying near-synonymy, since the phrase *more exactly* can also indicate the augmentation of minor semantic traits as a way of further specification. This allows other semantic relations such as hypernymy and hyponymy to pass the acceptability test as well:

(4) Research is, by its nature, *cyclical* or, more exactly, *helical*.

Despite the above concerns for the scope of its definition, near-synonymy is nonetheless a challenging and interesting topic in lexical semantics research. The acceptability test is a prime example of how near-synonyms exhibit interesting differences while at the same time without being too different, which is in accordance with some existing definitions such as "almost synonyms, but not quite" (Hirst, 1995). This delicate balance can be regarded as a "sweet-spot" on the scale of similarity since semantic distinctions are maximally highlighted between major and minor semantic traits. In addition, contrasting other types of synonymy, near-synonymy not only acknowledges difference among synonyms, but also expands the types of minor semantic traits to an open class, which can be formulated and studied as first-order objects (DiMarco et al., 1993; Hirst, 1995). Computationally, due to the relaxation on truth condition preservation, near-synonyms exhibit larger semantic distance than cognitive synonyms, which can be directly translated into higher accuracy in distributional models.

### 1.2.2 Polysemy and Similarity

Polysemy is the semantic relation between multiple senses that share the same word form (spelling). It has been an important research topic with sustained popularity in the NLP community (Hirst, 1987; Lesk, 1986; Pedersen and Bruce, 1998; Yarowsky, 1995; Navigli, 2009). Its relevance to synonymy and similarity studies is manifested in several important aspects in meaning. For example, Word Sense Disambiguation (WSD) is the problem of sense classification given the same word form, whereas near-synonym lexical choice (NSLC) is the problem of word form classification given the same (or highly similar) sense and hence the dual problem of WSD. In addition, synonymy itself is a concept defined on the sense level rather than in the form of string tokens. This fact has been largely ignored in near-synonym lexical choice evaluations, making the data collected for NSLC less representative of the actual meaning of the target near-synonyms. There also exist several NLP tasks that are essentially WSD under different names and formulations. Examples include that of Mitchell and Lapata (2008) (a similarity rating task) and McCarthy and Navigli (2009) (a lexical substitution task), which can be considered WSD tasks since they all involve identifying semantically related words corresponding to different senses of polysemous target words. A detailed review of polysemy and WSD is given in Section 9.2.

Similarity is a more generic notion encompassing several types of semantic relations. The relation between synonymy and similarity largely depends on the definitions of the terms. In fact, some early studies in lexical semantics do not make distinctions between the two (e.g., that of Rubenstein and Goodenough, 1965). Intuitively, the difference between the two relations lies in the difference between *sameness* and *closeness* in meaning, i.e., a matter of degree in semantic overlap. There are many examples of words that can be considered semantically similar without being synonymous (e.g., *mist–fog*, *money–cash*, etc.). Particularly with the rising popularity of distributional approaches to semantics, the definition of similarity is more driven by contextual similarity rather than closeness in denotational meaning.

Lexical similarity has been employed in many research topics and applications including information retrieval (Mandala et al., 1999; Turney, 2001), cognitive science (Gershman and Tenenbaum, 2015), and various lexical semantics studies (Patwardhan et al., 2003; Hill et al., 2014). It is also the building block for many phrase- and sentence-level semantic models (Mitchell and Lapata, 2010; Gershman and Tenenbaum, 2015). Especially with recent developments in distributed lexical representations (Mikolov et al., 2013; Pennington et al., 2014), lexical similarity has become an important benchmark to evaluate the quality of lexical representations. In the neural-network-based NLP literature, the relation has also been successfully applied in sentence similarity (Kalchbrenner et al., 2014; Le and Mikolov, 2014), language modelling (Bengio et al., 2003), and machine translation (Hill et al., 2014).

Contextual similarity has been established as a necessary condition for semantic similarity by several existing studies (Harris, 1954; Rubenstein and Goodenough, 1965). However, it does not entail semantic similarity. For example, antonyms, among many other semantically related words, can share highly similar linguistic environments. With similar contextual distributions, these words are often referred to as being *semantically related*. Traditionally, relatedness is defined among words whose meanings follow one of a prescribed set of semantic relations (e.g., hypernymy, meronymy, etc.). More recently, its definition has been shifting towards distributional similarity and extending to word pairs such as *stock–market*, *computer–software* (Finkelstein et al., 2001), etc.

It is rather difficult to distinguish between similarity and relatedness with distributional approaches. Many distributional similarity models assign a high degree similarity to antonyms and semantically related words (Pereira et al., 1993; Lin, 1998). Existing studies have been trying to address these issues. For example, Lin et al. (2003) used a rule-based method to identify and exclude semantically incompatible words as well as bilingual dictionaries to identify synonyms. Levy and Goldberg (2014) and Wang et al. (2015) used syntactic knowledge to improve semantic similarity metrics over relatedness in lexical embedding learning. Hill et al. (2015b) provided a lexical similarity dataset that makes explicit distinctions between semantic similarity and relatedness. Nonetheless, distinguishing the two types of relations remains a challenging task in the application of the distributional hypothesis.

## 1.3 Distributional Semantic Models

### 1.3.1 Definition

In the context of this thesis, the term *distributional* is adopted as a narrow reference to the class of semantic models that uses the distributional hypothesis (DH) as the underlying assumption for modelling lexical and compositional meaning. The scope of this definition intends to set the proposed models apart from other contrasting approaches of semantic representation and models such as the following:

**Lexicographic representations** Meaning is represented by the hand-crafted dictionary definitions built upon a lexicographer's studies and expertise. The representation is often in the form of natural language texts, and it can be either declarative as in a monolingual dictionary (Webster, 1971), comparative as in a bilingual dictionary, or contrastive as in a synonym dictionary (Hayakawa, 1994).

**Graphic representations** Meaning is represented by a node in a semantic graph and in particular expressed by its structural relations to other nodes in the network. A variety of network structures are allowed including hierarchical structures such as WordNet (Fellbaum, 1998) and categorical structures such as a thesaurus (Roget, 1911).

**Semantic specification** Meaning is represented by semantic specification of membership to a set of prescribed semantic dimensions, such as the result of content analysis (Stone et al., 1966) or semantic decomposition (Gao, 2001).

Note that many of the approaches mentioned above are implicitly related to each other and thus by no means mutually exclusive. For example, definition texts from a monolingual dictionary can be used collectively to derive distributional contexts for lexical representations (Section 4.3.6). Taxonomy-based similarity measures are also shown to benefit from refining the definition of structural features motivated by their natural distributions (Chapter 5).

## 1.3.2 Computational Challenges

### Distributional Representations and Modelling

Computational models are usually designed to capture patterns in the functional mapping between the input space (e.g., texts) and the output space (e.g., textual meaning). In a supervised setting for example, the modelling process is implemented by exposing the model to a collection of examples consisting of input-output pairs. The goal is to *train* the model to be able to accurately predict an output given an input that the model has not been previously exposed to.

Many distributional semantic models follow this general framework of learning. For instance, lexical similarity (Rubenstein and Goodenough, 1965; Miller and Charles, 1991) is often formulated as a regression task to model the mapping $\mathbb{V}^2 \mapsto \mathbb{R}$, where $\mathbb{V}$ is the lexical space (i.e., vocabulary), and $\mathbb{R}$ is the 1-dimension real space.[1] In other words, the input consists of pairs of word tokens, and the model tries to assign a real-valued scalar indicating the semantic similarity between a given input word pair. WSD on the other hand is usually a classification problem $\mathbb{V}^* \mapsto \mathbb{S}$ that classifies the sense of a polysemous word to one of several senses in a predefined sense repository $\mathbb{S}$ given a context of variable length. NSLC has a similar setup, with the output space being a set of near-synonym candidates instead of a sense repository.

The first challenge in the modelling process is to construct numerical representations for lexical items that are computationally and statistically efficient, so that (1) the resulting model is computationally tractable, and (2) the complexity of the data does not surpass the capacity of the model. To achieve either type of efficiency involves careful choice of the model architecture, novel ways of leveraging external knowledge to facilitate learning, and quite often, close examination of the nature of the data. Many examples are provided in later sections to show how these practices influence model quality.

Once a set of efficient lexical representations is obtained (with efficiency being scientifically assessable), another challenge is composition, i.e., how to use the lexical representations in $\mathbb{V}$ to construct representations in $\mathbb{V}^n$ (which is the actual input space in all three tasks mentioned above).[2] Many efforts have been made towards semantic composition, but simple baselines (e.g., that of Mitchell and Lapata, 2010) are surprisingly difficult to outperform. With the recent popularity of artificial neural network models (in particular, recurrent neural network (RNN) models) in compositionality research (Tai et al., 2015; İrsoy and Cardie, 2014; Zhu et al., 2015), it is unclear how these models compare in the context of synonymy and similarity.

Finally, in one way or another, virtually all NLP models leverage external linguistic knowledge, and the way that this knowledge is used is itself a crucial part of the challenge in distribu-

[1]The exponential on $\mathbb{V}$ is a short-hand for a successive application of the Cartesian product between the lexical space and itself.

[2]This is under the assumption that it is not possible to model the $\mathbb{V}^n$ space without going through the $\mathbb{V}$ (lexical) space. Some recent studies attempt to model $\mathbb{V}^n$-space representations jointly with lexical representations (Le and Mikolov, 2014; Kiros et al., 2015). Although the $\mathbb{V}^n$-space representations are updated separately in training, the learning signal nonetheless comes from some form of composition of the lexical representations.

tional models. This topic is of particular relevance in synonymy and similarity since, given the level of semantic granularity involved, it can potentially benefit the most from incorporating external knowledge sources into the modelling process.

**Challenges in Lexical Similarity**

Distinguishing near-synonyms — or similar words in general — is challenging even for native speakers of a language (DiMarco et al., 1993; Hirst, 1995). Empirical evidence suggests that, compared to many lexical semantic tasks, human agreement in a Cloze-style (Taylor, 1953) NSLC task[3] is rather low (78.5%, Inkpen, 2007). Naturally, these difficulties are embodied as computational difficulties in distributional models since semantic sameness or closeness implies a high degree of homogeneity in the output semantic space (e.g., certain aspects of meaning are hardly distinguishable). For example, the current state-of-the-art NSLC performance is about 76.7% (Section 8.3).

Even assuming the availability of an efficient lexical representation and a reliable composition model, evaluation is another fundamental challenge, particularly in synonymy studies. For example, an algorithm is often evaluated by a single number from a NSLC test (Edmonds, 1997), which reports the accuracy of recovering the near-synonym that was originally used in the given context. Although NSLC is an important task by itself, it does not necessarily assess a model's capability in capturing the semantic nuances of near-synonyms and their distinctions. This is because many factors other than semantic considerations (e.g., syntactic, idiomatic, collocational, etc.) influence near-synonym lexical choice. As shown in later chapters, higher NSLC accuracy might result from modelling some of these non-semantic characteristics in synonymy and similarity data. Without a more carefully constructed evaluation scheme, it is not easy to measure what exactly a distributional model is learning in synonymy and similarity studies.

This issue is especially confounded by the less-than-careful way NSLC data is usually

[3]See Section 3.2 for a detailed description of the task.

collected, which can potentially compromise several integral aspects in the definition of synonymy. For example, consider the following NSLC example between the synonyms *difficult* and *hard*:

(5) She was lying on a hard surface.

(6) ? She was lying on a difficult surface.

The reason that Sentence (6) has low acceptability upon lexical substitution is that the sense of *hard* in this context is not synonymous with the sense exhibited by the synonym set {*hard, difficult*}. In lexical semantics, synonym differences in contextual functions are assumed not to be interfered with by irrelevant senses of polysemous word forms (Cruse, 1986). Nonetheless, without sense annotation, Sentence (5) qualifies as a sample in NSLC data collection despite the fact that the word form *hard* appears in a sense irrelevant to the one considered in the near-synonym set. Consequently, even if a lexical choice system is able to make a correct prediction (which is probable given the prevalent collocation of *surface* with *hard* than with *difficult*), it does not necessarily follow that any *synonymy-related* semantic knowledge has been acquired.

For the evaluation of lexical similarity, correlation with human judgement is the most commonly adopted benchmark. However, in many of the most popular datasets (Rubenstein and Goodenough, 1965; Finkelstein et al., 2001), no distinction is made between similarity and relatedness. There is also a pronounced lack of variety with regard to the syntactic categories of the target words used in these datasets. More recently, Hill et al. (2015b) proposed a dataset that directly addresses both issues (Section 4.3.2). Nonetheless, systematic comparisons between the datasets in the context of similarity versus relatedness are rare at best in the literature. Novel approaches in incorporating extrinsic knowledge into the modelling process are also needed in order to take advantage of the syntactic variety available in the new dataset and explore the syntactic dimension in the lexical similarity tasks. Consequently, distinctions are to be explicitly made between similarity and relatedness in the studies presented in this thesis so as to highlight the semantic emphasis in the proposed models.

# Part I

# Lexical Representations

# Chapter 2

# Improving Representation Efficiency in Lexical Distributional Models

The *distributional hypothesis* (DH) states that words that occur in the same contexts tend to have similar meanings (Harris, 1954). The significance of this observation is that contextual similarity can be used as a proxy for semantic similarity. Consequently, the hypothesis is instrumental in distributional semantics due to the implied computational convenience that under this assumption, it is possible to model lexical meaning using the co-occurrence patterns present in the vast amount of textual data.

Despite the conceptual intuitiveness, however, several critical issues remain to be addressed for successful applications of the DH. For example, contextual similarity is a necessary but insufficient condition for semantic similarity. Semantic relatedness (among many other semantic relations) also entails contextual similarity, and distinguishing between similarity and relatedness remains an unresolved challenge for existing distributional models. Another open question of fundamental significance is the definition of *context* (also known as the *semantic space*). In practice, if the DH is to be taken literally, for example, the resulting semantic space would coincide with the lexical space (consisting of the co-occurring words), the size of which can often range from tens of thousands to millions. Sparsity quickly arises as a prominent issue due to the high dimensionality, which in turn causes much inaccuracy in probability estimation especially for rare events. Consequently, a careful design of the semantic space is a crucial step in building distributional models.

In this chapter I will review several semantic space designs including lexical space truncation, dimension reduction, and concept-level semantic spaces. These techniques are frequently used in NLP applications to improve computational and statistical efficiencies of distributional models, including the near-synonym lexical choice models (Chapter 3) and lexical embedding models (Chapter 4) proposed in later chapters.

## 2.1 Truncation

One simple method to alleviate sparsity is to truncate the lexical space by excluding less significant contexts with regard to co-occurrence. Given a target word $w \in \mathbb{V}$ (the lexical space), co-occurrence significance can be measured by a scoring function $s : \mathbb{V} \mapsto \mathbb{R}$ such that the scalar $s(w')$ correlates with the semantic relatedness between $w$ and its co-occurring word $w'$. After choosing a threshold $t$, words with scores lower than $t$ are simply removed from $\mathbb{V}$ to form the truncated semantic space. The scoring function takes many forms (often as preprocessing steps) in various NLP tasks. For example, frequency counts are used under the assumption that probability estimates for low-frequency words are often inaccurate and thus should not be included to affect the co-occurrence distribution. In addition, due to the Zipfian distribution of word frequencies in texts (Li, 1992), removing low-frequency words also has the advantage of significantly reducing the semantic space dimensionality and thus improves computational efficiency of the resulting model.

In the context of near-synonymy and semantic similarity, a common practice is to use statistical and information-theoretic metrics in constructing the scoring function as a measure of relatedness. Edmonds (1997), for example, used both *t-score* and *mutual information* to determine if a co-occurrence relation is to be included when constructing a lexical co-occurrence network for near-synonym lexical choice. The choice for threshold $t$, on the other hand, is largely determined by data size and the nature of the problem in question. In near-synonymy studies, since data size is usually small (on average a few thousand training instances for each near-synonym set), probability estimations are accurate only for the most frequently observed contexts and thus truncation can often be rather aggressive. For example, Inkpen (2007) limited the semantic space to the 500 most frequent co-occurring words in the development set (1989 *Wall Street Journal*). However, truncation like this introduces very strong limitations into the model since at test time, a sentence must have at least one of the 500 words in order to have a non-zero representation.

It is also possible to have dynamic thresholds conditioned on the distribution of the target or the context words. In a lexical similarity model, Islam and Inkpen (2006) used *pointwise mutual information* (*PMI*) as the scoring function. For a target word, context is defined by co-occurring words with top-$\beta$ highest PMI scores. For a word pair, the scores of words in the intersection of their contexts were summed and used as semantic similarity for the pair. The threshold $\beta$ was conditioned on the frequency of the target words as well as the size of the corpus with which the PMI scores were calculated. The authors claimed that $\beta$ values correlate with the generality of the context and have a significant impact on the evaluation performance on lexical similarity.

## 2.2 Dimension Reduction

Many challenges in NLP can be attributed to the difficulty in accurately estimating the probability of rare events. Truncation essentially avoids the estimation rather than improving it, and the semantic information conveyed by rare events are lost in the truncated semantic space. In contrast, instead of discarding rare events altogether, dimension reduction techniques use these events as “bridges” to capture higher-order co-occurrence statistics, and the goal is to reduce the lexical space dimension while maintaining important distributional properties of the lexical representations in the resulting semantic space.

### 2.2.1 Latent Semantic Analysis

One of the most popular dimension reduction techniques is *latent semantic analysis* (*LSA*, Landauer and Dumais, 1997). Latency reflects the fact that unlike the lexical space, the resulting semantic space dimensions no longer explicitly correspond to lexical dimensions. In information retrieval, LSA has been used almost exclusively to represent words and documents in low-dimensional space. Specifically, given a co-occurrence matrix $M_{n\times m}$ between $n$ word types and $m$ documents, LSA uses *singular value decomposition* (*SVD*) to factorize $M$ as follows:

$$M = U_{n\times k}\Sigma_{k\times k}V^T_{m\times k},$$

where $k$ is equal to the rank of $M$ and the subscripts indicate the dimensions of the matrices. In practice, $k$ is often further truncated to the order of $1e2$ to $1e3$, which is significantly smaller than $n$ and $m$.[1]. The matrix $\Sigma$ consists of eigenvalues of $M$ sorted in descending order along the diagonal (and zeros elsewhere). Words and documents can be represented in a $k$-dimensional space (details in Section 3.3.1), and the assumption is that local co-occurrence data encoded in matrix $M$ is transformed through SVD to implicitly induce global knowledge (Landauer and Dumais, 1997). Cohen et al. (2010) empirically demonstrated the knowledge induction by evaluating indirect inference through the similarity between a given cue word and its “nearest-indirect neighbours” (i.e., words that do not directly co-occur with the cue word). Their study showed that, compared to other dimension reduction techniques such as *random indexing* (Section 2.2.3), similarity between the LSA vectors of the indirect neighbours correlates noticeably better with human intuition.

Landauer and Dumais (1997) performed LSA on word-document co-occurrence in encyclopaedic articles. For evaluation, the resulting word vectors were used to solve TOEFL syn-

[1]The optimal value of $k$ is often task-specific and thus determined empirically. As reported by Brooke et al. (2010), for example, the order of the value of $k$ can be as low as $1e1$.

onym questions, achieving an accuracy of 64.38%. Rapp (2008) showed that LSA is more effective when performed on a word-word co-occurrence matrix, achieving an accuracy of 92.50% on the same task. Note, however, that there are compounding factors that might lead to the difference in performance, such as different co-occurrence weighting and different corpora (Rapp used the *British National Corpus* for the co-occurrence model). Nonetheless, evaluation performed on near-synonym lexical choice seems to confirm the fact that lexical semantic models can benefit more from the more fine-grained level of association on lexical co-occurrence than from word-document co-occurrence (details to follow in Chapter 3).

Computational complexity and memory footprints are the two major disadvantages of LSA. Calculating all three component matrices involves a complexity of $n^2m + m^3$ (Golub and Van Loan, 2012), and it is necessary to read the entire co-occurrence matrix and store it in memory. These are strong limitations with regard to both model efficiency and scalability, especially when the dimensionality of the lexical and the document space are often quite large in modern corpora (between $10^5$ and $10^7$).

### 2.2.2 Lexical Embeddings

Lexical embeddings are real-valued vector representations of words, which are often referred to as distributed representations in existing studies. Note that there are distinct differences between the use of the term "distributed" in the literature of distributional semantic models versus that of neural language models. In the former, the term refers to the belief that meaning is distributed by nature in that meaning is defined as the global distributional history rather than a specific set of semantic features or components (Lenci, 2008). In the latter context, distributedness specifically refers the real-valued, vector space representation of linguistic elements over a set of (usually latent) dimensions, as opposed to the local, sparse representation for indices or co-occurrence (such as the 1-hot vector or a co-occurrence matrix). The distributed representation is highly compact and thus computationally more efficient. More importantly, as a continuous function in the semantic space, distributed representation is capable of capturing the latent interrelations between linguist elements, which essentially serves as the foundation for the principle of non-local generalization (Bengio, 2009).

In practice, embedding learning is often formulated as an optimization problem, where the objective function is designed to take word embeddings as input and the amplitude of the output is in positive correlation with the semantic relatedness of the input words. As a result, after sufficient training, the distances among the resulting vectors correlate positively with the semantic distances of the corresponding words. Embedding learning can be considered as a dimension reduction technique in lexical semantic models since the dimensionality of the resulting embeddings is usually in the order of $10^2$ – $10^3$, which is much lower than that of the lexical space.

A popular implementation of embedding learning is to factorize the lexical co-occurrence matrix as the product between a target embedding matrix $M$ and a context embedding matrix $M'$. Cells in both matrices are initialized with random values. During training, given a target word and a definition of context, the model measures the similarity between the target word embedding (one row from matrix $M$) and the embeddings of the context words (multiple rows from matrix $M'$). The resulting vector similarity is correlated with the frequency of their co-occurrence observed in the training corpus, which in effect realizes the correlation between semantic similarity and vector similarity. Upon convergence, matrix $M$ is used to obtain the lexical representations while $M'$ is usually discarded.

Traditionally, lexical embeddings have been used in neural language modelling as distributed lexical representations (Bengio et al., 2003; Mnih and Hinton, 2009; Mnih and Teh, 2012). More recently, embedding models have been shown to help improve model performance in many traditional NLP tasks. Turian et al. (2010), for example, used embeddings from existing neural language models as lexical features to improve named entity recognition and chunking. Mikolov et al. (2013) developed the `word2vec` model and used negative sampling (similar to that of Collobert and Weston, 2008) and hierarchical softmax (similar to that of Mnih and Hinton, 2007) to approximate the computationally costly softmax function in the objective function, which greatly improved model efficiency and scalability.

Compared to LSA, however, embedding models with only lexical co-occurrence information exhibit the limitation that only highly local contexts are taken into account when calculating parameter updates. Several studies addressed this limitation by incorporating global co-occurrence information. For example, Huang et al. (2012) extended the local *n*-gram score proposed by Collobert and Weston (2008) by adding a global context score, which is derived from a "document vector" (centroid of the composing word vectors weighted by *idf*) that is concatenated to word vectors as model input. Pennington et al. (2014) proposed to explicitly factorize the global co-occurrence matrix between words, and the resulting log bilinear model achieved state-of-the-art performance in lexical similarity, analogy, and named entity recognition. Note that it is necessary to sample the entire co-occurrence matrix in order to derive global co-occurrence knowledge. As a result, many of these models suffer from larger-than-usual memory footprint.

As mentioned earlier, the definition of context is a crucial component in distributional semantic models. Most existing embedding models adopted the simplistic, position-based definition of context using window frames around target words. These approaches are particularly limited by their inability to capture long-distance dependencies. In addition, when a window frame spans across adjacent constituents, the model suffers from higher false-positive rates with regard to semantic relatedness. Some recent studies addressed these limitations by extending the `word2vec` model to predict words that are *syntactically* related to target words. For example, given a target word $w$ in a sentence $S$, Levy and Goldberg (2014) used words that are syntactically related to $w$ in $S$ as contexts. Bansal et al. (2014) used a manually constructed sequence of dependency relations as a template to match sequences of word tokens that follow this pattern in a training corpus. Compared to that of Levy and Goldberg (2014), this approach is more selective in the types of syntactic relations that are considered as context. Besides first-order dependency pairs, Zhao et al. (2014) is an example of exploiting higher-order syntactic relations by using ancestors, siblings, and descendants of $w$ in the parse tree of $S$ as contexts.

Semantic information has also been explored as a source of relatedness in existing studies. Bollegala et al. (2015), for example, trained lexical embeddings with three types of relational graphs (corresponding to lexical, POS, and dependency patterns) that were constructed from the sentences containing frequently co-occurring word pairs. Each relation was represented by a *pattern representation matrix*, which was combined and updated together with the *word representation matrix* in a bilinear objective function. The word representation matrix is used as the final embedding matrix. Faruqui et al. (2015) proposed to retrofit existing embeddings (i.e., published, pre-trained embeddings including those by Mikolov et al. (2013); Huang et al. (2012); Mnih and Teh (2012)) to semantic lexicons. For target word $w$, the goal is to train its embedding $\mathbf{e}$ so that it is not only similar to the pre-trained embedding vector for $w$, but also those for the ontological neighbours of $w$ specified by a semantic lexicon. The objective function interpolates two parts that clearly correspond to these two constraints. When evaluated on six tasks using six semantic lexicons (variants and combinations of WordNet (Fellbaum, 1998), *FrameNet* (Baker et al., 1998), and the *Paraphrasing Database* (Ganitkevitch et al., 2013)), the resulting model compared favourably to embeddings derived using window-based contexts.

### 2.2.3 Random Indexing

Another highly efficient embedding algorithm is *random indexing* (*RI*, Sahlgren, 2005). It is an incremental model with linear complexity $O(\max(n,m))$, where $n$ and $m$ are the dimensions of the co-occurrence matrix. Each lexical representation is initialized with a $k$-dimensional vector $v$ consisting of randomly generated values from $\{-1,1\}$. During training, the algorithm simply adds the co-occurring words' vectors to each other, resulting in lexical representations that are "effectively the sum of the words' contexts" (Sahlgren, 2005).

Theoretically, RI is supported by the *Johnson-Lindenstrauss Lemma* (Johnson and Lindenstrauss, 1984), which states that distances among the rows in a matrix can be approximately preserved during a random, linear transformation to a lower dimensionality. As a result, lexical representations are guaranteed to be near-orthogonal with the random initialization in RI, and dependency is introduced only by the vector summations during training.

Meanwhile, RI can also be interpreted as an embedding model with a simple objective function:

$$f(e(w), e(\mathbf{c})) = \sum_{c_i} e(w) \cdot e(c_i),$$

where $w$ is the target word, $\mathbf{c} = \{c_i | i = 1, \cdots, n\}$ is the collection of its contexts, and $e(*)$ is the corresponding embeddings of the words. Compared to Equation (4.1), the incremental summation of context vectors is equivalent to optimizing $f$ with stochastic gradient descent (SGD) since during each model update, the target embedding $e(w)$ is updated by adding the context vectors $\sum e(c_i)$, which happen to be the gradient $\frac{\partial f}{\partial e(w)}$ of the objective function w.r.t. the context vector. Compared to an objective function with contrastive divergence (e.g., the `word2vec` model with negative sampling), a major difference is that the optimization does not penalize non-co-occurring words by increasing the distance between their corresponding vectors and $e(w)$.

RI has been applied in several NLP tasks including text categorization (Sahlgren and Cöster, 2004), bilingual lexicon extraction (Sahlgren and Karlgren, 2005), and — in the context of synonymy and lexical similarity — solving TOEFL synonym questions (Kanerva et al., 2000; Karlgren and Sahlgren, 2001). Performance of the model is mostly on par with traditional dimension reduction techniques such as LSA, while gaining significant improvement in complexity and scalability. Notably, the largest performance improvement is achieved on the TOEFL synonym task by performing RI on a word-word co-occurrence matrix (Karlgren and Sahlgren, 2001), confirming the observations made in other studies (Rapp, 2008; Wang and Hirst, 2010).

## 2.3 Semantic Space Clustering

Another popular dimension reduction technique is to cluster lexical items that are semantically or conceptually similar. The intuition is to estimate the probability of rare events with the help of more frequent events that are conceptually similar. To cluster two words $w_1$ and $w_2$ together, the distance function used in the clustering algorithm should be chosen to correlate with lexical semantic distance, and the resulting clusters are often referred to as concepts. As a result, the sparse, lexical level co-occurrence is transformed into a denser, concept-level co-occurrence, from which more-reliable probability estimations can be derived.

Since the distance function is essentially a semantic similarity measure, it can be constructed by using either corpus statistics or hand-crafted linguistic resources such as a thesaurus or semantic taxonomy. Applying the DH on concept clusters often involves making assumptions about contextual similarity given semantic similarity among lexical items. Specifically, suppose two words $w_1$ and $w_2$ (e.g., *orange* and *apple*) are known *a priori* to be semantically similar. If a third word $w_3$ (e.g., *peeler*) is observed to frequently co-occur with $w_1$, then it is also reasonable to assume that $w_3$ is a valid context of $w_2$ — even if they have not been observed to significantly co-occur.

Examples include that of Dagan et al. (1999), where a class-based language model was formulated to calculate the conditional probability of word $w_2$ in the context of word $w_1$ as:

$$p(w_2|w_1) = \sum_{w_i \in S(w_1)} \mathrm{sim}(w_1, w_i) p(w_2|w_i).$$

Here, $S(w_1)$ is the concept cluster of $w_1$. Essentially, the model is a linear interpolation of the conditional probabilities $p[w_2|w_i \in S(w_1)]$ weighted by the similarity between $w_1$ and $w_i$ (i.e., the concept cluster of $w_2$). Several similarity metrics were experimented with as weight functions, and the proposed model resulted in notable performance improvement on a disambiguation task and modest but statistically significant improvement on a language modelling task.

Another example is that of Mohammad and Hirst (2006), where words were collated into coarse-grained categories defined in the *Macquarie Thesaurus* (Bernard, 1986). The categories are assumed to correspond to concepts or word senses. A word-category co-occurrence matrix was constructed to capture the denser, category-level co-occurrence, which outperformed lexical level co-occurrence in a lexical similarity task (Rubenstein and Goodenough, 1965) and a real-word error correction task (Hirst and Budanitsky, 2005).

# Chapter 3

# Near-Synonym Lexical Choice in Latent Semantic Space[1]

## 3.1 Introduction

*Lexical choice* is the process of selecting content words in language generation. Consciously or not, people encounter the task of lexical choice on a daily basis — when speaking, writing, and perhaps in inner monologues. Its application also extends to various domains of natural language processing including natural language generation (NLG, Inkpen and Hirst, 2006), writing assistance systems (Inkpen, 2007), and teaching and learning English as a second language (Ouyang et al., 2009).

In the context of *near-synonymy*, the process of lexical choice becomes significantly more complicated. As discussed in Chapter 1, this is partly because of the subtle nuances among near-synonyms, which can arguably differ along an infinite number of dimensions. Each dimension of variation yields differences in style, connotation, or sometimes truth conditions on an utterance being generated (Cruse, 1986), making the seemingly intuitive problem of "choosing the right word for the right context" non-trivial even for native speakers of a language.

[1]This study was first published as a conference paper in COLING 2010 (Wang and Hirst, 2010).

When the goal is to make plausible or even elegant lexical choices that best fit the *context*, the representation of context becomes a key issue. The proposed approach addresses this problem in *latent semantic space*, where transformed local co-occurrence data is capable of implicitly inducing global knowledge (Landauer and Dumais, 1997). A latent semantic space is constructed by reducing the dimensionality of the co-occurrence matrix between words and documents. This is referred to as *document level of association* (*LoA*). Document LoA can potentially benefit topical level classification tasks (e.g., as in document retrieval, Deerwester et al., 1990), but as discussed in Section 2.2, it is not necessarily suitable for lexical level tasks that require more fine-grained co-occurrence information. Empirically, significant improvement is observed from models with lexical LoA. The intuition is that lexical level co-occurrence may help recover subtle high-dimensional nuances between near-synonyms.

This claim, however, is as imprecise as it is intuitive. The notion of *subtlety* has mostly been used qualitatively in the literature to describe the level of difficulty involved in near-synonym lexical choice. Consequently, the current study also attempts to formalize and quantify the concept of subtlety based on our observations of the relationship between "subtle" concepts and their lexical co-occurrence patterns.

## 3.2 Near-synonym Lexical Choice Evaluation

In the context of near-synonymy and near-synonym lexical choice, since there does not exist much variation in the type of composition models used to represent contexts, the major distinction among the existing models is the lexical representations. Due to the symbolic nature of earlier studies, near-synonym lexical choice remained largely qualitative until the introduction of the "fill-in-the-blank" (FITB) task (Edmonds, 1997). In this task, sentences containing any member of a given set of near-synonyms were collected from the 1987 *Wall Street Journal* (WSJ). Each occurrence of any of the near-synonyms was replaced with a blank word space, and the objective was to recover the near-synonyms originally used in the sentence. Assuming the correctness of lexical choice by the original authors, the FITB test for near-synonym lexical choice provides an objective benchmark for empirical evaluation. Meanwhile, since no further annotation is required for harvesting FITB data, large volumes of gold-standard data are available at minimal cost.

Several studies adopted the FITB task for evaluating the quality of their near-synonym lexical choice models. Edmonds (1997) constructed a second-order lexical co-occurrence network on 1989 WSJ. *Word-word distance* was measured using *t-score* inversely weighted by their distance in position and the order of co-occurrence. The candidate which minimizes the sum of its distance from all words in the sentence (*word-context distance*) was considered the correct answer. The model achieved an average accuracy of 55.7% on a FITB task of seven near-synonym sets. Inkpen (2007) modelled word-word distance using *pointwise mutual information* (PMI) calculated from word counts by querying the *Waterloo Multitext System*; the PMI scores between the candidate near-synonym and context words within a window-size of $\pm 5$ were summed together as word-context distance. An unsupervised model using word-context distance achieved an average accuracy of 66.0% while a supervised method with lexical features added to the word-context distance further improved the accuracy to 69.2%. Islam and Inkpen (2010) approached the task with a 5-gram language model, in which the candidate with the highest probability given the (four-word) context was proposed as the correct answer. N-gram counts were collected from the *Google Web1T Corpus* smoothed with *missing counts*, yielding an average accuracy of 69.9%.

## 3.3 Context Representation

### Notations

Formally, the FITB task consists of the following components. Given a sentence $t$ consisting of words $\{w_1, \ldots, w_i = s_j, \ldots, w_m\}$, there is a near-synonym $s_j$ belonging to a set of synonyms $S = \{s_1, \ldots, s_n\}, 1 \leq j \leq n$. A FITB test case is created by removing $s_i$ from $t$, and the *context* (incomplete sentence) $c = t - \{s_i\}$ is presented to subjects whose task is to identify the missing word from $S$.

### 3.3.1 Latent Semantic Space Representation for Context

As we saw in Section 2.2, the first step in LSA is to build a *co-occurrence matrix* $M$ between words and documents, which is then decomposed by *singular value decomposition* (SVD) as follows:

$$M_{v\times d} = U_{v\times k}\Sigma_{k\times k}V^T_{k\times d}.$$

Here, subscripts indicate dimensions of matrices; $U$, $\Sigma$, and $V$ are the decompositions of $M$ (output of SVD), $v$ and $d$ are the vocabulary size and the number of documents, respectively, and $k$ is the dimension for the latent semantic space. A context $c$ is represented by a vector of length $v$ with ones and zeros, each marking the presence or absence of a word from the vocabulary in the context. This *lexical space* vector $c_{v\times 1}$ is then used as a *pseudo-document* and transformed into a *latent semantic space* vector by the following equation:

$$\hat{c} = \Sigma^{-1}U^T c. \tag{3.1}$$

A context is called a pseudo-document because elements in a context vector $c$ are marked in the same way a document is marked in matrix $M$, which makes $c$ essentially a new column in $M$; if we did SVD with $c$ in $M$, it would become a new column in $V$ in the latent space:

$$\langle M_{v\times d}, c_{v\times 1}\rangle = U_{v\times k}\Sigma_{k\times k}\langle V_{k\times d}, \hat{c}_{v\times 1}\rangle^T,$$

or equivalently:

$$c_{v\times 1} = U_{v\times k}\Sigma_{k\times k}\hat{c}^T_{v\times 1}, \tag{3.2}$$

where $\langle \cdot, \cdot \rangle$ stands for matrix concatenation. In our experiments, the inversion of $\Sigma$ is observed to be empirically harmful, causing about a 30% decrease in the performance of the unsupervised model. Consequently, $\Sigma$ is used instead of $\Sigma^{-1}$ in later experiments. Explanations of the difference between the two are given in Section 3.5.2.

## 3.4 Model Description

### 3.4.1 Unsupervised Model

If *cosine similarity* is used for measuring vector distance, the context representation introduced in Section 3.3.1 is equivalent to a *weighted centroid* of the context word vectors, and the lexical choice among the near-synonym candidates becomes a *k-nearest neighbour* search in the latent space (with $k = 1$ to choose the best near-synonym for the context). This is a result of the binary representation of the pseudo-document vector $c$. When transforming $c$ into the latent space representation $\hat{c}$, vector $c$ is multiplied by $\Sigma^{-1} \cdot U^T$ (Equation 3.2), and the product is essentially the sum of the rows in $U$ corresponding to the context words (and thus a centroid for the context) weighted by $\Sigma^{-1}$. This also explains the worse performance of using $\Sigma^{-1}$ instead of $\Sigma$ (Section 3.3.1), because the diagonal elements in $\Sigma$ are the singular values of the co-occurrence matrix along its diagonal, and the amplitude of a singular value (intuitively) corresponds to the significance of a dimension in the latent space. When the inverse of the matrix is used to weight the centroid, it is in effect assigning more weight to less-significantly co-occurring dimensions and thus worsens the performance.

The latent space dimensionality $k$ is an important parameter in LSA models. Due to the lack of any principled guideline for parameter selection, a grid search is conducted for the best $k$ value for each LoA according to the performance of the unsupervised model. In Figure 3.1, performance on FITB using the unsupervised model is plotted against $k$ for different LoA's. Since there are only 500 documents in the *Brown Corpus*, higher dimensional information is simply unavailable for document LoA. Lexical LoA by contrast increases quickly around $k =$

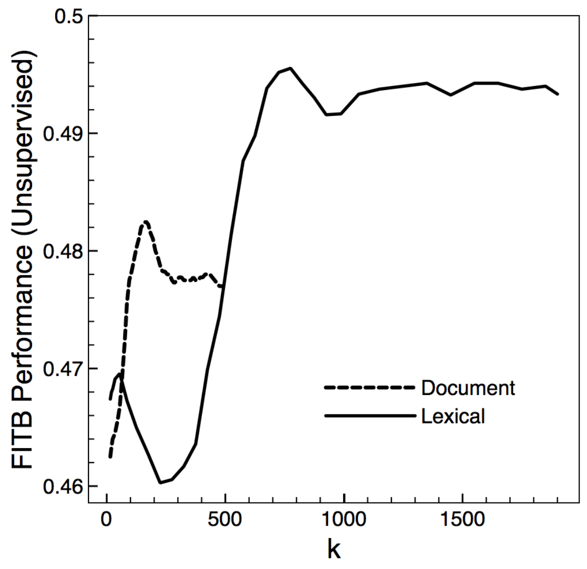


Figure 3.1: Performance of different LoA's on FITB along the latent semantic space dimensions.

550 and peaks around $k = 700$. Nonetheless, the performances differs by about one percentage point for the two LoAs under the unsupervised setting. This modest improvement suggests that, in contrast to the supervised method (Section 3.5.2), the unsupervised model is not capable of taking advantage of the high-dimensional information made available by lexical LoA.

### 3.4.2 Supervised Learning on the Latent Semantic Space Features

In traditional latent space models, the latent space vectors are almost always used in unsupervised vector space models (Landauer and Dumais, 1997). Although the number of dimensions has been reduced in the latent semantic space, the inter-relations between the high-dimension data points may still be complex and non-linear, and such problems lend themselves naturally to supervised models with non-linear mappings between the input and the output. Consequently, the near-synonym lexical choice problem is formulated as a supervised classification problem with latent semantic space features. For a test sentence in the FITB task, for example, we first get vector representations for the context in the semantic space as discussed in Section 3.3.1. The latent space representation is then paired with the correct answer (the near-synonym removed from the sentence) to form one training instance.

| Near-synonyms | Edmonds (1997) | Inkpen (2007) | Islam and Inkpen (2010) | Supervised (SVM) |
|---|---|---|---|---|
| *difficult*, *hard*, *tough* | 41.7 | 57.3 | 63.2 | 61.7 |
| *error*, *mistake*, *oversight* | 30.9 | 70.8 | 78.7 | 82.5 |
| *job*, *task*, *duty* | 70.2 | 86.7 | 78.2 | 82.4 |
| *responsibility*, *burden*, *obligation*, *commitment* | 38.0 | 66.7 | 72.2 | 63.5 |
| *material*, *stuff*, *substance* | 59.5 | 71.0 | 70.4 | 78.5 |
| *give*, *provide*, *offer* | 36.7 | 56.1 | 55.8 | 75.4 |
| *settle*, *resolve* | 37.0 | 75.8 | 70.8 | 77.9 |
| AVERAGE | 44.9 | 69.2 | 69.9 | **74.5** |

Table 3.1: LSA-SVM performance on the FITB task (% accuracy). The most-frequent baseline is 55.8%.

The *support vector machine* (SVM) is chosen as the learning algorithm for its proven advantage in classification accuracy[2] as well as its non-linear kernels capabilities. The model is trained on the 1989 WSJ and tested on 1987 WSJ to ensure maximal comparability with other results[3]. The optimal $k$ value for the latent space is 415 (determined by grid search on the training data).

Table 3.1 lists the supervised model performance on the FITB task together with results reported by previous studies. Different context window size around the gap in the FITB test sentences is also observed to affect the model performance. In addition to using the words in the original sentence, we also enlarge the context window to neighbouring sentences and shrunk to a window frame of $n$ words on each side of the gap. It is interesting to observe that, when making the lexical choice, the model tends to favour more local information — the best accuracy is achieved on a window frame of $\pm 2$. This is in accordance with observations made by previous studies (Inkpen, 2007).

[2]We experimented with several models and SVM gives noticeably better performance on the task.

[3]In Section 3.5.2 when we have to compare the supervised and unsupervised models before being able to generate the results in Table 3.1, we used *The Brown Corpus* for training and 1988 and 1989 WSJ for testing in order to keep the test data unseen.

## 3.5 Formalizing Subtlety in the Latent Space

So far in the discussion, subtlety, dimensionality, and the problem of lexical choice have been referred to in very loose terms, and the notion that near-synonym lexical choice involves differentiating subtleties remains intuitive and imprecise. These concepts and their inter-relations will be formalized in this section. Based on a unique co-occurrence property of subtlety, we empirically prove the existence of differences of subtlety in lexical choice by comparing the two learning models discussed in previous sections.

### 3.5.1 Characterizing Subtlety with *Collocating Differentiator of Subtlety*

In language generation, subtlety can be viewed as a subordinate semantic trait in a linguistic implementation of an intention. A key observation on subtlety is that it is difficult to *characterize* subtle differences between two linguistic units using their collocating linguistic units. More interestingly, the difficulty in such characterization can be approximated by the difficulty in finding a third linguistic unit satisfying both of the following constraints:

1. The unit must frequently collocate with at least one of the two linguistic units to be differentiated.

2. The unit must be *characteristic* of the major difference(s) between the pair.

The approximation is meaningful in that it transforms the abstractness of *characterization* into the concrete task of finding a third, co-occurring linguistic unit. Consider, for example the differences between the two words *pen* and *pencil*. According to the above hypothesis, the level of subtlety is in correlation with the level of difficulty in finding a third word or phrase satisfying the two constraints. If it is relatively easy to conclude that the word *sharpener* satisfies both constraints, i.e., that it frequently co-occurs with the word *pencil*, and that it is characteristic of the difference between *pen* and *pencil*, then the conclusion is the difference between the two words are not subtle. The same rule applies to word pairs such as *weather* and *climate* with *forecast*, *glass* and *mug* with *wine*, etc.

Notably, there are important differences between the collocational differentiating term (*CDT*) and the conventional sense of collocation. On the lexical level, CDTs are not only words that collocate more with one word than with the other in a pair, but also they have to be characteristic of the differences between the pair. One can easily find a collocating word exclusively for *forest* but not for *woods* — as in the example of *national forest* but **national woods*; however, unlike CDTs in the previous examples, the word *national* does not characterize any of the major differences between *forest* and *woods* in size, primitiveness, proximity to civilization, and wildness (Edmonds and Hirst, 2002) — none of these properties are necessary for something to be *national*. The word *gorilla*, on the other hand, might be more characteristic but collocating far infrequently (and thus failing the first constraint). Consequently, the lack of CDTs — or at least the relative difficulty for finding one — makes the differences between *forest* and *woods* more subtle than the previous examples. Similar observations can be made with word pairs such as *enemy–foe*, *presume–assume*, etc.

### 3.5.2 Subtlety and Latent Space Dimensionality

The concept of CDT makes it possible to relate subtlety to dimensionality in the latent semantic space. As mentioned in Section 3.4.1, dimensions in the latent semantic space are arranged in descending order with regard to co-occurrence significance since, by the nature of SVD, the information within the first dimensions in the latent space are formulated from more significantly collocating linguistic units. Relating this to the discussion above, it follows that it should be relatively easy to find CDTs for target words that can be well distinguished in a lower-dimensional sub-space of the latent semantic space. Under the hypothesis regarding CDT in the previous section, the differences should therefore be less subtle for these target words. Consequently, co-occurrence-based information capable of characterizing more subtle differences must then reside in higher dimensions in the latent space vectors.

Empirically, the supervised FITB task performance is observed to correlate positively with the number of dimensions in the latent semantic space (Figure 3.2). This suggests that the

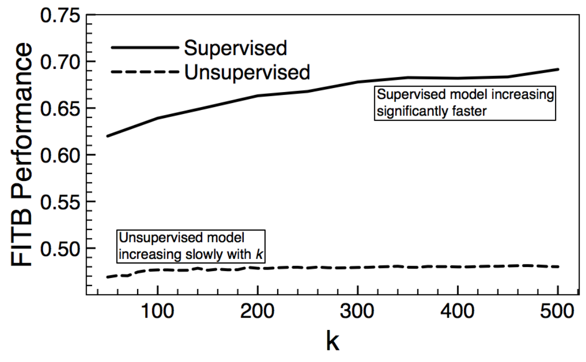


Figure 3.2: Supervised method gaining further improvement over unsupervised method in higher dimensions.

lexical choice process is indeed capturing the implicit information on subtlety in the higher dimensions of the latent vectors. A comparison between the supervised and unsupervised methods further confirmed our intuition about the complexity of subtlety. As shown in Figure 3.2, performance of the supervised method keeps increasing towards the higher end of dimensionality, while that of the unsupervised mostly remains flat. Assuming one of the differences between the supervised and unsupervised methods used here is that the former is better at unravelling the non-linear relations in training data (with the non-linear kernel), this comparison then suggests that the subtlety-related information in the higher dimensions exhibits complex mapping functions between the input (contexts) and output (near-synonyms) that are beyond the capability of the *k*-nearest-neighbour model.

The fact that the unsupervised performance remains flat is not surprising considering, again, the construction of the latent space representation of the near-synonyms and context (Section 3.3.1). Elements in higher dimensions of these vectors are significantly *weighted down* by $\Sigma$, whose diagonal elements are in descending order in value. Consequently, higher dimensions in the representation would have very little influence over the cosine distance function in the unsupervised model. In other words, the information about subtlety in higher dimensions does not have enough influence over whatever decision is made in the lower dimensions based on explicit co-occurrence data, and performance is almost invariant with respect to latent space dimensionality. It will eventually decrease significantly when sparsity becomes an issue.

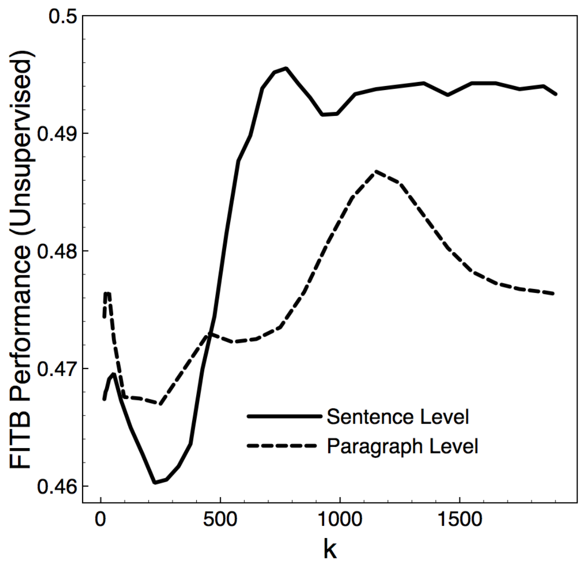


Figure 3.3: LoC (level of context) in correlation with latent space dimensionality for optimal model performance.

### 3.5.3 Subtletly and the Level of Context

The lexical LoA associates words within the same sentence. Lexical LoA also allows us to relate words that co-occurs in different *levels of context* (LoC) such as paragraph or even the entire document. This gives us an approximate measurement on how much context a lexical LoA model uses for word-word co-occurrence. Intuitively, if two words are distinguishable only through larger LoC, their differences also tend to be more subtle.

To test this hypothesis, we compared the performance of models built on different LoC's. Figure 3.3 plots the unsupervised performance for models with sentence and paragraph LoC. There is a significant difference in the optimal $k$ value for the performance of the models. Sentence LoC performance peaks around $k = 700$ — much lower than that of paragraph LoC which is around $k = 1,100$. Under the previous hypothesis between latent space dimensionality and the level of subtlety, this difference in $k$ value suggests that each model may have its own "sweet spot" for the degree of subtlety it aims at distinguishing; models on larger LoC are better at differentiating between more subtle nuances, which is in accordance with intuition.

# Chapter 4

# Dependency-based Embedding Models and its Application in Lexical Similarity[1]

## 4.1 Using Syntactic and Semantic Knowledge in Embedding Learning

The discussions in Section 2.2.2 suggests that compared to LSA, distributed representations not only better capture semantic relations among words but also are computationally less expensive to obtain. Several existing embedding models attempted to address the position-based definition of context using window frames. However, all of these models have their respective limitations in context representation. Syntactically, dependency information is often used in an overly simplistic manner, in which, for instance, many different types of syntactic relations of heterogeneous natures are conflated in the same contextual space without distinction. On the other hand, semantic knowledge used in existing models either is derived from automatic but complex procedures that are difficult to reproduce, or relies heavily on expensive linguistic resources (such as semantic ontologies or manually annotated corpora), while ignoring simple, effective, and widely available knowledge sources. Consequently, one of the major objectives

[1]This study was first published as a conference paper in ACL 2015 (Wang et al., 2015).

of this study is to address these limitations in existing embedding models.

### 4.1.1 Relation-dependent Models

As shown in Section 1.1, the definition of context is an integral part in the application of the distributional hypothesis (DH). Consequently, we first formulate definition of lexical association (between a target and its context) as a unified framework to incorporate different types of linguistic environments into embedding learning. Formally, a certain definition of association (denoted $\tilde{R}_k$) defines not only a particular pair of input/output vocabulary space ($V_I^{(k)}$ and $V_O^{(k)}$), but also a probability distribution of all possible combinations of input-output pairs being associated under $\tilde{R}_k$. Position-based co-occurrence, for example, can be regarded as an instance of association, and the probability of words being associated under this relation (e.g., co-occurring within a window frame) defines either a bigram language model or a bag-of-words model depending on whether the input-output lexical spaces are ordered.

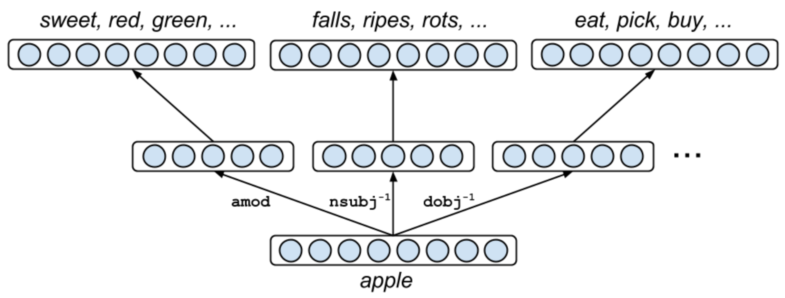
The abstraction is important in that it facilitates the incorporation of syntactic and semantic information into language modelling — and by extension, into embedding learning. For example, association can be defined as the dependency relation "adjective modifier". Given a target word *apple*, the resulting model is able to assign higher probability to *red* or *green* than to *black* or *blue*. Similarly, other aspects of the meaning of an input word can be learned by having different definitions of association using other syntactic relations. From the perspective of the DH, the knowledge about the target word's distribution in its linguistic environment is at least in correlation with its meaning. Consequently, although far from truly understanding the global meaning of a target, a model with this type of distribution knowledge can still be considered semantically informed about certain attributes of the target (e.g., more probable taste or colours of the target), which can be used to improve the performance of many downstream NLP applications. Compared to existing embedding models, information thus obtained about a target word's meaning is more accurate and more fine-grained than both the window-based co-occurrence and the non-discriminative definition of syntactic relatedness.

Note that the definition of association is not limited to syntactic dependencies. For example, the defining relation from a monolingual dictionary has been demonstrated to be highly informative of semantic associations (Alshawi, 1987; Wang and Hirst, 2012). Specifically, definition texts can be used to establish association between *apple* and the words it defines such as *pippin* and *cider*, etc., or its defining words such as *fruit* and *rosaceous*, etc. The advantage of using these lexicographic relations is two-fold. Firstly, although the defining terms are highly related in meaning to the target words, the degree of relatedness can be severely undermined in a distributional model since the co-occurrence statistics among these words might be insignificant, especially given the low frequency of either the target or the defining words (e.g., *pippin*). Secondly — and perhaps less conspicuously, the most fundamental or representative properties of an entity are quite possibly missing from its distributional contexts. For example, we might not expect the phrase *rosaceous apple* to occur with significant frequency — if at all. Consequently, the lexicographic relation can effectively supplement the distributional contexts by including these important descriptive terms.

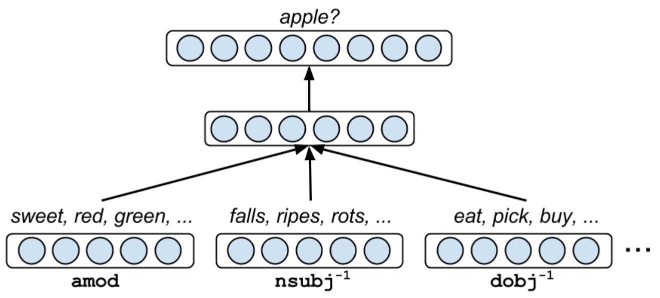
### 4.1.2 Relation-independent Models

Each of the resulting models makes use of only one of the syntactic or semantic relations between words and thus, they are named *relation-dependent models*. However, the majority of NLP tasks take words as inputs in the form of string tokens without syntactic or semantic context. As a result, an embedding model should ideally also be able to produce relation-independent embeddings for words. Consequently, I also propose a post-hoc, *relation-independent model* to achieve this goal by combining the relation-dependent embeddings.

The basic idea for the relation-independent model is to add an extra layer to the network to reconstruct the *original* input word given the relation-dependent representations. The intuition is that, the relation-dependent embedding models learn the meaning of a word by predicting its syntactically- or semantically-related words, while the relation-independent model essentially performs a post-hoc, “riddle-solving” procedure that predicts the original word given its related



(a)



(b)

Figure 4.1: (a) Training relation-dependent models with dependency-based contexts. (b) Training a relation-independent model by predicting the target word given its relation-dependent embeddings.

words. For example, as shown in Figure 4.1-(a), for an input noun *apple*, the relation-dependent model is trained to answer questions such as what can it do (e.g., *grows*, *rots*, *falls*, or its other noun-subject governors), what can be done to it (e.g., *buy*, *eat*, *pick*, or its other direct-object governors), what attributes it possesses (e.g., *sweet*, *big*, *red*, or its other adjective-modifier dependents), etc. Conversely, if the algorithm knows that something can be sweet, big, or red, one can pick it, buy it, eat it, etc., then the post-hoc, relation-independent phase is to train a model that "solves the riddle" and predicts that this "something" is an apple (Figure 4.1-(b)). The two-tier structure in the proposed model essentially makes use of the individual syntactic and semantic relations to guide the intermediate lexical representation learning in the relation-dependent phase, which is superior than existing models that directly learn relation-independent representations.

## 4.2 Optimization

The relation-dependent model is trained with an objective function similar to that of Mikolov et al. (2013):

$$\log \sigma(\mathbf{e}_g^T \mathbf{e}'_d) + \sum_{i=1}^{k} \mathbf{E}_{\hat{d}_i}[\log \sigma(-\mathbf{e}_g^T \mathbf{e}'_{\hat{d}_i})], \tag{4.1}$$

where $\mathbf{e}_*$ and $\mathbf{e}'_*$ are the target and the output (context) embeddings for the corresponding words, respectively, and $\sigma$ is the sigmoid function. The subscripts $g$ and $d$ indicate whether an embedding corresponds to the governor or the dependent of a dependency pair. When deriving relation-independent embeddings, target words are always assumed to be governors in relevant relations (and contexts to be dependents). Inversion is performed whenever necessary in order to satisfy this assumption. Finally, $\hat{d}_*$ correspond to random samples of dependents drawn with unigram frequency from the dependent vocabulary, and $\mathbf{E}_{\hat{d}_i}$ stands for expectation over these samples. These words are assumed to be semantically unrelated to the target word. Lexicographic knowledge is represented by adopting the same terminology used in syntactic dependencies: definienda as governors and definientia as dependents. For example, *apple* is related to *fruit* and *rosaceous* as a governor under `def`, or to *cider* and *pippin* as a dependent under `def`$^{-1}$.

In contrast, relation-independent embeddings are derived by applying a linear transformation $M$ on the concatenation of the relation-dependent embeddings $\tilde{\mathbf{e}}_g$ of target words, and the objective function is defined as:

$$\log \sigma(\mathbf{e}_g'^T M \tilde{\mathbf{e}}_g) + \sum_{i=1}^{k} \mathbf{E}_{\hat{d}_i}[\log \sigma(-\mathbf{e}_{\hat{d}_i}'^T M \tilde{\mathbf{e}}_g)].$$

Notations follow those used in the relation-dependent case, (e.g., $\mathbf{e}_*$ and $\mathbf{e}'_*$ are the target and the context embeddings for the corresponding words, respectively). There are two notable differences from the relation-dependent model. Firstly, $\hat{d}_i$'s are negative samples generated according to the unigram frequency of the entire training corpus (instead of only the dependent vocabulary of a certain dependent relation), and they are used to generate contrast against the target words. Secondly, parameter updates for $M$ are defined as:

$$\frac{\partial}{\partial M} = [1 - \sigma(\mathbf{e}_g'^T M \tilde{\mathbf{e}}_g)]\mathbf{e}_g' \tilde{\mathbf{e}}_g^T - \sum_{i=1}^{k} [\sigma(-\mathbf{e}_{\hat{d}_i}'^T M \tilde{\mathbf{e}}_g)]\mathbf{e}_{\hat{d}_i}' \tilde{\mathbf{e}}_g^T.$$

Since $\tilde{\mathbf{e}}_g$ is a real-valued vector (instead of a 1-hot vector as in the relation-dependent models), $M$ can not be updated one column at a time (as with the target and the output embedding matrices in the relation-dependent case). Instead, as indicated by the outer product between $\mathbf{e}_g'$ and $\tilde{\mathbf{e}}_g^T$ (and between $\mathbf{e}_{\hat{d}_i}'$ and $\tilde{\mathbf{e}}_g^T$), each update is a dense matrix of the same size as $M$ ($d \times n \cdot d$, with $d$ being the dimensionality of the relation-dependent embeddings and $n$ being the number of relations to be composed by the relation-independent model). In practice, however, training is nonetheless quite efficient, since the dimensionality of $M$ is considerably lower than that of the relation-independent embedding matrices, and convergence is achieved after very few epochs.[2]

## 4.3 Evaluation

### 4.3.1 Data Preparation

The relation-dependent models are trained on the 4-billion-word *Annotated English Gigaword Corpus* (Napoles et al., 2012, dubbed *4B*). A 17-million-word subset of the corpus (dubbed *17M*) is also used to study the effect of data size. Dependency information is obtained from the dependency parses that comes with the distribution of the corpus. All dependency relations are used except the following four types: `det` (determiner, as in *man–det–the*), `expl` (existential there, as in *is–expl–there*), `dep` (indeterminable relations or parser error), and `root` (the head verb of a sentence). The first two are excluded due to the low degree of semantic association

[2]We also experimented with updating the relation-dependent embeddings together with $M$, but this worsened evaluation performance.

between governors and dependents, and the last one due to the fact that `root` is a unary relation.

On the lexical level, words with either *SYM* (symbols), *LS* (list item marker), or non-alphabetical part-of-speech (usually punctuations) are excluded. Numbers and proper nouns (identified by their POS tags) are conflated into special tokens (`_CD_` and `_NP_`, respectively). This pre-processing step implicitly excludes many dependency relations such as `num` (numeric modifier), `punct` (punctuation), etc., which are clearly insignificant as far as relatedness between the governors and the dependents is concerned. A frequency threshold of 30 is applied on the lexical level, and dependency pairs containing infrequent words (either as the governor or the dependent) are excluded from the training data. This significantly reduces the vocabulary size and improves efficiency with regard to both runtime and storage space. However, no frequency threshold is applied for the *17M* corpus.

Semantic relations are derived from the *Online Plain Text English Dictionary*[3] (version 0.0.3, dubbed *OPTED*). There are approximately 806,000 definition pairs in total, with 33,000 distinct definienda and 24,000 distinct defining words. The entire corpus has 1.25 million words contained in a 7.1MB compressed file. No additional processing is performed on this data.

Both the syntactic and semantic relations are represented as matrices with governors as rows, dependents as columns, and their co-occurrence counts as cell values. Data points are presented to the models in the form of dependency pairs by random sampling. Before sampling, however, each pair is presented once to the models to ensure that the models are exposed to all pairs including the rare ones. Subsequent data points (governor-dependent pairs) are then sampled according to their frequency. Since the vocabulary size is usually much smaller (approximately 5%) compared to the number of data points, the first pass through the entire vocabulary does not significantly affect the sample frequencies.

Hyperparameters used to obtain the reported results are listed in Table 4.1.

[3] `http://www.mso.anu.edu.au/~ralph/OPTED/`

| Corpus | Dimension | Learning Rate | Neg. Sample Size | Epoch |
|---|---|---|---|---|
| Relation-Dependent Models | | | | |
| *17M* | 50 | .025 | 15 | 30 |
| *4B* | 50 | .025 | 5 | 1 |
| *OPTED* | 50 | .025 | 15 | 30 |
| *OPTED*$^{-1}$ | 50 | .025 | 15 | 60 |
| Relation-Independent Models | | | | |
| – | 50 | .025 | 15 | 60 |

Table 4.1: Hyperparameters for the proposed embedding models.

### 4.3.2 Lexical Semantic Similarity

The resulting word embeddings are evaluated on the lexical semantic similarity task, where datasets consist of a number of word pairs, each rated by human judges with regard to their semantic similarity. A vector space model (such as an embedding model) can then be applied to produce a vector for each of the words, and the cosine similarity between the vectors for a pair is used as the system score for the pair in question. *Spearman's* $\rho$ is used to measure the correlation between the system scores and the human ratings, and the degree of correlation indicates how well the model under evaluation captures the semantics of the target words.

Six datasets are used in the experiments. *RG* (Rubenstein and Goodenough, 1965) is the earliest of the sets and consists of 65 noun pairs. *MC* (Miller and Charles, 1991) is a 30-pair subset of the *RG* dataset. *FG* (Finkelstein et al., 2001) consists of 353 noun pairs, which is superset of the *MC* dataset. *SimLex* (Hill et al., 2015b) contains 999 pairs of mixed POSs. In our experiments, this dataset is split by POS into $SL_n$ of 666 noun pairs, $SL_v$ of 222 verb pairs, and $SL_a$ of 111 adjective pairs. Semantically, *FG* does not make explicit distinctions between similarity and relatedness, whereas the other three datasets are more similarity oriented, particularly the *SimLex* data. For example, the pair *clothes–closet* is rated 8.00/10 in *FG* but 1.29/10 in $SL_n$). As shown in later sections, the definition of contexts may determine whether a distribution model is able to distinguish true similarity from relatedness. As a result, this difference among the datasets can serve as a suitable benchmark for evaluating this particular aspect of

| Dataset | *MC* | *RG* | *FG* | $SL_n$ | $SL_v$ | $SL_a$ |
|---:|---|---|---|---|---|---|
| d.f. | 30 | 65 | 353 | 666 | 222 | 111 |
| *t*-value | 1.70 | 1.67 | 1.65 | 1.65 | 1.65 | 1.66 |
| $\rho$ | 0.31 | 0.21 | 0.09 | 0.06 | 0.11 | 0.16 |

Table 4.2: Degree of freedom, critical *t*-values (after rank permutation), and critical values of Spearman's $\rho$ (i.e., the minimal values of $\rho$ for the results to be statistically significant in each dataset).

an embedding model. More detailed discussions on the distinction between relatedness and similarity can be found in Section 1.2.2.

Statistical significance of the correlation is tested by performing a permutation test on the Spearman's $\rho$, the result of which conforms to Student's *t* distribution Rosenhead (1963). For a confidence interval of 0.05, the critical values for the datasets are shown in Table 4.2, which suggest that the all results reported below exhibit statistical significance in correlation with the human judgements of similarity.

However, there exists much controversy in parametric significance tests on the *difference* between multiple Spearman's $\rho$'s. For example, some existing studies treated the Spearman's $\rho$ as the Pearson correlation significance test (Gabrilovich and Markovitch, 2009), since the latter has a more established parametric method for significance testing. However, due to the difference in the nature of the two correlation metrics, the practice actually received much criticism for the conversion. This is because a lack of significant difference in two similarity metrics measuring the *linear* correlation (i.e., Pearson correlation) does not necessarily imply the lack of significant difference in the rank-based correlation (i.e., Spearman's $\rho$) on the same observations. Without a more scientific way of comparing the correlations, we opt for comparing the correlations without significance testing — which is the de facto practice in exiting studies involving lexical similarity tasks.

| Relation | Description | Example | Governor | Dependent |
|---|---|---|---|---|
| `amod` | Adjective modifier | *a red apple* | *apple* | *red* |
| `nn` | Noun modifier | *apple farm* | *farm* | *apple* |
| `nsubj` | Noun subject | *the apple falls* | *falls* | *apple* |
| `dobj` | Direct object | *eat an apple* | *eat* | *apple* |

Table 4.3: The relations used for relation-dependent models in the lexical similarity task.

| Model | *MC* | *RG* | *FG* | $SL_n$ | $SL_v$ | $SL_a$ |
|---|---|---|---|---|---|---|
| `amod` | **.766** | **.798** | .572 | **.566** | .154 | .466 |
| `amod`$^{-1}$ | .272 | .296 | .220 | .218 | .248 | **.602** |
| `nsubj` | .442 | .350 | .376 | .388 | **.392** | .464 |
| `nn` | .596 | .620 | .514 | .486 | .130 | .068 |
| Baselines | | | | | | |
| `DEP` | .640 | .670 | .510 | .400 | .240 | .350 |
| `w2v` | .656 | .618 | **.600** | .382 | .237 | .560 |
| `GloVe` | .609 | .629 | .546 | .346 | .142 | .517 |

Table 4.4: Correlation between human judgement and *cosine* similarity of embeddings (trained on the Gigaword corpus) on six similarity datasets.

### 4.3.3 Baseline Models

Three existing embedding models are used for comparison with the proposed models. To demonstrate the advantage of using syntactic and lexicographic knowledge in embedding learning, we choose `w2v` (Mikolov et al., 2013) and `GloVe` (Pennington et al., 2014), both of which use window-based co-occurrence to define lexical association. Both models are widely adopted for their efficiency and scalability. However both require very large amounts of training data to achieve optimal results, which is also an important aspect for comparison particularly with our proposed dictionary-based model. Another baseline model is a dependency-based model (Levy and Goldberg, 2014) that does not distinguish among different types of dependency relations. The model is included in order to show the effect of making such distinctions particularly in the aspect of forcing the model input to be syntactically and semantically more focused.

### 4.3.4 Relation-dependent Models

Considering the fact that the majority of the target words in the similarity datasets are nouns, four noun-related syntactic relations are chosen for the relation-dependent model experiments including `amod`, `nn`, `nsubj`, and `dobj`[4]. Details of these relations are given in Table 4.3. Evaluation performance of the relation-dependent models are listed in Table 4.4.

#### Parser Errors and Exceptions

Given a dataset, relations with better performance should be the ones whose governors are of the dominant POS of that dataset (e.g., noun for *MC*, verb for $SL_v$, etc.). Empirically, `amod`, and $\texttt{dobj}^{-1}$ indeed perform significantly better than their inverse relations on the noun-dominant datasets, whereas `nn` and $\texttt{nn}^{-1}$ are approximately on the same level. However, the assumption fails on the subject-verb relation on the noun-dominant dataset since performance of `nsubj` is actually higher than $\texttt{nsubj}^{-1}$.[5] Error analysis reveals that a large number of the target words have appeared as governors in `nsubj` — which are expected to be verbs. For the *MC* dataset, for example, after excluding the nouns that are frequently used as verbs (*implement*, *shore*, *smile*, and *string*), there are 8,043 occurrences of the remaining nouns as governors in `dobj`, while the count is 170,921 for `nsubj`. In other words, when normalized by the respective data size for the two relations, the proportion of noun governors in the `nsubj` is about 12.4 times the proportion in `dobj`. Further investigation reveals that this abnormally large discrepancy is caused by an intended feature in the Stanford parser. When the head verb of a sentence is copula or periphrastic, the corresponding nominal or and adjectival predications are used as governors of `nsubj` instead of the verb itself. In future work, distinctions need to be made for these scenarios to model the noun-verb relation more accurately.

Parser error is another source of inaccuracy in the data for the relation-dependent model. For four example target nouns from *MC*, Table 4.5 lists some randomly selected examples

[4]Indirect object is not included due to its overly small data size (less than 5% of that of `dobj`).

[5]Correlations for the $\texttt{nsubj}^{-1}$ model are not statistically significant when the model is trained on the *17M* corpus and hence there results are omitted in Table 4.4.

| Governors | dependents | | | | |
|---|---|---|---|---|---|
| car | explosion | runner-up | color | coupe | industry |
| automobile | cause | interests | hybrid | mainland | system |
| boy | underling | company | problem | fighter | prodigy |
| lad | boy | dad | kid | man | son |

Table 4.5: Examples of noun target words from the *MC* dataset appearing as verb governors in the `nsubj` relation and their corresponding noun dependents.

of their occurrences as verb governors of the `nsubj` relations caused by parser error. Manual inspection suggests that this type of errors will not significantly affect model performance since most of these examples nonetheless exhibit semantic relatedness to the target words.

### Comparing Adjectival and Nominal Modifiers

For target words of the same POS, there exists a large performance gap among different relations. For example, the relation `amod` is noticeably more effective for nouns than other relations. Curiously, although both are noun-modifying relations, `nn` (or its inverse) does not perform nearly as well as `amod`. Data inspection reveals that compared to `amod`, the `nn` relation has a much higher proportion of proper nouns as modifiers. An interesting difference between the two relations can be inferred from this observation: for a nominal concept, `amod` often describes its *attributes* (e.g., *uncharted shore*, *scenic coast*), whereas `nn` phrases are often instantiations (e.g., *Mediterranean shore*, *California coast*). Semantically, the former is apparently more informative about the nominal governors than the latter, which, empirically, is in accordance with the better performance in the similarity tasks. We experimented further by training the same models after excluding pairs with proper noun modifiers from the training data. Performance of $\texttt{nn}^{-1}$ is improved over all noun-dominant datasets, while the same procedure does not have noticeable effect on the performance of `amod`.

### The Effect of Training Data Size

Training data size appears to play a more complex role than simply “the more the better”. Among the three relations `amod`, `nn`, and `nsubj`, for example, performance is in fact negatively correlated with the amount of data available for each relation. As is seen from the previous comparisons, the quality of the data — and most importantly, the nature of the syntactic relation in relation to the meanings of the target words — plays a more important role than data size. Furthermore, for *MC* and *RG*, the best-performing relation-dependent model (`amod`) is on par with the baseline models when trained on the smaller *17M* corpus. With the full Gigaword corpus, the model outperforms both `word2vec` and `GloVe` by large margins. We offer two possible explanations for the data-size-induced difference in performance. Firstly, co-occurrence data become sparser when factorized into multiple syntactic relations, and thus each individual relation requires more data to achieve sufficient co-occurrence statistics for the resulting embeddings to capture lexical semantic relatedness. Secondly, with sufficient data, individual syntactic dependencies are able to provide more “focused” (Levy and Goldberg, 2014) co-occurrence statistics and thus achieve better performance than distributional contexts. This result empirically confirms the fundamental hypothesis of this study.

### Comparison Between Similarity and Relatedness

On the *FG* dataset, the best-performing relation-dependent model (`amod`) performs better than `GloVe` but not as well as `word2vec` (which also outperforms the non-factorized dependency-based embedding model `DEP`). A possible explanation is that *MC* and *RG* are more oriented towards semantic relatedness, while words in *FG* are often related through distributional similarity (e.g., *bread–butter*, *movie–popcorn*, etc.). The difference in performance might suggest that the proposed syntax-based models are better at capturing semantic similarity rather than distributional relatedness. Remember that Hill et al. (2015b) made clear distinctions between similarity and relatedness: related words with high (relatedness) scores in *FG* are rated much lower in the *SL* datasets (Section 4.3.2). Not coincidentally, the relation-dependent models

| Model | *MC* | *RG* | *FG* | $SL_n$ | $SL_v$ | $SL_a$ |
|---|---|---|---|---|---|---|
| `amod` | .512 | .486 | .380 | .354 | | |
| $\texttt{dobj}^{-1}$ | .390 | .380 | .360 | .304 | | |
| Combined | **.570** | **.550** | .392 | **.360** | | |
| `nsubj` | | | | | .222 | |
| `dobj` | | | | | .206 | |
| Combined | | | | | **.238** | |
| $\texttt{amod}^{-1}$ | | | | | | .394 |
| $\texttt{dobj}^{-1}$ | | | | | | *.236* |
| Combined | | | | | | .338 |
| Baselines | | | | | | |
| `DEP` | .568 | .548 | .498 | .352 | .108 | **.436** |
| `w2v` | .563 | .491 | **.562** | .287 | .065 | .379 |
| `GloVe` | .306 | .368 | .308 | .132 | −.007 | .254 |

Table 4.6: Lexical similarity performance of relation-independent models (trained on the *17M* corpus) combining top two best-performing relations for each POS.

outperform the baselines with much larger margins on these datasets, further confirming the hypothesis that the proposed models are better at capturing true similarity than relatedness.

### 4.3.5 Relation-independent Models

As mentioned in Section 2.2.2, one of the advantages of the proposed models over existing dependency-based embedding models is that the interaction between the individual relations can be investigated by factorizing different syntactic and semantic relations. Using the relation-independent model proposed in Section 4.2, different combinations of relation-dependent models are implemented and evaluated in this section.

Performance of relation-independent models is summarized in Table 4.6. Since the major motivation for the relation-independent model is to alleviate sparsity caused by factorizing co-occurrence information into individual dependency relations, we only report the results obtained on the smaller *17M* corpus. Given the large number of relations available, it is impractical to experiment with all possible combinations in an exhaustive manner. Instead, for each POS, only two relation-dependent models with the best performance are chosen according to the experiments in Section 4.3.4. Improvements are observed consistently for all datasets except for $SL_a$. The reason for the lower performance on this dataset is that the second-best performing relation for adjectives (`dobj`$^{-1}$) does not yield statistically significant correlation with human judgements. In other words, the relation-independent model is essentially adding noise to the best performing relation for adjectives and hence the decrease in performance.

In contrast, combining syntactic relations does more harm than help on the larger, *4B* corpus (not listed in Table 4.6). Since the relation-dependent embeddings are of the same dimension on the *17M* and *4B* corpora (both 50), the only difference with regard to model training is the vocabulary size. Note that the input vocabulary of a relation-independent model is the intersection of the vocabularies of its upstream, relation-dependent models. The intersection does not significant change the input vocabulary for the *4B* corpus due to the large vocabulary size for the individual relations. On *17M*, however, it is observed that combining multiple relations sometimes increases the size of the out-of-vocabulary (OOV) set. Since the OOV words are always infrequent words, which are usually more difficult to model due to their inaccurate co-occurrence profiles, the shrinkage during vocabulary intersection might have given the *17M* model an implicit advantage over the *4B* model, and hence performance improvement is witnessed on the former but not the latter.[6] It is also possible that the relation-independent model requires different hyper-parameters such as negative sample size or number of epochs to offset the difference in vocabulary size. In either case, further investigations are needed to offer more insightful explanations for the performance discrepancy on the two corpora.

### 4.3.6 Using Dictionary Definitions

Both relation-dependent and relation-independent models are evaluated with dictionary definition texts as training data, and results are listed in Table 4.7. Both the defining relation (`def`) and its inverse (`def`$^{-1}$) perform surprisingly well with the relation-dependent model, especially considering the highly compact size of the training data. The proposed models are trained on 1.2 million tokens of hand-crafted data. When compared to three existing models

[6]It is possible to mimic the advantage on the *4B* data by proper choices of frequency thresholds for the relation-dependent models to be combined, which is beyond the scope of discussions in this study.

| Model | *MC* | *RG* | *FG* | $SL_n$ | $SL_v$ | $SL_a$ |
|---|---|---|---|---|---|---|
| `def` | .640 | .626 | .378 | .332 | .320 | .306 |
| `def`$^{-1}$ | .740 | .626 | .436 | .366 | .332 | .376 |
| Combined | **.754** | **.722** | .530 | **.410** | **.356** | **.412** |
| `DEP` | .530 | .558 | .506 | .346 | .138 | **.412** |
| `w2v` | .563 | .491 | **.562** | .287 | .065 | .379 |
| `GloVe` | .306 | .368 | .308 | .132 | −.007 | .254 |

Table 4.7: Lexical similarity performance of models using dictionary definitions compared to `word2vec` trained on the Gigaword corpus.

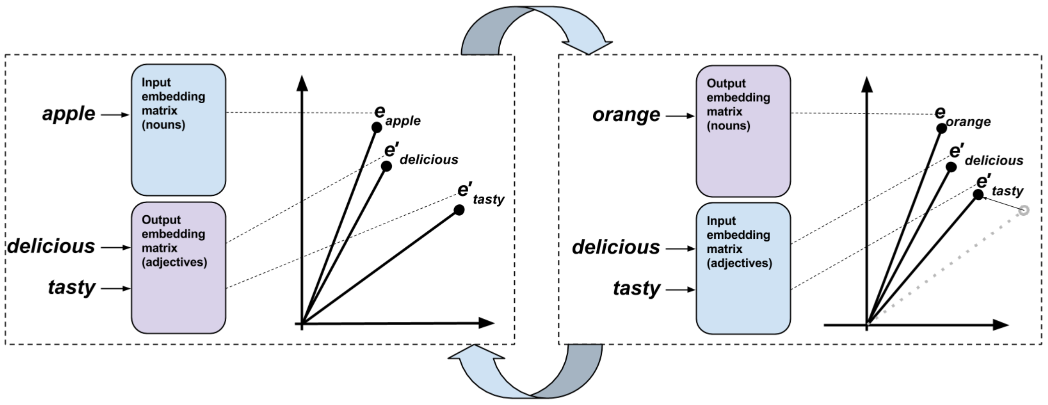

apple
Input embedding matrix (nouns)
$e_{apple}$
$e'_{delicious}$
$e'_{tasty}$
delicious
tasty
Output embedding matrix (adjectives)
orange
Output embedding matrix (nouns)
$e_{orange}$
$e'_{delicious}$
$e'_{tasty}$
delicious
tasty
Input embedding matrix (adjectives)

Figure 4.2: Co-training embeddings between the input and the output space.

trained on the 4-billion-word *Gigaword Corpus*, the relation-dependent models (the first two rows in Table 4.7) perform better on almost all of the semantically oriented similarity tasks. The relation-independent model brings consistent improvement by combining the two individual relations, and the results compare favourably to other models on all five semantically oriented dataset. The results also suggest that, similar to dependency relations, lexicographic relations are a reliable knowledge source for capturing similarity than relatedness.

## 4.4 Input-output Embedding Co-training

As mentioned earlier in Section 4.2, the embedding space of target words $\{\mathbf{e}\}$ is independent from that of the contexts $\{\mathbf{e}'\}$. Algebraically, the independence is a result of formulating the embedding learning problem as a matrix factorization problem, where some form of the lexical co-occurrence matrix is factorized into two matrices corresponding to the two embedding

| Model | *MC* | *RG* | *FG* | $SL_n$ | $SL_v$ | $SL_a$ |
|---|---|---|---|---|---|---|
| amod | .512 | .486 | .380 | .354 | | |
| $\texttt{amod}_{ct}$ | .578 | .544 | .432 | .384 | | |
| nsubj | | | | | .222 | |
| $\texttt{nsubj}_{ct}$ | | | | | .250 | |
| $\texttt{amod}^{-1}$ | | | | | | .394 |
| $\texttt{amod}_{ct}$ | | | | | | .428 |

Table 4.8: Comparison of lexical similarity performance before and after embedding co-training by inverting dependency relations and input-output embedding matrices.

spaces. Practically, the difference entails that the output embedding matrix only plays an auxiliary role to support better embedding learning in the input matrix. For example, in negative sampling, the contrastive learning signal is generated by the difference between the target word and the negative samples. According to Equation (4.1), these signals only go to the input embedding matrix. In other words, the principle of contrastive divergence is applicable to the input embedding matrix only, and hence the input embeddings are updated with more information than the output embeddings. This difference is consistent with the empirical observations that the input embeddings result in better evaluation performance than the output embeddings.

However, we hypothesize that given the directed nature of the relation-dependent models (i.e., with the input and the output corresponding to the governors and dependents of a dependency relation, respectively), the output embeddings must encode semantic information about the dependents. By modifying the dependency-based embedding models (Section 4.1), we propose a co-training mechanism that can make use of the output embeddings to improve the quality of the input embeddings. For example, suppose the goal is to train an `amod` model with noun governors as inputs and adjective dependents as outputs (the left diagram in Figure 4.2). According to the previous discussions, the adjective (output) embeddings might not be optimized to capture the meaning of adjectives due to the lack of the contrastive learning signals. The basic idea of co-training is to switch the input and the output embedding matrices and inverse the governor-dependent relation in the original training data. Specifically, we use the current cell values of the output (adjectival) matrix as an initial state to train an $\texttt{amod}^{-1}$ model, and use a modified version of the `amod` data by swapping the governors and dependents. During training, the adjectival embeddings should benefit from from the contrastive signals from the nominal negative samples (right diagram in Figure 4.2). The resulting adjectival embeddings are expected to be better at capturing the meanings of the adjectives and in turn serve as a better initial state for the next phase of training an `amod` model when the switching procedure is repeated. Essentially, the idea is to recycle the states of both input and output embedding matrices on the same training data, and is of particular value when training data size is limited. Empirically, one iteration of co-training brings performance improvement in all six datasets on the small *17M* corpus (Table 4.8).

# Chapter 5

# Refining the Notions of Depth and Density in Taxonomy-based Semantic Similarity Measures[1]

## 5.1 Introduction

Semantic similarity measures are widely used in natural language processing for measuring distance between meanings of words. The vector-space models seen in Chapters 3 and 4 often employ large amounts of text as the basis for reliable lexical co-occurrence information. In contrast, another popular approach makes use of hand-crafted information such as dictionaries (Chodorow et al., 1985; Kozima and Ito, 1997) or thesauri (Jarmasz and Szpakowicz, 2003), and often lexical taxonomies such as WordNet (Fellbaum, 1998). Despite their high cost of compilation and limited availability, semantic taxonomies have been widely used in similarity measures under the hypothesis that the complex notion of lexical semantic similarity can be intuitively approximated by the distance between words (represented as nodes) in their hierarchical structures. Even simple methods such as "hop counts" between nodes (e.g., that of

[1]This study was first published as a conference paper in EMNLP 2011 (Wang and Hirst, 2011).

Rada et al., 1989 on the English WordNet) can be highly effective. Meanwhile, taxonomy-based methods have been constantly refined by incorporating various structural features such as depth (Sussna, 1993; Wu and Palmer, 1994), density (Sussna, 1993), the types of connection (Hirst and St-Onge, 1998; Sussna, 1993), word class (sense) frequency estimates (Resnik, 1999), and a combination of these features (Jiang and Conrath, 1997). The major difference among these models is how the features are weighted, however the rationales for adopting these features in the first place remain largely intuitive without much scientific validation. To the best of our knowledge, there is no empirical study directly investigating the effectiveness of adopting structural features such as depth and density. This serves as the major motivation for this study.

### 5.1.1 Taxonomy-based Similarity Measures

Formally, given a node/concept $c$ in WordNet, depth refers to the number of nodes between $c$ and the root of WordNet, (i.e., the root has depth zero, its direct hyponyms depth one, and so on). There are more variations in the definition of density, but it is usually defined as the number of edges leaving $c$ (i.e., its number of child nodes) or leaving its parent node(s) (i.e., its number of sibling nodes). We choose to use the latter since it is the more widely adopted version in the existing literature.

The basic assumption for using the notions of depth and density in WordNet-based semantic similarity measures is that everything else being equal, two nodes are semantically closer if they reside in a region that is (a) deeper in a taxonomy, or (b) more densely connected. This is the working assumption for virtually all WordNet-based semantic similarity studies using depth and/or density. For depth, the intuition is that adjacent nodes deeper in the hierarchy are likely to be conceptually close, since the differentiation is based on more fine-grained nuances (Jiang and Conrath, 1997). Sussna (1993) termed the use of depth as *depth-relative scaling*, claiming that “only-siblings deep in a tree are more closely related than only-siblings higher in the tree”. Richardson and Smeaton (1994) gave an hypothetical example illustrating this "only-siblings" situation, where *plant–animal* are the only two nodes under *living things*, and *wolfhound–foxhound* under *hound*. They claimed the reason that the former pair can be regarded as conceptually farther apart compared to the latter is related to the difference in depth.

As for the relation between density and similarity, the intuition is that if the overall semantic mass for a given node is constant (Jiang and Conrath, 1997), then the more neighbouring nodes there are in a locally connected sub-network, the closer its members are to each other. For example, *animal*, *person*, and *plant* are more strongly connected with *life form* than *aerobe* and *plankton* because the first three words all have high density in their local network structures (Richardson and Smeaton, 1994). Note that the notion of density here is not to be confused with the *conceptual density* used by Agirre and Rigau (1996), which itself is essentially a semantic similarity measure.

In general, both observations on depth and density conform to intuition and are supported qualitatively by several existing studies. The main objective of this study is to empirically examine the validity of this assumption.

## 5.1.2 Semantic Similarity Measures Using Depth and Density

One of the first examples of using depth and density in WordNet-based similarity measures is that of Sussna (1993). The weight $w$ on an edge between two nodes $c_1$ and $c_2$ with relation $r$ (denoted $\rightarrow_r$) in WordNet is given as:

$$w(c_1, c_2) = \frac{w(c_1 \rightarrow_r c_2) + w(c_2 \rightarrow_r c_1)}{2d},$$

where $d$ is the depth of the deeper of the two nodes. As depth increases, weight decreases and similarity in turn increases, conforming to the basic hypothesis. The edge weight was further defined as:

$$w(c_1 \rightarrow_r c_2) = max_r - \frac{max_r - min_r}{n_r(c_1)},$$

where $n_r(X)$ is "the number of relations of type $r$ leaving node $X$", which is essentially an implicit form of density. Here, $max_r$ and $min_r$ are the maximum and minimum of $n_r$ in WordNet, respectively. Note that this formulation of density actually contradicts the basic hypothesis since it is proportional to edge weight (left-hand-side) and thus negatively correlated with similarity.

Wu and Palmer (1994) proposed a concept similarity measure between two concepts $c_1$ and $c_2$ as:

$$sim(c_1, c_2) = \frac{2 \cdot dep(c)}{len(c_1, c) + len(c_2, c) + 2 \cdot dep(c)}, \tag{5.1}$$

where $c$ is the lowest common subsumer (LCS) of $c_1$ and $c_2$, and $len(\cdot, \cdot)$ is the number of edges between two nodes. The basic idea is to adjust "hop count" (the first two terms in the denominator) with the depth of LCS: similarity between nodes with same-level LCS is in negative correlation with hop counts, while given the same hop count, a "deeper" LCS pulls the similarity score closer to 1.

Jiang and Conrath (1997) proposed a hybrid method incorporating depth and density information into an information-content-based model (Resnik, 1999):

$$w(c, p) = \left[\frac{dep(p) + 1}{dep(p)}\right]^{\alpha} \times \left[\beta + (1 - \beta)\frac{\bar{E}}{den(p)}\right] \times [IC(c) - IC(p)]\, T(c, p). \tag{5.2}$$

Here, $p$ and $c$ are parent and child nodes in WordNet, $dep(\cdot)$ and $den(\cdot)$ denote the depth and density of a node, respectively, $\bar{E}$ is the average density over the entire network of WordNet, and $\alpha$ and $\beta$ are two parameters controlling the contribution of depth and density values to the similarity score. $IC(\cdot)$ is the information content of a node based on probability estimates of word classes from a small sense-tagged corpus (Resnik, 1999), and $T(c, p)$ is a link-type factor differentiating different types of relations between $c$ and $p$.

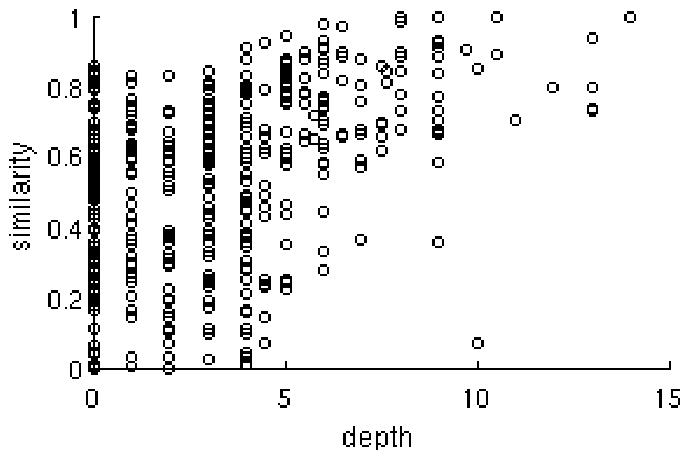


Figure 5.1: Correlation between depth and similarity.

## 5.2 Limitations of the Current Definitions

The objective of this section is to investigate the actual contributions of depth and density actually in establishing an accurate semantic similarity measure. A direct assessment of the effectiveness of using depth and density is to examine their correlation with similarity. Empirical results in this section are obtained using the following definitions. Depth is defined as the number of edges between the lowest common subsumer (LCS) of two nodes under comparison, and density as the number of siblings of the LCS.[2] Similarity benchmark is measured by human judgement on similarity between word pairs. We combine the *RG* and *FG* datasets from Section 4.3.2 in order to maximize data size. Human ratings $r$ on individual sets are normalized to $r_n$ on 0 to 1 scale by the following formula:

$$r_n = \frac{r - r_{\min}}{r_{\max} - r_{\min}}$$

where $r_{\min}$ and $r_{\max}$ are the minimum and maximum of the original ratings, respectively. Correlation is evaluated using *Spearman's* $\rho$.

[2]We also tried several other variants of this definition, e.g., using the maximum or minimum depth of the two nodes instead of the LCS. With respect to statistical significance tests, these variants all gave the same results as our primary definition.

## Depth

The distribution of similarity over depth on the combined dataset is plotted in Figure 5.1. For depth values under 5, similarity scores are fairly evenly distributed over depth, showing no statistical significance in correlation. For depth 5 and above, the distribution exhibits an upper-triangular shape, suggesting that (1) correlation with similarity becomes stronger in this value range of depth, and (2) data points with higher depth values tend to have higher similarity scores, but the reverse of the claim is not necessarily true, i.e., word pairs with "shallower" LCS can also be judged quite similar by humans. This phenomenon can arise in two scenarios. On the one hand, there are similar word pairs that reside shallow in the taxonomy corresponding to the dots in the upper-left corner in Figure 5.1. On the other hand, for word pairs that are highly dissimilar, their LCS is naturally higher up in the taxonomy, resulting in the dots in the lower-left corner. In either case, the observations point to the fact that, contrary to the intuitions behind using the depth of LCS in the existing similarity measures, depth of lower values is not indicative of similarity.

Note that there are much more data points with lower depth values than with higher depth values in the combined dataset. In order to have a fair comparison of statistical significance tests on the two value ranges for depth, we randomly sample an equal number (100) of data points from each value range, and the correlation coefficient between depth and similarity is averaged over 100 of such samplings. Correlation coefficients for depth value under 5 versus 5 and above are $\rho = 0.0881, p \approx 0.1$ and $\rho = 0.3779, p < 0.0001$, respectively, showing an apparent difference in degree of correlation.

Two interesting observations can be made from these results. Firstly, the notion of depth is relative to the distribution of number of nodes over depth value. For example, depth 20 by itself is meaningless since it can be quite large if the majority of nodes in WordNet are of depth 10 or less, or quite small if the majority depth value are 50 or more. According to the histogram of depth values in WordNet (Figure 5.2), the distribution of number of nodes over depth value approximately conforms to a normal distribution $\mathcal{N}(8,2)$. Clearly, the actual

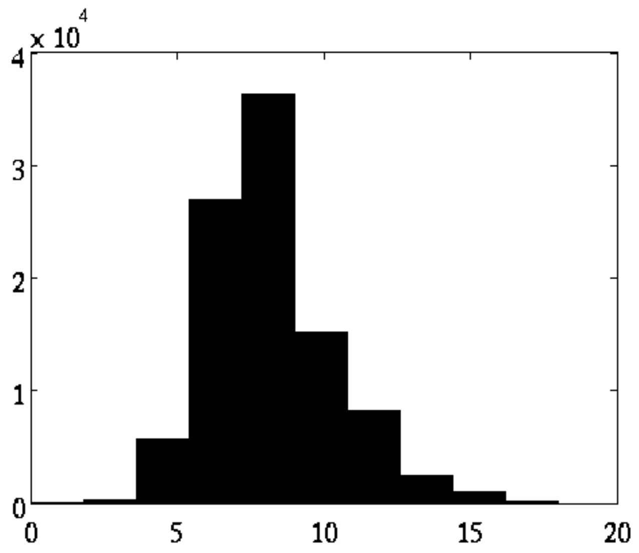


Figure 5.2: Histogram of depth in WordNet.

quantity denoting how deep a node resides in WordNet is conflated at depth values below 5 or above 14. In other words, the distribution makes it rather inaccurate to say, for instance, that a node of depth 4 is twice as deep as a node of depth 2. This might explain the low degree of correlation between similarity and depth under 5 in Figure 5.1 (manifested by the long, vertical stripes spanning across the entire range of similarity scores between 0 and 1), and also how the correlation increases with depth value. Unfortunately, we do not have enough data for depth above 14 to make conclusive remarks on the higher end of the depth spectrum.

Secondly, even on the range of depth values with higher correlation with similarity, there is no definitive sufficient and necessary relation between depth and similarity. The triangular shape of the scatter plot suggests that semantically more similar words are not necessarily deeper in the WordNet hierarchy. Data analysis reveals that the LCS of highly similar words can be quite close to the hierarchical root. Examples include *coast–shore*, which is judged to be very similar by humans (9 on a scale of 0–10 in both datasets). The latter is a hypernym of the former and thus the LCS of the pair, yet it is only four levels below the root node *entity* (via *geological formation*, *object*, and *physical entity*). Another scenario arises due to the fact that no distinction is made between relatedness and similarity in the datasets, and WordNet fails to

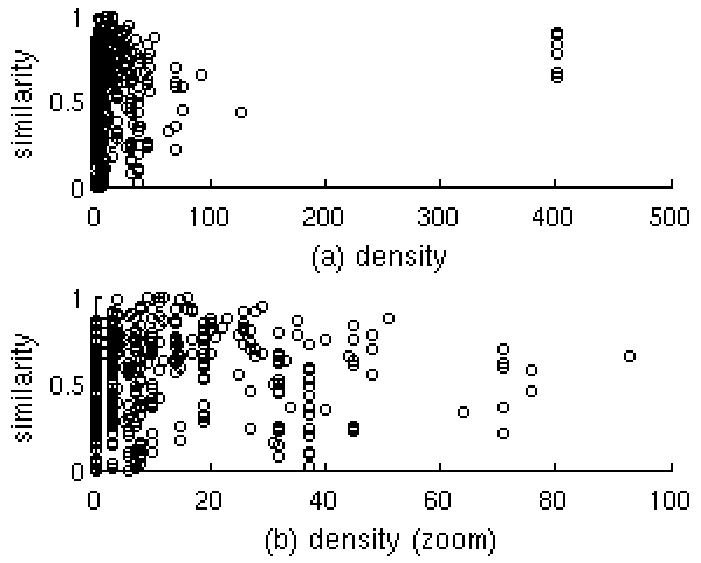


Figure 5.3: Correlation between density and similarity.

capture the relatedness with its hierarchical structure of lexical semantics: the pair *software–computer* can only be related by the root node *entity* as their LCS, although the pair is judged quite "similar" by humans (8.5 on 0 to 10 scale).

The only conclusive claim that can be made about depth and similarity is that word pairs with deeper LCS's tend to be more similar. One exception is the pair *stock–jaguar* in the *FG* set: *stock* is used in the sense of *livestock, stock, farm animal: any animals kept for use or profit*, which is closely connected to *jaguar* through a depth-10 LCS *placental, placental mammal, eutherian, eutherian mammal*. However, human judges gave very low similarity ratings (0.92 on 0–10) due to the relatively lower frequency of this sense of *stock* (especially given the multiple occurrences of the same word under its financial sense in the dataset, e.g., *stock–market*, *company–stock*).

## Density

Comparing to depth, density exhibits much lower correlation with similarity. We conducted correlation experiments between density and similarity with the same setting as for depth and similarity above. Data points with extremely high density values (up to over 400) are mostly

| | *MC* | *RG* | *FG* |
|---|---|---|---|
| *dep* | 0.7056*** | 0.6909*** | 0.3701*** |
| *den* | 0.2268 | 0.2660* | 0.1023 |

Table 5.1: Correlation between depth, density, and similarity on individual datasets. Number of asterisks indicates different confidence intervals ("*" for $p < 0.05$, "***" for $p < 0.0001$).

idiosyncratic to the densely connected regions in WordNet (Figure 5.3-a). Consequently, nodes with density values above 100 are excluded in the experiment (Figure 5.3-b).

Empirically, no correlation is observed between density and similarity on the combined dataset. To confirm the result, we also conduct the experiments on the three individual datasets, and the results are listed in Table 5.1. The correlation coefficient between density and similarity ranges from 0.10 to 0.27. Correlation is statistically significant only on the *RG* dataset ($p = 0.0366$).

Data analysis reveals that density values are often biased by the high degree of connectedness in local regions in WordNet. Qualitatively, Richardson and Smeaton (1994) previously observed that "the irregular densities of links between concepts result in unexpected conceptual distance measures". Empirically, more than 90% of WordNet nodes have density values less than or equal to 3. This means that for 90% of the LCS's, there are only three integer values for density to distinguish the varying degrees of similarity. In other words, such a range might be too narrow to have any real distinguishing power over similarity. In addition, there are outliers with extreme density values particular to the perhaps overly fine-grained sub-categorization of some WordNet concepts, and these nodes can be LCS's of word pairs of drastically different level of similarity. The node {*person, individual*}, for example, can be the LCS of similar pairs such as *man–woman*, as well as quite dissimilar ones such as *boy–sage*, where the large density value does not necessarily indicate high degree of similarity.

Another limitation of the definition of density is the information loss on specificity. If density is adopted to capture the degree of specificity of a concept (i.e., nodes in densely connected regions in WordNet are more specific and thus semantically closer to each other), then this information of a given node should be inherited by its taxonomic descendants, since

| | *MC* | *RG* | *FG* |
|---|---|---|---|
| $dep_u$ | 0.7201*** | 0.6798*** | 0.3751*** |
| $den_u$ | 0.2268 | 0.2660* | 0.1019 |
| $den_i$ | 0.7338*** | 0.6751*** | 0.3445*** |

Table 5.2: Correlation with human judgement of similarity under the new definitions of depth and density.

specificity should monotonically increase as one descends in the hierarchy. For example, the node *piano* has a density value of 15 under the node *percussion instrument*. However, the density value of its hyponyms *grand piano*, *upright piano*, and *mechanical piano*, is only 3. Due to the particular structure of this sub-graph in WordNet, the *grand–upright* pair might be incorrectly regarded as less specific (and thus less similar) than, say, between *piano* and *gong* — both as percussion instruments.

## 5.3 Refining the Definitions of Depth and Density

In this section, we formalize new definitions of depth and density to address their current limitations discussed in Section 5.2.

### Depth

The major problem with the current definition of depth is its failure to take into account the uneven distribution of number of nodes over the depth values. As seen in previous examples, the distribution is rather "flat" on both ends of depth value, which does not preserve the linearity of using the ordinal values of depth and thus introduces much inaccuracy.

To avoid this problem, we "re-curve" depth value to the cumulative distribution. Specifically, if we take the histogram distribution of depth value in Figure 5.2 as a probability density function, our approach is to project cardinal depth values onto its cumulative distribution func-

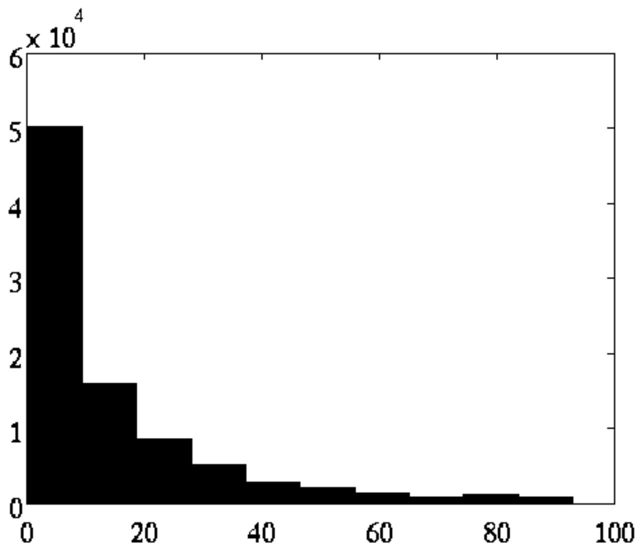


Figure 5.4: Histogram of density in WordNet.

tion. The new depth is denoted $dep_u$ and is defined as:

$$dep_u(c) = \frac{\sum_{c' \in WN} |\{c' : dep(c') \leq dep(c)\}|}{|WN|}$$

Here, $dep(\cdot)$ is the original depth value, $WN$ is the set of all nodes in WordNet, and $|WN|$ is its size. The resulting depth values not only reflect the flat ends, but also preserve linearity for the depth value range in the middle. In comparison with Table 5.1, correlation between $dep_u$ and similarity increases marginally over the original depth values on two of the three datasets (first row in Table 5.2). Later in Section 5.4, we show that these marginal improvements can translate into significant improvements in semantic similarity measures.

### Density

In theory, a procedure analogous to the above cumulative definition can also be applied to density, i.e., by projecting the original values onto the cumulative distribution function. However, due to the Zipfian nature of density's histogram distribution (Figure 5.4, in contrast to Gaussian for depth in Figure 5.2), this is essentially to collapse most density values into a very small number of discrete values (which correspond to the original density of 1 to 3). Experiments show that it does not help in improving correlation with similarity scores (second row in Table 5.2 for $den_u$): correlation remains the same on *MC* and *RG*, and decreases slightly on *FG*.

We therefore resort to addressing the issue of information loss on specificity by inheritance. Intuitively, the idea is to ensure that a node be assigned no less density mass than its parent node(s). In the "piano" example (Section 5.2), the concept *piano* is highly specific due to its large number of siblings under the parent node *percussion instruments*. Consequently, the density of its child nodes *upright piano* and *grand piano* should inherit its specificity on top of their own.

Formally, we propose a recursive definition of density as follows:

$$\begin{aligned} den_i(r) &= 0, \\ den_i(c) &= \frac{\sum_{h \in hyper(c)} den_i(h)}{|hyper(c)|} + den(c), \end{aligned}$$

where $r$ is the root of the WordNet hierarchy (with no hypernym), and $hyper(\cdot)$ is the set of hypernyms of a given concept. The first term is the recursive part normalized over all hypernyms of $c$ in case of multiple inheritance, and the second term is the original value of density.

The resulting density values correlate significantly better with similarity. As shown in row 3 in Table 5.2, the correlation coefficients are about tripled on all three datasets with the new density definition $den_i$, and the significance of correlation is greatly improved from non-correlating or marginally correlating to strongly significantly correlating on all three datasets.

## 5.4 Applying the New Definitions to Semantic Similarity

In this section, we test the effectiveness of the new definitions of depth and density by using them in WordNet-based semantic similarity measures. The two similarity measures we experiment with are that of Wu and Palmer (1994) and Jiang and Conrath (1997). The first one used depth only, and the second one used both depth and density. Note that there are other WordNet-based similarity measures using depth and/or density that we opt to omit for various reasons. Some of them were not designed for the particular task at hand (e.g., that of Sussna, 1993, which gives very poor correlation in similarity task), while others used depth of the entire WordNet hierarchy instead of individual nodes as a scaling factor (e.g., that of Leacock and Chodorow, 1998), which is unsuitable for illustrating the improvement resulting from our new definitions of depth and density.

The task is to correlate the similarity measures with human judgement on similarity between word pairs. We use the same three datasets as in Section 5.2. Correlation coefficient is calculated using *Spearman's* $\rho$. Some earlier studies used parametric tests such as the *Pearson correlation coefficient*. Nonetheless, we believe that the similarity scores of the word pairs in these datasets do not necessarily conform to normal distributions. Rather, we are interested in testing whether the algorithms would give higher scores to pairs that are regarded closer in meaning by human judges. Consequently, a rank-based test like the Spearman's $\rho$ suits better for this scenario.

Parameterization of the weighting of depth and density is a common practice to control their individual contribution to the final similarity score (e.g., $\alpha$ and $\beta$ in Equation (5.2)). Jiang and Conrath already had separate weights in their original study. In order to parameterize depth used by Wu and Palmer in their similarity measure, we also modify Equation (5.1) as follows:

$$sim(c_1, c_2) = \frac{2 \cdot dep^{\alpha}(c)}{len(c_1, c) + len(c_2, c) + 2 \cdot dep^{\alpha}(c)},$$

where depth is raised to the power of $\alpha$ to vary its contribution to the similarity score.

For a number of combinations of the weighting parameters, we report both the best performance and the averaged performance over all the parameter combinations. The latter number is meaningful in that it is a good indication of numerical stability of the parameterization. In

| | Best | | | Average | | |
|---|---|---|---|---|---|---|
| | *MC* | *RB* | *FG* | *MC* | *RB* | *FG* |
| *dep* | 0.7671 | 0.7824 | 0.3773 | 0.7612 | 0.7686 | 0.3660 |
| $dep_u$ | 0.7824 | 0.7912 | 0.3946 | 0.7798 | 0.7810 | 0.3787 |

Table 5.3: Correlation between human judgement and similarity score by Wu and Palmer (1994) using two versions of depth.

| | Best | | | Average | | |
|---|---|---|---|---|---|---|
| | *MC* | *RB* | *FG* | *MC* | *RB* | *FG* |
| $dep, den$ | 0.7875 | 0.8111 | 0.3720 | 0.7689 | 0.7990 | 0.3583 |
| $dep_u$, den | 0.8009 | 0.8181 | 0.3804 | 0.7885 | 0.8032 | 0.3669 |
| $dep, den_i$ | 0.7882 | 0.8199 | 0.3803 | 0.7863 | 0.8102 | 0.3689 |
| $dep_u, den_i$ | 0.8189 | 0.8202 | 0.3818 | 0.8065 | 0.8194 | 0.3715 |

Table 5.4: Correlation between human judgement and similarity score by Jiang and Conrath (1997) using different definitions of depth and density.

addition, parameterization is able to generate multiple correlation coefficients, on which statistical tests can be run in order to show the significance of improvement. We use the range from 0 to 5 with step 1 for $\alpha$ and from 0 to 1 with step 0.1 for $\beta$.

Table 5.3 and 5.4 list the experiment results. In both models, the cumulative definition of depth $dep_u$ consistently improve the performance of the similarity measures. In the Jiang and Conrath (1997) model, where density is applicable, the inheritance-based definition of density $den_i$ also results in better correlation with human judgements. The optimal result is achieved when combining the new definitions of depth and density (row 4 in Table 5.4). For average performance, the improvement of all the new definitions over the original definitions is statistically significant on all three data sets using paired *t-test*.

# Chapter 6

# Conclusions for Part I

When the distributional hypothesis is applied in lexical distributional models, sparsity often poses significant challenges to obtaining efficient lexical representations. In this part of the thesis, some popular techniques for alleviating the issue of sparsity were reviewed, and two dimension reduction techniques and one taxonomy-based approach were proposed and applied to near-synonym lexical choice and lexical similarity.

The latent semantic space representation of near-synonyms and their context allows investigation of several important aspects of the near-synonym lexical choice problem. By employing supervised learning on the latent space features, the representation outperforms all previously proposed models on the "fill-in-the-blank" task (Edmonds, 1997). In addition, the notion of *subtlety* was formalized through the dimensionality of the latent semantic space. Empirical analysis suggests that subtle differences among near-synonyms reside in higher dimensions in the latent semantic space representing non-salient co-occurrence patterns, and that the amount of context needed for proper lexical choice is in positively correlation with the degree of subtlety. These observations are consistent with intuition and are supported by empirical evaluations.

A more fundamental problem underlying the lexical choice task is to derive better vector space representations for words. To this end, we proposed several lexical embedding models that are capable of incorporating syntactic and lexicographic knowledge. The basic premise is that in contrast to the window-based definitions, a syntactically or semantically motivated definition of context (and by extension, relatedness) is able to capture semantic information more accurately due to the lower false-positive and false-negative rates in lexical association. Compared to existing syntax-based embedding models, the proposed embedding models benefit from factorizing syntactic information by individual dependency relations. Empirically, syntactic information from individual dependency types results in notable improvement in model performance at a much faster rate of convergence. Lexicographic knowledge from monolingual dictionaries also helps improve lexical embedding learning, since the defining relation in dictionaries offers accurate lexical association between the definienda and the definientia. Embeddings trained on a highly compact, hand-crafted resource rival state-of-the-art models trained on free texts thousands of times larger in size.

Two additional models were proposed to address the issue of sparsity especially with smaller datasets. Firstly, a relation-independent model was proposed to combine multiple types of syntactic or lexicographic knowledge from individual relations. By training the model to predict the original target word, the model is able to improve the lexical representation by combining information from multiple heterogeneous knowledge sources. A co-training model was also proposed to make use of the semantic information of the dependents in the output embedding matrix, which is usually discarded in existing studies. By repeatedly swapping the input and the output embedding matrices and inverting the governors and dependents in the training data, both matrices showed improvement in capturing lexical semantic information. Empirically, both methods yielded performance improvement in the lexical similarity tasks with training data of smaller sizes.

In addition to vector space models, lexical taxonomies such as WordNet are also frequently used in measuring lexical similarity. We investigated the application of taxonomic features in existing similarity measures of this category and focused on two particular types of features, depth and density, by examining their correlation with human judgements on lexical similarity.

Empirical evidence suggests that depth correlates with similarity only on certain value ranges, while there is no correlation between density and human ratings of similarity.

Further investigation revealed that the problem for the use of depth lies in its simplistic representation as ordinal integer values. The linearity in this representation fails to take into account the conflated quantity of depth in the two extreme ends of its value spectrum. A prominent issue with density, on the other hand, is the information loss on specificity of WordNet concepts, which results in an inaccurate representation of the quantity that is biased by the idiosyncratic structures in densely connected regions in WordNet.

We then proposed new definitions of depth and density to address these issues. For depth, linearity in different value ranges was better modelled by projecting the depth value to its cumulative distribution function. For density, the loss of specificity information was corrected by allowing concepts to inherit specificity information from their taxonomic ancestors. The new definitions resulted in significant improvement in correlation with human judgements of semantic similarity. In addition, when used in existing WordNet-based similarity measures, they consistently improved performance as well as numerical stability of the parameterization of the two features.

# Part II

# Semantic Composition

# Introduction

Techniques developed in Part I can be used to construct computationally and statistically efficient vector space representations on the *lexical* level. In many NLP studies and applications, it is often desirable to *compose* the lexical representations into representations of longer sequences consisting of multiple words. To date, many popular approaches in composition models often resort to simple vector arithmetic such as the additive model or the multiplicative model (Mitchell and Lapata, 2010). These models are widely used in the NLP literature due to good performance in many tasks especially given their simplicity. Empirically, as will be shown in Section 7.4, a randomly initialized neural composition model actually converges to an additive model when trained to learn distributed representations for phrases. As for natural language understanding, these models ignore even the most basic linguistic features, so that there is hardly any resemblance to human behaviour in natural language understanding. For example, multi-word contexts are often regarded as bags of words without taking into account word-order or other linguistic information.

Consequently, the main objective of the second part of the thesis is to address these limitations by (1) developing novel alternatives using neural networks and Bayesian statistics and (2) exploring the use of syntactic and lexicographic knowledge to improve performance in NLP tasks. Specifically, Chapter 7 reviews some popular composition models. Through model visualization, a simple neural composition model for learning phrase representations is shown to converge to a model that closely resembles an additive model. To compose longer texts as sequential input, we adopt a recurrent neural network (RNN) model with long short-term memory (LSTM) units in Chapter 8 and analyse the correlation between its gating behaviour and the lexical distribution at each time step in the input sequence. Confirming our hypothesis, syntactic features including word-order and part-of-speech (POS) information are shown to improve model performance when evaluated on language modelling and the near-synonym lexical choice task. Finally in Chapter 9, a Bayesian compositional similarity model is proposed to measure the similarity between two bags of words without explicitly relying on traditional lexical composition (Wang and Hirst, 2014). Among the proposed models, the near-synonym lexical choice model (Section 8.3) and the unsupervised word sense disambiguation system (Chapter 9) both yield state-of-the-art results in their respective tasks. The syntax-augmented language model (Section 8.2) also outperforms existing models of equivalent complexity with statistical significance.

# Chapter 7

# Learning and Visualizing Composition Models for Phrases

## 7.1 Related Work in Semantic Composition

The goal of semantic composition is to compose representations of shorter linguistic elements (e.g., words) into representations of longer sequences (e.g., phrases, sentences, and documents). Although it has often been argued that the meaning of the whole is more than the meaning of the parts (Lakoff, 1977), composition is nonetheless of great computational advantage (compared to assigning separate representations for the whole), since composition not only avoids the combinatorial complexity of the whole, but also bears closer resemblance to human behaviour in language understanding.

In a highly general form, Mitchell and Lapata (2010) proposed a framework that formalizes composition as a function acting on the component representations **u** and **v** conditioned on their syntactic relation $R$ and additional knowledge $K$:

$$\mathbf{p} = f(\mathbf{u}, \mathbf{v}, R, K),$$

where **p** is the resulting representation of the larger constituent subsuming **u** and **v**. In practice, it is rather challenging to model $R$ and $K$ in composition models while maintaining model performance and scalability. As a result, the most widely-adopted implementations of $f$ are unfortunately often based on simple vector operations on $\mathbf{u}$ and $\mathbf{v}$. For example, an *additive model* uses vector summation to calculate $\mathbf{p}$ as a centroid of the components. A *multiplicative model*, on the other hand, uses element-wise multiplication of the component vectors $\mathbf{u}$ and $\mathbf{v}$, which intuitively resembles an `AND` gate that yields an intersection of the properties in each dimension of the semantic space of the components.

Notably, the input and the output share the same semantic space in both additive and multiplicative models. This is a highly desirable property for improving the scalability of vector space representations since it allows the model to be recursively applied to build representations of longer constituents. For example, to compose vector representations for adjective-noun phrases, Baroni and Zamparelli (2010) proposed to represent adjective modifiers with matrices as a form of transformation applied to their corresponding head nouns represented as vectors. Phrasal composition is formulated as matrix-vector multiplication. Clearly, the resulting phrasal representations are vectors, and it is straightforward to recursively compose representations of longer phrases consisting of one head noun with multiple adjective modifiers. In contrast, partly due to their disadvantages with regard to recursion, many other implementations of $f$ including, e.g., tensor products and circular convolution between the component vectors (Mitchell and Lapata, 2010) haven't gained as much traction as additive and multiplicative models.

An implicit assumption in composition models using simple matrix or vector operations is that all information needed by the composition is encoded in the component vectors, which introduces two major limitations. Linguistically, composition is believed to depend "on the semantic relation being instantiated in a syntactic structure" (Guevara, 2010). According to this view, it follows that the implementation of composition functions should be abstracted from the lexical level representations. Computationally, conflating the composition function with the lexical representations can greatly increase computational cost especially when the lexical space is constructed in the form of matrices or tensors instead of vectors. For example, one strong limitation of the matrix-vector model by Baroni and Zamparelli (2010) is that each adjective is associated with a different matrix, which is computationally rather costly especially when the vocabulary size is large. If the composition function is abstracted from the component representations, it is then possible to adopt representations of lower complexity on the components level.

Many extensions of the simple additive and multiplicative models are developed in the existing literature to address these limitations. For example, Mitchell and Lapata (2008) proposed a more general form of additive models as follows:

$$\mathbf{p} = A\mathbf{u} + B\mathbf{v}.$$

In contrast to the basic additive model, by introducing the parameters $A$ and $B$, composition is regarded as a function that exists independently from the representations of the composing constituents. More importantly, the parameterization makes it possible to encode syntactic information into the composition function. For example, syntax-level composition is implicitly realized by simply conditioning the parameters on the syntactic relation of the components.

Examples of the general additive model include that of Mitchell and Lapata (2008), where a weighted sum $\alpha\mathbf{u} + \beta\mathbf{v}$ was used as the composition function. The model is essentially a special case of the general additive model since it is equivalent to constructing $A$ and $B$ as $\alpha I$ and $\beta I$ with $I$ being the identity matrix. The model was evaluated and compared with several other variants including circular convolution and a hybrid model combining and additive model and a multiplicative model. Examples of abstracting composition function from component representations include that of Guevara (2010), where *partial least square regression* (a supervised regression model) was used to learn phrasal composition functions by minimizing the distance between the composed vectors and the vectors of the observed phrases. Since only adjective-noun phrases were considered in the learning process, the resulting composition function is thus dependent on (and specific to) the syntactic relation in adjective-noun phrases. In comparison, recent studies explicitly used syntactic structures to condition the composition parameters (Socher et al., 2012; Kalchbrenner et al., 2014). These models achieved state-of-the-art performance in some tasks such as sentiment analysis (Tai et al., 2015; Zhu et al., 2015) but are outperformed by simple additive and multiplicative models when evaluated on other tasks such as phrasal similarity and paraphrase classification (Blacoe and Lapata, 2012).

As mentioned earlier, much creativity and engineering is needed to devise computational representations and models of syntactic relations $R$ and additional knowledge $K$ in semantic composition, and many existing studies have taken on the challenge. Erk and Padó (2008), for example, proposed a vector space composition method that adjusts the "expectations" of the model with knowledge on selectional restrictions and selectional preferences derived from dependency parses. Specifically, when two co-occurring words $w_a$ and $w_b$ are semantically related through syntactic dependency relation $r$, the *in-context* representations $a'$ and $b'$ are formulated as follows:

$$a' = (v_a \odot R_b^{-1}, R_a - \{r\}, R_a^{-1}),$$
$$b' = (v_b \odot R_a, R_b, R_b^{-1} - \{r\}),$$

where $v_*$ is *context-less* representation of a word, $\odot$ is a vector combination function such as the additive or the multiplicative model, and $R_*$ is the set of syntactic relations in which a word participates in the given context. Essentially, composition is implemented by consolidating the actual and the expected distributional profiles of the target words: the actual distributions are encoded in the traditional vector representations $v_*$, and the expectations are expressed through the selectional restrictions or preferences through the dependency relations such as $R_b^{-1}$ and $R_a$. The model was evaluated on a lexical similarity rating task (Mitchell and Lapata, 2008) and a lexical substitution task (McCarthy and Navigli, 2009) and was shown to outperform simple composition functions such as that of Mitchell and Lapata (2008) with statistical significance.

More recently, many composition models have been developed using artificial neural networks. For example, Kalchbrenner et al. (2014) used a Convolutional Neural Network (CNN) to model sentences, which was applied to sentiment analysis and question type classification. Hermann and Blunsom (2013) proposed to use Recursive Autoencoders (RAEs) to model linguistic knowledge represented by Combinatory Categorical Grammar (CCG). Several variants were proposed that differed mainly by the parameterization of the RAEs, for example, by sharing the trainable weights globally or by conditioning the weights on different aspects of the CCG representation such as the rules, the syntactic categories of the generated constituents, and the syntactic categories or the component constituents in each application of the rules. Evaluation was performed on both sentiment analysis and phrasal similarity tasks. The variant with shared parameters are consistently outperformed by the more complex variants, although among the latter, there does seem to exist a clear distinction in model performance. As mentioned in Section 1.3, recurrent neural networks (RNNs) have also been a popular choice for semantic composition (Socher et al., 2012; Tai et al., 2015; Zhu et al., 2015). Many of these existing studies rely on the hierarchical structures of parse trees for incorporating linguistic information, and the fundamentally difference of the RNN models proposed later in this thesis is that our proposed models use simple syntactic knowledge such as syntactic categories in text sequences without making structural assumptions about the input.

## 7.2 Phrasal Embedding Learning

In this study, we propose a composition model to learn distributed representations (embeddings) for phrases with a prescribed set of syntactic structures. The model is developed under the following assumptions. Firstly, there exist annotations that identify target phrases in the training corpus. For a sentence $s$ consisting of $n$ words $w_1, \dots, w_n$, a phrase $p$ of length $m < n$ within the sentence is marked by aliasing the composing words $w_i, \dots, w_{i+m-1}$ as $p_1, \dots, p_m$, respectively, where $i \in \{1, \dots, n-m+1\}$ is the left boundary of $p$ within $s$. Secondly, the model is assumed to have access to the syntactic structure of $p$. For example, the syntactic structure for the phrase *book store* is marked as noun compound, and that for *intriguing plot* as modifier-noun, etc. All such syntactic structures constitute a finite set $\mathscr{S}$. Finally, lexical embeddings are assumed to be available for both the composing words $w_i, \ldots, w_{i+m-1}$ of $p$ and the context words $s \setminus p$ in sentence $s$. Suppose the dimensionality of the lexical embeddings is $d$, then a composition function $f : \mathscr{S} \times \mathbb{R}^{d \times m} \mapsto \mathbb{R}^d$ takes the syntactic structure of a phrase and the lexical embeddings of its composing words and composes them into a phrasal embedding that is of the same dimensionality $d$ as the lexical embeddings.

Due to our particular interests in examining what exactly is being learned by a distributional composition model, $f$ is implemented with a simple neural network to facilitate visualization. On the lexical level, each word $w_i$ is assigned a distributed representation $\mathbf{e}_i$. When $m$ words $w_1, \ldots, w_m$ form a phrase with syntactic structure $syn$, the composition function is defined as follows:

$$f(syn, \mathbf{e}_1, \ldots, \mathbf{e}_m) = \sigma(W_{syn} \cdot \langle \mathbf{e}_1^T, \ldots, \mathbf{e}_m^T \rangle^T) + \mathbf{b}_{syn}, \tag{7.1}$$

where $\sigma$ is the sigmoid function, $W_{syn}$ is a $d$ by $d \times m$ weight matrix, $\langle \cdot \rangle$ stands for matrix concatenation, and $\mathbf{b}_{syn} \in \mathbb{R}^d$ is the bias term.

During training, when the model encounters a phrase $p$, it retrieves the appropriate weight matrix and bias term according to the syntactic structure of $p$, as well as the lexical embeddings of the composing words. With this information, the model then calculates the phrasal embedding $\mathbf{e}^p$ using Equation (7.1). Similar to `word2vec`, parameter updates are calculated with a contrastive divergence objective given the lexical embeddings of the context words:

$$\log \sigma((\mathbf{e}^p)^T \mathbf{e}'_c) + \sum_{i=1}^{k} \mathbf{E}_{\hat{c}_i}[\log \sigma(-(\mathbf{e}^p)^T \mathbf{e}'_{\hat{c}_i})], \tag{7.2}$$

where $\mathbf{e}'_*$ is the output embedding of context word $w_c$, and the subscripts $\hat{c}_i$ refer to the embeddings of negative samples (drawn according to unigram frequencies). The remaining notations

| Types | verb-noun | noun-noun | adjective-noun |
|---|---|---|---|
| Positive | lose interest | food shortage | last season |
| | build engine | labor group | nuclear test |
| | defeat indifference | finance minister | political system |
| Negative | existing machine | (of) course life | slow turning (pitch) |
| | hitting streak | morning radio (show) | global ad (agency) |
| | governing coalition | repeated trip | red university (jacket) |

Table 7.1: Positive and negative examples of phrases extracted from the Annotated Gigaword Corpus using POS bigram heuristics.

bear the same meaning as in Equation (4.1).

## 7.3 Phrase Generation

The Annotated English Gigaword Corpus (Napoles et al., 2012) is used for generating the phrasal data. Using the annotations provided by the corpus, POS bigrams are identified to compose the three types of phrases used by Mitchell and Lapata (2010). For example, if the POS tags of two adjacent words are adjective and noun (in this particular order), then the words are proposed as an adjective-noun-type phrase. The same rule applies to verb-noun and noun-noun bigrams. Manual data inspection of a random, small sample of the extracted phrases reveals that the majority of them are grammatically valid and semantically meaningful, and that exceptions mainly come from either complex syntactic structures or POS tagging errors (Table 7.1). For example, most of the verb-noun errors are caused by the POS tagger's tendency to tag the adjectival participles as verbs. The compound nouns and adjective-noun phrases, on the other hand, often suffer from incorrectly extracted phrases out of larger constituents.

## 7.4 Composition Function Visualization

To investigate what type of knowledge the composition models have acquired, the weight matrices of the models are plotted with heat maps in Figure 7.1. The x-axis corresponds to the input dimensions, i.e., the dimensions of the concatenated lexical embeddings of the composing words of a phrase (Equation (7.2)), and the y-axis corresponds to the dimensions of the phrasal embeddings (i.e., the output of the composition function). The colour temperature indicates the amplitude of the cell values in the matrices. Sub-figure (a) shows that cell values for the verb-noun composition function are centred around zero upon random initialization.[1] After training, the composition functions corresponding to all three relations exhibit similar patterns as shown in Sub-figure (b) to (d). Note that for two-word phrases, $m = 2$ and thus a weight matrix $W_* \in \mathbb{R}^{d\times 2d}$ is a horizontal concatenation of two square matrices. Cell values are notably larger on the diagonals of both square matrices, as indicated by the bright lines across these diagonals against the colder-coloured background. Numerically, the resulting composition functions essentially converge to a state resembling a weighted additive model. In other words, the distributed representation of a phrase as a whole is empirically best approximated by the sum of the lexical embeddings of its composing words, suggesting a high degree of semantic compositionality for the phrases annotated in the training data.

The lack of uniformity of the cell values distinguishes the resulting composition functions from an additive model, which concatenates two identity matrices instead of near-diagonal matrices. The variance in cell values differs across different syntactic relations of the phrases. For example, the first word in a verb-noun phrase is the head verb, and the first word in a noun-noun phrase is the nominal dependent. During composition, the concatenation in Equation (7.1) dictates that the embedding of the first word is transformed by the left-most square matrix in $W_*$. By comparing the colour temperature of the diagonals on the left, it appears that the average value of the weights for the head verbs in Sub-figure (b) is noticeably smaller (i.e., in darker

[1]This is the case for all three syntactic relations of the phrases considered and thus the figures for the other two relations are omitted.

colours) than that for the nominal dependents in Sub-figure (c). The difference indicates that when minimizing the difference between the phrasal embeddings and those of the surrounding contexts, the nominal dependents in noun-noun phrases are considered semantically more salient than the head verbs in verb-noun phrases. From the limited examples in Table 7.1, the difference conforms to linguistic intuition, e.g., the verbs *lose*, *build*, and *defeat* do not carry as much semantic content as the noun modifiers *food*, *labor*, and *finance* with regard to topicality and semantic cohesion.

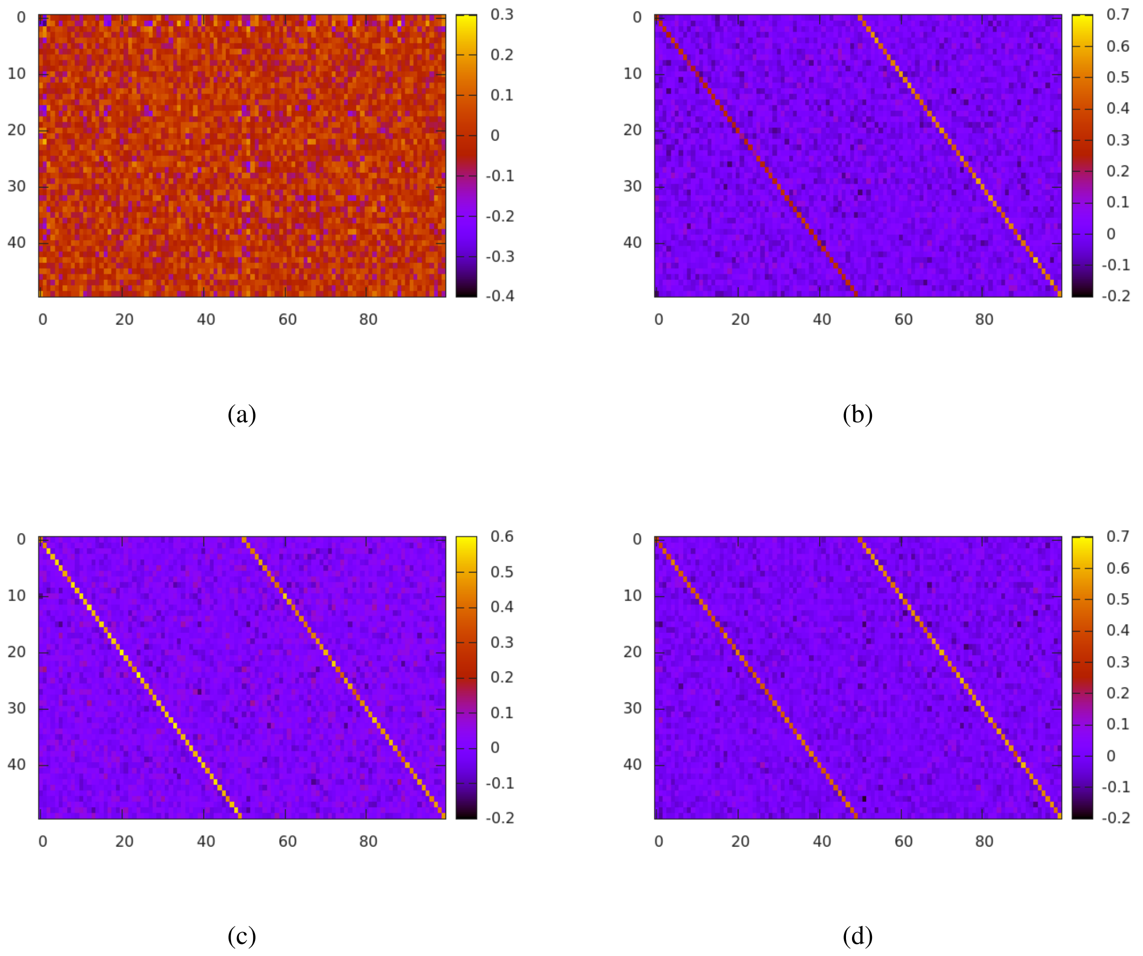


Figure 7.1: Visualization of the weight matrices in phrasal composition functions. (a) Randomly initialized weights for verb-noun phrases (similar for the other two syntactic relations). (b–d) Model weights approximating an additive model after contrastive divergence training for: (b) verb-noun construct, (c) noun-noun construct, and (d) adjective-noun construct.

# Chapter 8

# Composition in Language Modelling and Near-synonymy Lexical Choice Using Recurrent Neural Networks

One of the major limitations of additive models is the simplistic treatment of the input data ignoring even the basic linguistic properties such as word-order. In recurrent neural networks (RNNs), on the other hand, word-order information is preserved by using texts as sequential input, which makes RNNs a natural choice to address the limitations.

Much success has been achieved with RNNs in many areas in NLP including machine translation (Sutskever et al., 2014), language modelling (Zaremba et al., 2014), sentiment analysis (Tai et al., 2015; Zhu et al., 2015), etc. In this study, RNNs are used as a composition model for multi-word contexts. A particular variant of RNNs with long short-term memory (LSTM) is used to avoid the problem of exploding or vanishing gradients — a phenomenon where gradients either diminish to zero or diverge to infinity during long time spans. The main hypothesis is that effectiveness of gradient propagation in LSTM can be improved by augmenting the input sequence with syntactic knowledge. We show that the gating behaviour of LSTM units can be affected by the distribution of syntactic functions over each time step in the textual sequences. Empirical evaluation confirms the hypothesis with performance improvement in both language modelling and near-synonym lexical choice tasks when adding syntactic knowledge to the LSTM models.

## 8.1 Gradient Analysis in LSTM Optimization

### 8.1.1 Gradients of Traditional RNNs

The defining feature of an RNN model is that the model's historical output (or hidden states) $\mathbf{y}$ from the previous time step $t-1$ is fed back to the model input at time $t$:

$$\mathbf{y}^{(t)} = f(A\mathbf{x}^{(t)} + B\mathbf{y}^{(t-1)} + \mathbf{b}), \tag{8.1}$$

where $A$ and $B$ are matrices of appropriate dimensions to perform linear transformation on the inputs, $\mathbf{b}$ is the bias term, and $f$ is a non-linear activation function. The feedback loop from $\mathbf{y}^{(t-1)}$ to $\mathbf{y}^{(t)}$ resembles that in dynamic systems (Luenberger, 1979). As with any deep neural network architecture, the problem of vanishing or exploding gradients occurs when a large number of time steps are involved (Hochreiter, 1998). Specifically, suppose the loss function $E$ is the *mean square error* (MSE):

$$E(\mathbf{y}^{(t)}) = \frac{1}{2}(\mathbf{y}^{(t)} - \hat{\mathbf{y}}^{(t)})^2,$$

where $\hat{\mathbf{y}}^{(t)}$ is the desired output at time $t$. The gradient (w.r.t. parameters, collectively denoted as $\theta$) at time $t$ is calculated by the following equation:

$$\frac{\partial E}{\partial \theta} = \frac{\partial E}{\partial \mathbf{y}^{(t)}} \frac{\partial \mathbf{y}^{(t)}}{\partial \theta} = (\mathbf{y}^{(t)} - \hat{\mathbf{y}}^{(t)}) \frac{\partial \mathbf{y}^{(t)}}{\partial \mathbf{y}^{(t-1)}} \cdots \frac{\partial \mathbf{y}^{(2)}}{\partial \mathbf{y}^{(1)}} \frac{\partial \mathbf{y}^{(1)}}{\partial \theta}.$$

Each of the recursive terms can be evaluated as:

$$\frac{\partial \mathbf{y}^{(\tau)}}{\partial \mathbf{y}^{(\tau-1)}} = f'(\cdot)B, \forall \tau = 2, \ldots, t.$$

When the activation function $f$ is omitted for simplicity, the gradient becomes:

$$\frac{\partial E}{\partial \theta} = (\mathbf{y}^{(t)} - \hat{\mathbf{y}}^{(t)})B^{t-1}\frac{\partial}{\partial \theta}(A\mathbf{x}^{(1)} + B\mathbf{y}^{(0)} + b) = \begin{cases} (\mathbf{y}^{(t)} - \hat{\mathbf{y}}^{(t)})B^{t-1}\mathbf{x}^{(1)}, \text{when } \theta = \text{A} \\ (\mathbf{y}^{(t)} - \hat{\mathbf{y}}^{(t)})B^{t-1}\mathbf{y}^{(0)}, \text{when } \theta = \text{B} \end{cases} \tag{8.2}$$

When $\|B\| \neq 0$, as $t$ increases, the exponential term $B^{t-1}$ makes the gradient either diminish to zero (and thus the parameters are no longer updated) or diverge to infinity (causing numerical instability in training).

### 8.1.2 Gradients of LSTM Units

One way to resolve the problem of vanishing or exploding gradients is to augment a traditional RNN model with LSTM units (Gers et al., 2000) to control the information flow in the model using three gates: an *input gate* to control the data influx into the model from the previous states, a *forget gate* to decide whether to accumulate or reset the LSTM memory cells, and an *output gate* to control what to emit as output. Formally, the gates are implemented as follows:

$$G_i = \sigma(w_{ix}\mathbf{x} + w_{iy}\mathbf{y}^{(t-1)} + b_i) \tag{8.3}$$

$$G_f = \sigma(w_{fx}\mathbf{x} + w_{fy}\mathbf{y}^{(t-1)} + b_f) \tag{8.4}$$

$$G_o = \sigma(w_{ox}\mathbf{x} + w_{oy}\mathbf{y}^{(t-1)} + b_o) \tag{8.5}$$

$$\tilde{\mathbf{c}} = \tanh(w_{cx}\mathbf{x} + w_{cy}\mathbf{y}^{(t-1)} + b_c) \tag{8.6}$$

$$\mathbf{c}^{(t)} = G_i \odot \tilde{\mathbf{c}} + G_f \odot \mathbf{c}^{(t-1)} \tag{8.7}$$

$$\mathbf{y}^{(t)} = G_o \odot \mathbf{c}^{(t)} \tag{8.8}$$

where $\sigma$ is the sigmoid function, tanh is the hyperbolic tangent function, and $\odot$ is the Hadamard (element-wise) product between vectors.[1] Equations (8.3) to (8.5) define the input gate $G_i$, the forget gate $G_f$, and the output gate $G_o$, respectively. The activation of each gate is determined jointly by the current input $\mathbf{x}$ and the recurrent state from the previous time step $\mathbf{y}^{(t-1)}$. Equation (8.6) defines the *cell state*, which is used to derive the *memory cell* $\mathbf{c}$ in Equation (8.7). Recurrence also exists in the definition of the memory cell, which is a linear combination of its historical value from the previous time step and the weighted input. The forget gate and the input gate are used as weights in the combination. The current memory $\mathbf{c}^{(t)}$, gated by the output gate $G_o$, is used to calculate the final output (or state) of the model in Equation (8.8).

In the original study by Gers et al. (2000), where the forget gate was first introduced, the gradient was truncated as soon as it leaves the memory cell. In other words, it is only in the memory cells that error messages are back-propagated through multiple time steps. Graves and Schmidhuber (2005) argued that the truncation not only hurts model performance but also makes it difficult to perform numerical verification of the derivatives. To address these issues, an alternative approach was proposed in the latter study to calculate the gradients across multiple time steps, which was empirically shown to improve model performance in a phoneme classification task.

The motivation for a full gradient over multiple time steps is rooted in the very motivation for RNN models: if a historical value of a particular parameter is believed to affect inference at a future time step, then the model parameter at that historical time step should also be updated by the relevant learning signal generated at that future time step. Take the language modelling task for example. Given an input sequence between time steps $\tau_0$ and $\tau_1$, the weight matrix $w_{ix}$ in the input gate at time $\tau_0$ will affect the prediction of the word at $\tau_1$. Consequently, when a learning signal is generated at $\tau_1$, it should be back-propagated through multiple time steps back to $\tau_0$ for updating $w_{ix}$ in hope for a more accurate influence in similar situations in future training.

[1]For simplicity, temporal superscripts are omitted whenever obvious (e.g., on $\mathbf{x}$ and the gates $G_*$).

Nonetheless, we suspect that the multi-time-step propagation formulated by Graves and Schmidhuber (2005) might be erroneous. This is because in their work, the parameter updates for the weight matrix $w_{ij}$ between time steps $\tau_0$ and $\tau_1$ proposed is calculated as follows:

$$\nabla_{ij}(S) = \sum_{\tau=\tau_0+1}^{\tau_1} \frac{\partial E}{\partial x_i(\tau)} \frac{\partial x_i(\tau)}{\partial w_{ij}} = \sum_{\tau=\tau_0+1}^{\tau_1} \delta_i(\tau) y_j(\tau-1).$$

Since the first terms in the summations were previously defined as being equal:

$$\frac{\partial E}{\partial x_i(\tau)} = \delta_i(\tau),$$

it follows that the following equality is implied for the second terms, i.e.:

$$\frac{\partial x_i(\tau)}{\partial w_{ij}} = y_j(\tau-1). \tag{8.9}$$

Equation (8.9) appears to be correct especially given the following definition of $x_i(\tau)$:

$$x_i(\tau) = \sum_{j \in N} w_{ij} y_j(\tau-1) + \sum_{c \in C} w_{ic} s_c(\tau-1).$$

However, since $N$ includes cell outputs, $y_j(\tau-1)$ is in fact dependent on $w_{ij}$ through recurrence, resulting in a non-linear relation between $x_i$ and $w_{ij}$ and thus compromising the validity of Equation (8.9).

This type of dependency turns out to be rather ubiquitous in deriving the full gradients for LSTM. For example, given a loss function $E$, the gradient with respect to the the weight matrix $w_{ox}$ in the output gate $G_o$ can be defined as:

$$\frac{\partial E}{\partial w_{ox}} = \frac{\partial E}{\partial \mathbf{y}^{(t)}} \frac{\partial \mathbf{y}^{(t)}}{\partial w_{ox}}. \tag{8.10}$$

The second term $\frac{\partial \mathbf{y}^{(t)}}{\partial w_{ox}}$ on the RHS cannot be easily calculated because both $G_o$ and $\mathbf{c}^{(t)}$ depend on $w_{ox}$ (Equation (8.8)). To resolve this issue, we propose to perform a full expansion of the gradients as follows. Take the weight matrices $w_{\alpha x}$ for example, which is the weight matrix applied to the input $\mathbf{x}^{(t)}$ for gate $\alpha \in \{i, f, o, c\}$. According to Equation (8.8), both terms in the model output $\mathbf{y}^{(t)}$ depend on $w_{\alpha x}$, the derivative of the model output w.r.t. this matrix is defined as:

$$\frac{\partial \mathbf{y}^{(t)}}{\partial w_{\alpha x}} = G_o \frac{\partial \mathbf{c}^{(t)}}{\partial w_{\alpha x}} + \mathbf{c}^{(t)} \frac{\partial G_o}{\partial w_{\alpha x}}. \tag{8.11}$$

According to Equation (8.7), the first derivative term is expanded as:

$$\frac{\partial \mathbf{c}^{(t)}}{\partial w_{\alpha x}} = G_i \frac{\partial \tilde{\mathbf{c}}^{(t)}}{\partial w_{\alpha x}} + \tilde{\mathbf{c}}^{(t)} \frac{\partial G_i}{\partial w_{\alpha x}} + G_f \frac{\partial \mathbf{c}^{(t-1)}}{\partial w_{\alpha x}} + \mathbf{c}^{(t-1)} \frac{\partial G_f}{\partial w_{\alpha x}}. \tag{8.12}$$

Note that the third derivative term is an inductive term of the LHS of the equation. For the other derivatives, the dependent variables all have analogous non-linear relations with the independent variable $w_{\alpha x}$ by recurrence through $\partial \mathbf{y}^{(t)}$, which can be used as inductive terms of Equation (8.11) to ground the gradients. For notational convenience, denote $G_c = \tilde{\mathbf{c}}^{(t)}$ (i.e., referring to the cell-state input as a gate), then for each gate $G_\beta, \forall \beta \in \{i, f, o, c\}$, the gradients w.r.t. $w_{\alpha x}$ can be defined as:

$$\begin{gathered}\frac{\partial G_\beta}{\partial w_{\alpha x}} = G'_\beta \left(\frac{\partial w_{\beta x}\mathbf{x}}{\partial w_{\alpha x}} + w_{\beta y} \frac{\partial \mathbf{y}^{(t-1)}}{\partial w_{\alpha x}}\right) = \begin{cases} G'_\beta (\mathbf{x} + w_{\beta y} \frac{\partial \mathbf{y}^{(t-1)}}{\partial w_{\alpha x}}), \text{when } \alpha == \beta \\ G'_\beta w_{\beta y} \frac{\partial \mathbf{y}^{(t-1)}}{\partial w_{\alpha x}}, \text{otherwise} \end{cases} \\ G'_\beta = \begin{cases} 1 - G_\beta^2, \text{when } \beta == \text{c} \\ G_\beta (1 - G_\beta), \text{otherwise} \end{cases}\end{gathered} \tag{8.13}$$

By substituting corresponding terms in Equation (8.10) with Equation (8.11) to (8.13), the final result is essentially a dovetailing recursion between $\frac{\partial \mathbf{y}^{(t)}}{\partial w_{\alpha x}}$ and $\frac{\partial \mathbf{c}^{(t)}}{\partial w_{\alpha x}}$. The base cases for the induction are $\frac{\partial \mathbf{y}^{(0)}}{\partial w_{\alpha x}}$ and $\frac{\partial \mathbf{c}^{(0)}}{\partial w_{\alpha x}}$, respectively, both of which are zero since the initial LSTM state and cell state are independent of the parameter $w_{\alpha x}$.

### 8.1.3 Gradient Verification

We adopt two independent approaches to verify the correctness of the above derivation: by manually performing gradient checking and by automatic gradient calculation using a symbolic algebraic system.

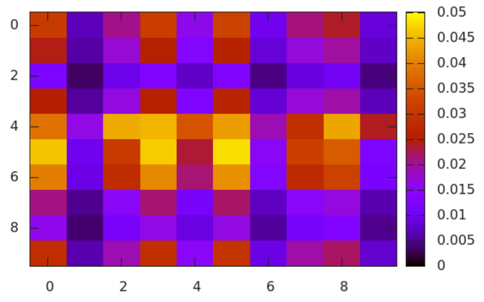
Gradient checking is essentially a numerical approximation of differentiation. Given an objective function $J(x)$, after adding a small perturbation $\varepsilon$ to the input $x$, the derivative $\frac{dJ}{dx}$ can then be approximated by the perturbation in the output:

$$\frac{dJ}{dx} \approx \frac{J(x+\varepsilon)}{2\varepsilon}.$$

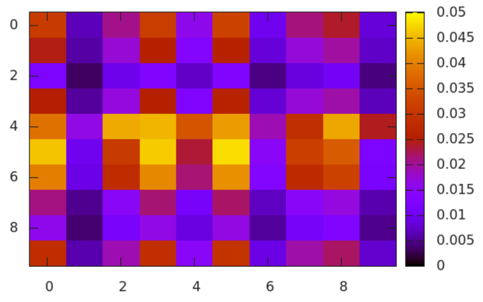
By definition of derivatives, precision of the approximation increases as $\varepsilon$ approaches zero. To apply gradient checking to our problem at hand, i.e., to calculate the gradient of an objective function w.r.t. a certain weight matrix, we simply add the perturbation to each cell in the weight matrix and observe the response in the objective function output.

The second approach is implemented using the Symbolic Differentiation module in *Theano* (Bergstra et al., 2010), a software library capable of evaluating mathematical expressions involving mult-dimensional arrays. The module makes it possible to compute partial derivatives between any dependent-independent variable pair in a given graphical model (in our case, between the objective function and the weight matrices in the LSTM model).

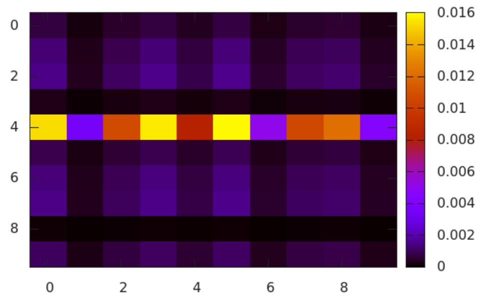
We use manual gradient checking and the symbolic gradients from *Theano* as independent ground truth to evaluate the correctness of the gradient derivation of Graves and Schmidhuber (2005) and derivation presented in Section 8.1.2. The experiments are done on a network of 10 hidden units and 10 input dimensions, and the results are plotted in Figure 8.1. Our derivation is clearly more consistent with both ground truth methods than that of Graves and Schmidhuber (2005). The visual verification also shows that the amplitude of the gradients by Graves and Schmidhuber (2005) is in general lower than the correct values, which may be explained by the missing channels from back-propagation such as the recurrent channels that we suspect are



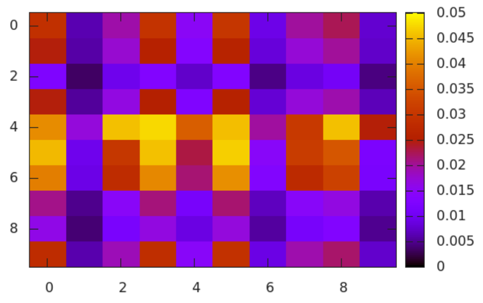
(a)



(b)



(c)



(d)

Figure 8.1: Visual verification of full gradients in an LSTM model with input and output dimension of 10. (a) Symbolic gradients calculated using *Theano*. (b) Numerical approximations using input-output perturbations. (c) Gradients derived by Graves and Schmidhuber (2005). (d) Gradients derived in Section 8.1.3.

erroneous.[2]

### 8.1.4 Regulating LSTM Gating Behaviour with Syntactic Constituents

The LSTM architecture dictates that the efficiency of parameter updates are affected by the gating behaviour of all three gates defined in the model. In sequential text data, since time steps correspond to word positions, a beneficial but implicit objective of LSTM is then to learn to segment a lengthy sequence (i.e., with a large number of time steps) into sub-sequences that

[2]The source code is available at `https://github.com/wangtong106/scribbles/blob/master/lstm.py`.

satisfy all three conditions. The intuition is that natural language sentences can often be "chunked" into multiple independent sub-sequences (e.g., corresponding to phrases or *linguistically meaningful constituents*). Although all chunks contribute to the overall meaning of the sentence, individual constituents can still be semantically self-contained and thus exhibit relative independence from each other. Ideally, an LSTM model is able to (1) identify the constituent boundaries, (2) "trap" the gradients within these constituents, and (3) release and reset the memory content when crossing the constituents boundaries. In other words, a properly trained LSTM must be able to identify the constituents by learning to open and close the gates at appropriate times — somewhat resembling how humans read with ad-hoc chunking and parsing.

The discussion above motivates two major hypotheses of this study. Firstly, it is desirable for the constituents identified by LSTM to coincide with a constituent that is semantically and syntactically meaningful (in contrast to, say, a sub-sequence that spans across two adjacent but different constituents). The reason for this desideratum is that gradient information (and hence the learning signal) is then able to be propagated through the entire constituent and only within the same constituent. With this condition satisfied, the sequence-final output of the model (which is often used as the compositional representation of the entire sequence) will be derived from the recurrence of one or more intermediate states corresponding to meaningful constituents rather than random sub-sequences.

Secondly, we hypothesize that adding syntactic knowledge to LSTM units can help the model to identify meaningful constituents. Intuitively, this hypothesis is supported by the fact that at any given time step (word position), the distribution of the syntactic information of words is usually of much lower entropy than the lexical distribution. For example, it is relatively easier to predict the POS of the word immediately preceding a noun than to predict the actual word. Similarly, if the model has access to some form of syntactic knowledge about the lexical input, it would be easier for the model to identify the boundaries of constituents than with the lexical information only. Empirical evidence will be presented in later sections through performance improvement in language modelling and near-synonym lexical choice.

## 8.2 Language Modelling Using Syntax-Augmented LSTM

### 8.2.1 Model Implementation

An LSTM-based language model is used in this section to test the hypothesis on the benefits of incorporating syntactic information. According to Equation (8.3) to (8.5), the gate activations are jointly determined by the historical states and the input at each time step. The cumulative nature of the states introduces much recurrent latency and hence difficulty to control, but it is relatively easy to incorporate syntactic knowledge into the input of the model.

Specifically, the embedding matrix in LSTM consists of columns of distributed lexical representations, each corresponding to a lexical input **x**. To add POS as syntactic information, a second embedding matrix $P$ is added to the model consisting of POS representations. For each occurrence of **x** in Equations (8.3) to (8.6), the POS embedding corresponding to the POS of **x** is retrieved and added to the model with a linear transformation. The new definition of the input gate is as follows:

$$G_i = \sigma(U_i\mathbf{x} + W_i \cdot \mathbf{p}_x + V_i\mathbf{y}^{(t-1)} + b_i),$$

where $\mathbf{p}_x \in P$ is the embedding of the POS of **x**, and $W_i$ is the additional parameter to model the POS information in the input gate. The linear functions $W_*$ differ for each gate as well as the weighted input. However, due to the small number of POS types, it is sufficient to have a very low dimensionality for the POS embeddings and thus the size of $W_*$ is small and does not impose much burden with regard to model complexity or computational cost.

Note that the model requires access to POS information at both training and test time. On the one hand, it can be argued that the resulting model is using more information on the input in order to achieve performance improvement. On the other hand, however, a particular strength of the proposed model is its ability to consolidate and take advantage of the additional linguistic information. The same observation can be made in many existing language modelling studies that either explicitly affix additional features to the model input (Ghosh et al., 2016) or implicitly incorporate external information such as similarity between input words (Dagan et al., 1999). In addition, POS information can be easily obtained with simple tools or even heuristics with fairly high accuracy, which does not impose much dependency on external linguistic resources on the proposed model.

### 8.2.2 Evaluation

In this section, we evaluate the effect of adding POS information to LSTM with a language modelling task. Following the conventional setup in language modelling experiments, the *Penn Tree Bank* (*PTB*) 2 dataset (Marcus et al., 1993) is used, with Sections 00–20 (912.8k words) for training, Sections 21–22 (95.3k words) for model validation, and Sections 23–24 (99.3k words) for testing. The texts are tagged with a stochastic POS tagger (Church, 1988) and the POS information is included in the tagged version of the *PTB2* release.

Two network structures are used to compare with the results reported by Zaremba et al. (2014): a small, non-regularized model consisting of 200 hidden units trained for 13 epochs, and a large, regularized model with 1500 hidden units trained for 55 epochs. Two LSTM layers are used in the larger network, with the output of the lower layer connected to the input of the upper layer at each time step. Regularization is implemented with a dropout rate of 0.65 (Srivastava et al., 2014), which is applied only between the time steps within the same layer but not in between the LSTM layers. In both models, the dimensionality of the POS embeddings is chosen to be 10% of the number of hidden units, and the lexical embedding dimensions are reduced accordingly to preserve the total number of hidden units and maintain model complexity for fair comparison. For example, in the small network, the 200 dimensions for the hidden units consist of 180 dimensions for the lexical embeddings and 20 dimensions for the POS embeddings.

Experiment results are listed in Table 8.1. `LEX` represents models with lexical features

| Network size | 200 | | 1500 | |
|---|---|---|---|---|
| Dataset | Val | Test | Val | Test |
| LEX | 120.7 | 114.5 | 82.2 | 78.4 |
| SYN | 110.8 | 105.7 | 77.5 | 75.2 |

Table 8.1: Incorporating POS information into LSTM input helps reduce language modelling perplexity on the Penn Tree Bank 2 dataset. Perplexity scores are averaged across 12 runs of the corresponding models using different seeds for randomization. The differences between the baseline and the proposed models are statistically highly significant ($p < 0.0001$) according to $t$ test between the means.

only (Zaremba et al., 2014) and `SYN` corresponds to models with syntactic (POS) information incorporated into the LSTM inputs. On both the small and the large models, POS information helps achieve notably better results on the *PTB2* data. On both the validation and the test data, larger improvement is observed for the smaller model than the larger one. One explanation is that the larger model might be better at implicitly inferring the syntactic information from the lexical input than the smaller model — either due to the higher capacity of the model or the dropout regularization.

## 8.3 Near-synonymy Context Composition Using LSTM

The motivation for applying RNN-LSTM to near-synonym lexical choice (NSLC) is two-fold. Firstly, as mentioned in Section 3.2, existing lexical choice models almost always employ the additive model to represent multi-word contexts of near-synonyms. Discussions in Section 3.3 and 3.4 also reveal that LSA is essentially an additive model weighted by the eigenvalues of the co-occurrence matrix. The prevalence of additive models in the existing literature calls for explorations of alternative composition models in near-synonymy study. Secondly, a commonality among many of LSTM's successful applications reported so far is that humans are able to excel with relative ease in these tasks (e.g., sentiment analysis (Zhu et al., 2015), text classification (Zhou et al., 2015), named entity recognition (Chiu and Nichols, 2015), etc.). In contrast, semantic distinctions in near-synonymy are highly fine-grained and difficult to differentiate even for native speakers of a language, which makes NSLC a unique challenge for the neural-network-based models. A successful application of the model in NSLC would be a meaningful addition to the model's success in NLP.

### 8.3.1 Task Description and Model Configuration

NSLC is formulated as a classification problem. The objective is to recover the word (a member of a group of near-synonyms) originally used at the position of a blank in a sentential context. On the lexical level, distributed representations (Section 2.2) are assigned to all words in a given context. Context words are used as sequential input to the model, with each word corresponding to a time step in the model. Model output is calculated using Equations (8.3) to (8.8) given both the input and the recurrent states from the previous time step. The sequence-final output (i.e., output emitted by LSTM at the end of the context sequence) is used for classification. The classifier consists of a single-layer neural network, and the class-based negative log-likelihood is used as the loss function for optimization. The class with the highest probability is proposed as the correct answer, which is compared against the gold label (i.e., the original word used at the position of the blank in the context). During training, the difference is used as the learning signal and back-propagated using stochastic gradient descent (SGD) to update the model parameters. For testing, the difference is used to calculate accuracy for evaluating model performance.

We use 300 dimensions for lexical embeddings, partly for the convenience of using published pre-trained embeddings such as `word2vec` (Mikolov et al., 2013). On the other hand, the dimensionality of LSTM hidden states is observed to have a positive influence on model performance, but the difference minimal and without statistical significance. We use 200 as the dimensionality for obtaining results presented below.

Weight parameters are initialized with normally distributed random values (mean 0.0, standard deviation 0.01). Regularization is performed by gradient-clipping with range $\pm 5.0$ and by applying dropout (0.5) between time steps. Mini-batch (100) is used for SGD, and momentum

(Sutskever et al., 2013) is implemented with $\epsilon = 0.33$. Learning rate starts from 0.75 and is linearly decreased to $7.5e-4$ at the end of training (100 epochs).

## 8.3.2 Evaluation

Experiments in this section are conducted using the *Wall Street Journal* (WSJ) data for fair comparison with existing NSLC systems. The 1989 WSJ is used for training and 1987 WSJ for testing. Due to the small size of the corpus, no frequency threshold is imposed. More importantly, we believe rare words can carry important information on the nuanced distinctions among the target near-synonyms. Note that this is in contrast to most of the existing studies where frequency thresholds are used to reduce the vocabulary space.

Stop words are removed for two reasons. Firstly, the omission of function words should not incur a significantly change of the overall meaning of a sentence. Secondly, if a model performs better with the function words removed than when they are present, it can be argued that the model relies more on the semantic information of the input rather than the syntactic aspects (e.g., grammatical, collocational, etc.). Empirically, no statistical significance is observed in the difference between the performance of the two variations.

#### Temporal Distribution

The first experiment is to study the *temporal distribution* in the model input and its effect on the composition performance. To define temporal distribution, consider the following lexical choice example from the WSJ dataset on the near-synonym set {*job*, *duty*, *task*}:

(7) Mr. Hamilton faces a daunting *BLANK* in reviving Ogilvy's
stagnant New York office.

Here, *BLANK* indicates where the original near-synonym was used in the sentence. Temporal distribution refer to the distribution of the syntactic functions of the context words given their positions relative to the blank. Note that temporal distributions are non-uniform. For example,

| | *difficult* | *error* | *job* | *responsibility* | *material* | *give* | *settle* | Avg. |
|---|---|---|---|---|---|---|---|---|
| `wf_c` | 68.8 | 74.8 | 86.2 | 72.4 | 75.2 | 81.7 | 77.7 | **76.7** |
| `wf_s` | 65.6 | 73.6 | 85.3 | 69.9 | 72.7 | 78.1 | 77.1 | 74.6 |

Table 8.2: Fixing the positions of the blanks achieves better accuracy (%) than random positions in the FITB task.

given the correct answer *task*, it is not surprising that the adjective *daunting* is the word at position $-1$ (i.e., the position immediately to the left of the blank), nor that the article *a* is the word at position $-2$, etc. The lack of surprise indicates the non-uniformity of the distributions and thus the *relative position* is informative of the syntactic function of the word at that position. On the other hand, however, when using the context as a sequential input, LSTM is oblivious to the relative position but instead only depends on the *in-sequence position* (e.g., 5 for *a*, 4 for *made*, etc.) at each time step in Equations (8.3) to (8.8). The hypothesis is that by aligning the relative position with the in-sequence position, at any given time step, the LSTM model is exposed to a set of temporal distributions with relatively lower entropy around the target near-synonym, which offers advantages to gaining statistical efficiency during training.

To test this hypothesis, two methods are compared for constructing window frames. The first one, dubbed `wf_c` uses the conventional definition of a window frame, where the blank is placed at the centre of a frame with exactly $n$ words on each side. The second method (`wf_s`) relaxes the centring constraint by randomly shifting the position of the blank off-centre while maintaining the frame size at $2n+1$. Clearly, only `wf_c` aligns the in-sequence position with the relative position, which, given the validity of the hypothesis, should lead to better model performance due to the relatively invariant distribution of the input at each time step. Empirical results confirm our hypothesis. As shown in Table 8.2, higher accuracy are achieved for all seven data sets when the blank position is centred.

### Window Size

The choice to represent contexts as window frames of $\pm n$ words around the blank is motivated by several reasons. Firstly, window frames yield contextual sequences of identical lengths

| | *difficult* | *error* | *job* | *responsibility* | *material* | *give* | *settle* | Avg. |
|---|---|---|---|---|---|---|---|---|
| `wf_1` | 68.0 | 69.0 | 86.0 | 69.9 | 70.0 | 75.8 | 68.1 | 72.4 |
| `wf_2` | 70.3 | 70.4 | 86.4 | 70.8 | 69.4 | 79.5 | 75.9 | 74.6 |
| `wf_3` | 68.8 | 74.8 | 86.2 | 72.4 | 75.2 | 81.7 | 77.7 | **76.7** |
| `wf_4` | 63.0 | 57.8 | 84.2 | 67.0 | 69.5 | 78.0 | 78.6 | 71.1 |
| `wf_5` | 61.4 | 50.7 | 81.8 | 62.7 | 70.1 | 76.6 | 75.2 | 68.3 |

Table 8.3: FITB performance comparison for different window frame sizes. The optimal window size is larger than usual for the proposed model, while smaller sizes are still favoured over larger ones.

$(2n+1)$, which facilitates the implementation of mini-batches in SGD for better efficiency and performance in optimization. More importantly, window frame size enables us to study the effect of co-occurrence locality. In the existing literature, NSLC models always favour highly local contexts (e.g., that of Inkpen, 2007). A relatively small window frame size (e.g., $\pm 2$) surrounding a target near-synonym usually expresses more collocational preferences than the semantics of the context (Inkpen and Hirst, 2002). Consequently, comparing the effect of window frame size with existing studies is helpful in understanding the type of information a model is learning from the contexts. Results in Table 8.3 shows that the proposed model performs best with window size 3 (which is used for the $wf_c$ model in the first row), indicating that the model actually favours a larger than usual window size. The observation also holds for the majority of the individual near-synonym sets. Nonetheless, increasing window size to 4 or 5 quickly worsens the model performance. Although not optimal, smaller window sizes of 1 or 2 still outperform larger window sizes.

### Lexical Embedding Pretraining

Note that the results above are obtained by initializing the lexical embeddings with pre-trained embeddings (Mikolov et al., 2013). Convergence is much slower with random initialization. The final accuracy without pre-trained embeddings is also much lower than the reported results (by close to 5 percentage points). A major difference between the two variants is that in the former, the lexical embedding layer of the network is essentially pre-trained using a much

| | *difficult* | *error* | *job* | *responsibility* | *material* | *give* | *settle* | Average | |
|---|---|---|---|---|---|---|---|---|---|
| Data size | 6630 | 1052 | 5506 | 3115 | 1715 | 11504 | 1594 | Macro | Micro |
| Inkpen (2007) | 57.3 | 70.8 | 86.7 | 66.7 | 71.0 | 56.1 | 75.8 | 69.2 | 65.2 |
| Wang and Hirst (2010) | 61.7 | 82.5 | 82.4 | 63.5 | 78.5 | 75.4 | 77.9 | 74.6 | 73.1 |
| `RNN-LSTM` | 68.8 | 74.8 | 86.2 | 72.4 | 75.2 | 81.7 | 77.7 | **76.7** | **78.0** |
| Most freq. baseline | 42.2 | 44.6 | 71.0 | 33.4 | 60.2 | 57.2 | 59.3 | 52.6 | 53.9 |

Table 8.4: Model comparison in the FITB task.

larger dataset. Another possible reason for the performance difference is that the WSJ dataset is simply too small for training reliable lexical embeddings.

Table 8.4 shows the comparison between the existing NSLC models, the most-frequent baseline, and the proposed model (`LSTM`). On the individual synonym sets, absolute accuracies significantly correlate with the most-frequent baselines ($\rho = 0.82$). Relative performance[3] exhibits some correlation with data size ($\rho = 0.61$), although the correlation is not statistically significant. The correlation indicates that the proposed model appears to be data-hungry and performs well when training data is abundant, which is consistent with observation made with other neural-network based models. The conjecture is better demonstrated by the fact that the micro-averaged accuracy is significantly higher than the macro average.

Compared to the previous state-of-the-art lexical choice model, the proposed model performs 2.1 percentage points higher in macro average ($p = 0.0027$) and 5.1 percentage points higher in micro average ($p \ll 0.001$).

[3]Calculated as the difference in accuracy between `LSTM` and the best accuracy in the existing models.

# Chapter 9

# Unsupervised Word Sense Disambiguation Using Compositional Similarity[1]

## 9.1 Introduction

As mentioned in Section 1.2.2, the objective of word sense disambiguation (WSD) is to model the functional mapping between a polysemous word form and its multiple senses. In practice, however, the occurrences of a polysemous word form are computationally indistinguishable, and thus the problem is circumvented by mapping the *contexts* associated with each occurrence of the word form to the appropriate sense in the word's sense inventory. Since contexts are usually multi-word units, the problem of WSD would often rely on composition models for deriving context representations.

To disambiguate homonyms in a given context, Lesk (1986) proposed a method that measured the degree of overlap between the glosses of the target homonym. Known as the Lesk algorithm, this simple and intuitive method has since been extensively cited and extended in the word sense disambiguation (WSD) community. Despite the intuitiveness of the Lesk algorithm (Lesk, 1986), however, its performance in several WSD benchmarks is less than satisfactory

[1]This study was first published as a conference paper in ACL 2014 (Wang and Hirst, 2014).

(Kilgarriff and Rosenzweig, 2000; Vasilescu et al., 2004). One of the popular explanations is related to a key limitation of the algorithm, that "Lesk's approach is very sensitive to the exact wording of definitions, so the absence of a certain word can radically change the results." (Navigli, 2009).

Compounding this problem is the fact that many Lesk variants limited the concept of overlap to the literal interpretation of string matching (with their own variants such as length-sensitive matching (Banerjee and Pedersen, 2002), etc.), and it was not until recently that overlap started to take on other forms such as tree-matching (Chen et al., 2009) and vector space models (Abdalgader and Skabar, 2012; Raviv et al., 2012; Patwardhan and Pedersen, 2006). To further address this limitation, a Naive Bayes model (NBM) is proposed in this study as a novel, probabilistic measurement of overlap in gloss-based WSD.

## 9.2 Related Work

In the extraordinarily rich literature on WSD, we focus our review on those closest to the topic of the Lesk algorithm and Naive Bayes models (NBMs) since these are most relevant to the model proposed in this study. Note that there are two major variants of the Lesk algorithm. The original algorithm measures the *gloss-gloss* overlap, i.e., by calculating the overlap between the gloss of the target polysemous word and the glosses of all its context words. A "simplified Lesk" (Kilgarriff and Rosenzweig, 2000) has also been developed, where inventory senses are assessed by *gloss-context* overlap that measures the overlap between the target's gloss and the actual context words. This variant prevents proliferation of gloss comparison especially on larger contexts (Mihalcea et al., 2004) and is shown (Vasilescu et al., 2004) to outperform the original Lesk algorithm. Nonetheless, the two algorithms are referred to interchangeably in the reviews below.

To the best of our knowledge, NBMs have been employed exclusively as classifiers in WSD — that is, in contrast to their use as a similarity measure in this study. Gale et al. (1992)

used an NB classifier resembling an information retrieval system, where a WSD instance is regarded as a document $d$, and candidate senses are scored in terms of "relevance" to $d$. When evaluated on a WSD benchmark (Vasilescu et al., 2004), the algorithm compared favourably to Lesk variants (as expected for a supervised method). Pedersen (2000) proposed an ensemble model with multiple NB classifiers differing by context window size. Hristea (2009) trained an unsupervised NB classifier using the EM algorithm and empirically demonstrated the benefits of WordNet-assisted (Fellbaum, 1998) feature selection over local syntactic features.

Among the many Lesk variants, Banerjee and Pedersen (2002) extended the gloss of both inventory senses and the context words to include words in their related synsets in WordNet. Senses were scored by the sum of overlaps across all relation pairs, and the effect of individual relation pairs was evaluated in a later work (Banerjee and Pedersen, 2003). Overlap was assessed by string matching, with the number of matching words squared so as to assign higher scores to multi-word overlaps.

Instead of string matching, Wilks et al. (1990) measured overlap as similarity between gloss- and context-vectors, which were aggregated word vectors encoding second order co-occurrence information in glosses. An extension by Patwardhan and Pedersen (2006) differentiated context word senses and extended shorter glosses with related glosses in WordNet. Patwardhan et al. (2003) measured overlap by *concept similarity* (Budanitsky and Hirst, 2006) between each inventory sense and the context words. Contrary to intuition, however, simple gloss overlap proposed in their earlier work actually out-performed all five similarity-based methods.

More recently, Chen et al. (2009) proposed a tree-matching algorithm that measured gloss-context overlap as the weighted sum of dependency-induced lexical distance. Abdalgader and Skabar (2012) constructed a *sentential* similarity measure (Li et al., 2006) consisting of *lexical* similarity measures (Budanitsky and Hirst, 2006), and overlap was measured by the cosine of their respective sentential vectors. A related approach (Raviv et al., 2012) also used Wikipedia-induced concepts to encoded sentential vectors. These systems compared favourably to existing methods in WSD performance, although by using sense frequency information, they are essentially supervised methods.

Distributional methods have been used in many WSD systems in quite different flavours than the current study. Kilgarriff and Rosenzweig (2000) proposed a Lesk variant where each gloss word is weighted by its *idf* score in relation to all glosses, and gloss-context association was incremented by these weights rather than binary, overlap counts. Miller et al. (2012) used distributional thesauri as a knowledge base to increase overlaps, which were, again, assessed by string matching.

In conclusion, the majority of Lesk variants focused on extending the gloss to increase the chance of overlapping, while the proposed NBM is designed to make better use of the limited lexical knowledge by delegating the measure of similarity to the vast amount of text corpora. The main hypothesis here is that distributional methods can offer a better measurement of the gloss-context overlap. Intuitively, the probabilistic nature of the proposed model offers a "softer" measurement of gloss-context association compared to string matching. Empirically, the resulting model achieves state-of-the-art performance when evaluated on two benchmarks of unsupervised WSD (Section 9.4).

## 9.3 Model and Task Descriptions

### 9.3.1 The Naive Bayes Model

Formally, given two sets $\mathbf{e} = \{e_i\}$ and $\mathbf{f} = \{f_j\}$ each consisting of multiple samples of some random event, the proposed model measures the probabilistic association $p(\mathbf{f}|\mathbf{e})$ between $\mathbf{e}$ and $\mathbf{f}$. Under the assumption of conditional independence among the events in each set, a Naive Bayes measurement can be formulated as:

$$p(\mathbf{f}|\mathbf{e}) = \prod_j p(f_j|\{e_i\}) = \prod_j \frac{p(\{e_i\}|f_j)p(f_j)}{p(\{e_i\})} = \frac{\prod_j [p(f_j)\prod_i p(e_i|f_j)]}{\prod_j \prod_i p(e_i)}. \tag{9.1}$$

When applied to language, the frequency of the events can be quite large, and hence logarithm is applied to improve numerical stability:

$$\log p(\mathbf{f}|\mathbf{e}) = \sum_i \log p(e_i) + \sum_i \sum_j \log p(f_j|e_i) - |\mathbf{e}| \sum_j \log p(f_j),$$

In addition, the above quantity can be estimated with maximum likelihood using marginal and joint frequencies of the samples:

$$\begin{aligned}
\log p(\mathbf{f}|\mathbf{e}) \approx & \sum_i \log \frac{c(e_i)}{c(\cdot)} + \sum_i \sum_j \log \frac{c(f_j, e_i)}{c(e_i)} - |\{e_i\}| \sum_j \log \frac{c(f_j)}{c(\cdot)} \\
= & \sum_i \log c(e_i) - \sum_i \log c(\cdot) + \sum_i \sum_j \log c(f_j, e_i) \\
& - \sum_i \sum_j \log c(e_i) - |\{e_i\}| \sum_j \log c(f_j) + |\{e_i\}| \sum_j \log c(\cdot) \qquad (9.2) \\
= & (1 - |\{f_j\}|) \sum_i \log c(e_i) - |\{e_i\}| \sum_j \log c(f_j) \\
& + \sum_i \sum_j \log c(f_j, e_i) + |\{e_i\}|(|\{f_j\}| - 1) \log c(\cdot).
\end{aligned}$$

To regularize for the sizes of **e** and **f**, one can also adjust the probability by the following normalization terms to ensure that the resulting probability $\tilde{p}(\mathbf{e}|\mathbf{f})$ is a proper distribution:

$$\tilde{p}(\mathbf{e}|\mathbf{f}) = \frac{p(\mathbf{e}|\mathbf{f})/(|\mathbf{e}| + |\mathbf{f}|)}{\sum_{(\hat{\mathbf{e}}, \hat{\mathbf{f}}) \in \mathscr{E} \times \mathscr{F}} \frac{1}{|\hat{\mathbf{e}}| + |\hat{\mathbf{f}}|}},$$

For a given data set and a lexical knowledge base, the denominator is a constant and thus can be omitted.

In step two in Equation (9.1), Bayes's rule is applied not only to take advantage of the conditional independence among $e_i$'s, but also to facilitate probability estimation, since $p(\{e_i\}|f_j)$ is easier to estimate in the context of WSD, where sample spaces of **e** and **f** become asymmetric (details to follow in Section 9.3.2).

### 9.3.2 Model Application in WSD

In the context of WSD, $\mathbf{e}$ can be regarded as an observed context of a polysemous word $w$, while $\mathbf{f}$ represents certain lexical knowledge about the sense $s$ of $w$ manifested in $\mathbf{e}$.[2] Note that in this formulation, $\mathbf{f}$ is in effect a function of $s$. WSD can then be formulated as identifying the sense $s^*$ in the sense inventory $\mathscr{S}$ of $w$ s.t.:

$$s^* = \operatorname*{arg\,max}_{s \in \mathscr{S}} p(\mathbf{f}(s)|\mathbf{e}) \tag{9.3}$$

In one of their simplest forms, $e_i$'s correspond to co-occurring words in the instance of $w$, and $f_j$'s consist of the gloss words of sense $s$. Consequently, $p(\mathbf{f}(s)|\mathbf{e})$ is essentially measuring the association between the context words of $w$ and the definition texts of $s$, i.e., the gloss-context association in the simplified Lesk algorithm (Kilgarriff and Rosenzweig, 2000). A major difference, however, is that instead of using hard, overlap counts between the two sets of words from the gloss and the context, the proposed probabilistic measure of association can implicitly model the distributional similarity among the elements $e_i$ and $f_j$ (and consequently between the sets $\mathbf{e}$ and $\mathbf{f}(s)$) over a wider range of contexts. The result is a "softer" proxy for association than the binary view of overlaps in existing Lesk variants.

The discussions above offer a second motivation for applying Bayes's rule on the second expression in Equation (9.1): it is easier to estimate $p(e_i|f_j)$ than $p(f_j|e_i)$, since the vocabulary for the lexical knowledge features ($f_j$) is usually smaller compared to that of the contexts ($e_i$). As a result, estimation of the former suffices on a smaller amount of data than that for the latter.

### 9.3.3 Incorporating Additional Lexical Knowledge

The input of the proposed NBM is bags of words, and thus it is straightforward to incorporate different forms of lexical knowledge (LK) for word senses, for example, by concatenating a tokenized knowledge source to the existing knowledge representation $\mathbf{f}$, while the similarity

[2] Think of the notations $\mathbf{e}$ and $\mathbf{f}$ mnemonically as *exemplars* and *features*, respectively.

| Senses | Hypernyms | Hyponyms | Synonyms |
|---|---|---|---|
| *factory* | building complex, complex | brewery, factory, mill, ... | works, industrial plant |
| *life form* | organism, being | perennial, crop... | flora, plant life |

Table 9.1: Lexical knowledge for the word *plant* under its two meanings *factory* and *life form*.

measure remains unchanged.

The availability of LK largely depends on the sense inventory used in a WSD task. WordNet senses are often used in Senseval and SemEval tasks, and hence senses (or synsets, and possibly their corresponding word forms) that are semantically related to the inventory senses under WordNet relations are easily obtainable and have been used by many existing studies.

As pointed out by Patwardhan et al. (2003), however, "not all of these relations are equally helpful." Relation pairs involving hyponyms were shown to result in better F-measure when used in gloss overlaps (Banerjee and Pedersen, 2003). The authors attributed the phenomenon to the multitude of hyponyms compared to other relations. We further hypothesize that, in addition to larger quantity, synonyms and hyponyms offer stronger semantic specification that helps distinguish the senses of a given ambiguous word, and thus are more effective knowledge sources for WSD.

Take the word *plant* for example. Selected hypernyms, hyponyms, and synonyms pertaining to its two senses *factory* and *life form* are listed in Table 9.1. Hypernyms can be overly general terms (e.g., *being*). Although conceptually helpful for humans in coarse-grained WSD, generality entails high frequency, which is likely to cause inaccuracy in the hypernyms' probability estimation. Hyponyms, on the other hand, help specify their corresponding senses with information that is possibly missing from the often overly brief glosses. For example, many of the technical terms as hyponyms in Table 9.1 — though rare — are likely to occur in the (possibly domain-specific) contexts that are highly typical of the corresponding senses. Particularly for the NBM, the co-occurrence is likely to result in stronger gloss-definition associations when similar contexts appear in a WSD instance.

We also observe that some semantically related words appear under rare senses (e.g., *still* as an alcohol-manufacturing plant, and *annual* as a one-year-life-cycle plant, both omitted from Table 9.1). This is a general phenomenon in gloss-based WSD and is beyond the scope of the current discussion.[3] Overall, all three sources of LK may complement each other in WSD tasks, with hyponyms particularly promising in both quantity and quality compared to hypernyms and synonyms.[4]

### 9.3.4 Probability Estimation

A most open-ended question is how to estimate the probabilities in Equation (9.1). In WSD in particular, the estimation involves the marginal and conditional probabilities of and between word tokens. Many options are available to this end in machine learning (MLE, MAP, etc.), information theory (Church and Hanks, 1990; Turney, 2001), as well as the rich body of research in lexical semantic similarity (Resnik, 1995; Jiang and Conrath, 1997; Budanitsky and Hirst, 2006). In this study, we choose maximum likelihood — not only for its simplicity, but also to demonstrate model strength with a relatively crude probability estimation. To avoid underflow, log probability is estimated according to Equation (9.2).

In addition, we investigate how model performance responds to estimation quality by varying the training data size. Specifically in WSD, a *source corpus* is defined as the source of the majority of the WSD instances in a given dataset, and a *baseline corpus* of a smaller size and less resemblance to the instances is used for all datasets. Clearly, a source corpus offers better estimates for the model than the baseline corpus, and the assumption is that using probability estimation of different quality should result in differences in model performance.

[3]We do, however, refer curious readers to the work of Raviv et al. (2012) for a novel treatment of a similar problem.

[4]Note that LK expansion is a feature of our model rather than a requirement. What type of knowledge to include is eventually a decision made by the user based on the application and LK availability.

## 9.4 Evaluation

### 9.4.1 Data, Scoring, and Pre-processing

Various aspects of the model discussed in Section 9.3 are evaluated in the English lexical sample tasks from Senseval-2 (Edmonds and Cotton, 2001) and SemEval-2007 (Pradhan et al., 2007). Training sections are used as development data and test sections are used for final testing. Model performance is evaluated in terms of WSD accuracy using Equation (9.3) as the scoring function. Accuracy is defined as the number of correct responses over the number of instances. Since the output of the NBM are real numbers, it is highly unlikely to have ties among the candidates,[5]. As a result, the model is designed to always proposes a unique answer, and accuracy is thus equivalent to F-score commonly used in existing reports.

Multiword expressions (MWEs) in the Senseval-2 sense inventory are not explicitly marked in the contexts. Several of the top-ranking systems implemented their own MWE detection algorithms (Kilgarriff and Rosenzweig, 2000; Litkowski, 2002). Without digressing to the details of MWE detection — and meanwhile, to ensure fair comparison with existing systems — we implement two variants of the prediction module, one completely ignorant of MWE and defaulting to `INCORRECT` for all MWE-related answers, while the other assuming perfect MWE detection and performing regular disambiguation algorithm on the MWE-related senses (n.b., not defaulting to `CORRECT`). All results reported for Senseval-2 below are harmonic means of the two outcomes.

Each inventory sense is represented by a set of LK tokens (e.g., definition texts, synonyms, etc.) from their corresponding WordNet synset (or in the coarse-grained case, a concatenation of tokens from all synsets in a sense group). The *MIT-JWI* library (Finlayson, 2014) is used for accessing WordNet. Usage examples in glosses (included by the library by default) are removed in our experiments.[6]

[5]This has never occurred in the hundreds of thousands of runs in our development process.

[6]We also compared the two Lesk baselines (with and without usage examples) on the development data but did not observe significant differences, which is consistent with the observations reported by Kilgarriff and Rosenzweig (2000).

| Dataset | *glo* | *syn* | *hpr* | *hpo* | *all* | 1st | 2nd | *Lesk* |
|---|---|---|---|---|---|---|---|---|
| *Senseval-2 Coarse* | .475 | .478 | .494 | .518 | **.523** | .469 | .367 | .262 |
| *Senseval-2 Fine* | .362 | .371 | .326 | .379 | .388 | **.412** | .293 | .163 |
| *SemEval-2007* | .494 | .511 | .507 | .550 | **.573** | .538 | .521 | – |

Table 9.2: Lexical knowledge sources and WSD performance (*F-measure*) on the Senseval-2 (fine- and coarse-grained) and the SemEval-2007 dataset.

Basic pre-processing is performed on the contexts and the glosses, including lower-casing, stopword removal, lemmatization on both datasets, and tokenization on the Senseval-2 instances.[7] *Stanford CoreNLP*[8] is used for lemmatization and tokenization. Identical procedures are applied to both the source and the baseline corpus used for probability estimation.

The binomial test is used for testing statistical significance, all improvements presented are statistically highly significant ($p < 0.001$) with one exception explicitly noted in Section 9.4.3 as statistically significant ($p < 0.01$).

### 9.4.2 Comparing Lexical Knowledge Sources

To study the effect of different types of LK in WSD (Section 9.3.3), for each inventory sense, we choose synonyms (*syn*), hypernyms (*hpr*), and hyponyms (*hpo*) as extended LK in addition to its gloss. The WSD model is evaluated with gloss-only (*glo*), individual extended LK sources, and the combination of all four sources (*all*). The results are listed in Table 9.2 together with existing results (1st and 2nd correspond to the results of the top two unsupervised methods in each dataset).[9]

By using only glosses, the proposed model already shows drastic improvements over the basic Lesk algorithm (92.4% and 140.5% relative improvement in Senseval-2 coarse- and fine-grained tracks, respectively).[10] Moreover, comparison between coarse- and fine-grained tracks

[7]The SemEval-2007 instances are already tokenized.

[8]`http://nlp.stanford.edu/software/corenlp.shtml`.

[9]We excluded the results of *UNED* (Fernández-Amorós et al., 2001) in Senseval-2 because, by using sense frequency information that is only obtainable from sense-annotated corpora, it is essentially a supervised system.

[10]Comparisons are made against the simplified Lesk algorithm (Kilgarriff and Rosenzweig, 2000) without usage examples. The comparison is unavailable in SemEval2007 since we have not found existing experiments with this exact configuration.

reveals interesting properties of different LK sources. Previous hypotheses (Section 9.3.3) are empirically confirmed that WSD performance benefits most from hyponyms and least from hypernyms. Specifically, highly similar, fine-grained sense candidates apparently share more hypernyms in the fine-grained case than in the coarse-grained case; adding to the generality of hypernyms, we postulate that their probability in the NBM is uniformly inflated among many sense candidates, and hence the decrease in distinguishability. Synonyms might help with regard to semantic specification, though their limited quantity also limits their benefits. These patterns on the LK types are consistently observed in all three tracks of the benchmarks.

When including all four LK sources, our model outperforms the state-of-the-art systems with statistical significance in both coarse-grained tasks. For the fine-grained track, it ranks second after that of Tugwell and Kilgarriff (2001), which used a decision list (Yarowsky, 1995) on manually selected features for each inventory sense, and thus is not subject to loss of distinguishability in the glosses as with the fully automated Lesk variants.

### 9.4.3 The Effect of Probability Estimation

In this section we evaluate how different probability estimations derived from different training corpora can affect performance of the proposed model. For each dataset, we use two corpora to perform probability estimation. The first corpus (referred to as the *source corpus*) is one from which the majority of the instances in a dataset are collected. The source corpus can be identified by the distribution of the *doc-source* attribute associated with each instance in a dataset. For example, 94% of the Senseval-2 instances come from the *British National Corpus* (*BNC*), and 86% of SemEval-2007 instances come from the *Wall Street Journal*. For comparison, we use a third corpus (the *Brown Corpus*) as the *baseline corpus*. Clearly, for a given WSD dataset, the source corpus is more representative of the lexical distributions of the contexts, and hence, probability estimation done on the source corpus is presumably more accurate than on the baseline corpus.

Figure 9.1 shows the comparison on the SemEval-2007 dataset. Across all experiments,

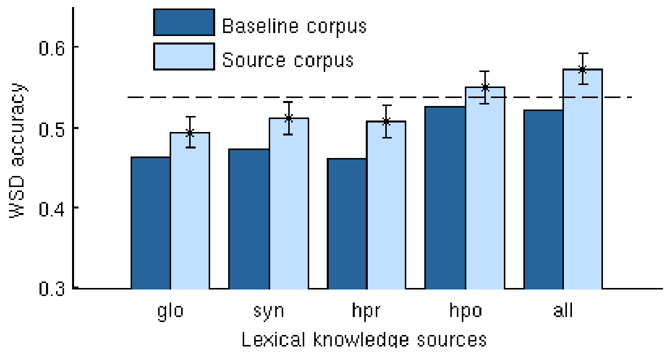


Figure 9.1: Model response to probability estimates of different quality on the SemEval-2007 dataset. Error bars indicate confidence intervals ($p < .001$), and the dashed line corresponds to the best reported result.

higher WSD accuracy is consistently witnessed using the source corpus; differences are statistically highly significant except for *hpo* (which is significant with $p < 0.01$).

# Chapter 10

# Conclusions for Part II

In this part of the thesis we focused on developing and evaluating distributional composition models that compose the lexical representations from Part I into representations of longer textual sequences such as phrases and sentences. We started with the classic additive model (Mitchell and Lapata, 2010) and showed that in addition to their appeal with regard to intuition and simplicity, additive models are in fact the empirical result of distributed representation learning for phrases.

Nevertheless, to address the limitations of additive models with regard to ignoring word-order and other linguistic information, we further investigated the effectiveness of some state-of-the-art composition models using recurrent neural networks (RNNs) and hypothesized that performance can be improved by modelling syntactic knowledge of the input. More specifically, we investigated a variant of RNN with long short-term memory that is proven by existing studies to be effective in learning representations of text sequences. Theoretically, the memory cell and the gating mechanism in LSTM are able to "trap" gradients within sub-sequences of sequential input (Gers et al., 2000). By analysing its full gradients, we hypothesized that the gating behaviour can be further improved by "teacher-forcing", i.e., explicitly marking the input with syntactic knowledge and thus providing the model with temporal cues to guide its gate activations. Empirical experiments confirmed our hypothesis with improved performance in both language modelling and near-synonym lexical choice.

In addition, we developed a compositional similarity measure that assesses the association between two multi-word units which, in contrast to traditional composition models, does not explicitly rely on vector-space representations. When applied to polysemy, the model was employed to replace string matching in the Lesk algorithm for word sense disambiguation with a probabilistic measure of gloss-context overlap. The base model on average more than doubled the accuracy of Lesk in Senseval-2 on both fine- and coarse-grained tracks. With additional lexical knowledge, the model also outperformed state-of-the-art results with statistical significance on two coarse-grained WSD tasks.

# Part III

# Summary

# Chapter 11

# Summary

## 11.1 Conclusions

Building effective and efficient representations of linguistic units (words, phrases, sentences, etc.) is a fundamental problem in computational linguistics. In this thesis, several aspects of this problem were explored in the context of near-synonymy, semantic similarity, and polysemy. The main objective was to improve computational and statistical efficiency of both lexical and compositional distributional models. I hypothesized that syntactic and lexicographic knowledge can help reduce both computational and statistical complexity in the data to achieve this goal.

Major contributions of the thesis are as follows. On the lexical level, due to the Zipfian nature of lexical distribution in language, sparsity is one of the most prominent issues in lexical semantics studies (Chapter 2). To address this issue, I proposed two dimension reduction models using matrix factorization techniques and applied them to building reliable lexical representations. Firstly, I used lexical level co-occurrence as the basis for dimension reduction with singular value decomposition (Chapter 3). The resulting lexical representations were used as features in kernel methods. Using the correlation between co-occurrence significance and the order of the eigenvalues, the concept of subtlety was quantified with the dimensionality of the resulting latent space, which was in turn used as empirical evidence of the non-linearity of near-synonym nuances. The resulting model, despite its simplicity, outperforms all previous near-synonym lexical choice models.

Secondly, motivated by the high false positive and false negative rates of semantic association in window-based definition of co-occurrence, I incorporated syntactic and lexicographic knowledge into the definition and applied it to a contrastive-divergence-based lexical embedding model (Chapter 4). Using the information about a target word's dependency was proven not only effective in semantic modelling, but also efficient in computation since the dependent space is much more compact than the lexical space of its positional neighbours. Even more compact is the defining relation established using a dictionary. I proposed a model that exploits the hand-crafted lexicographic knowledge in dictionary definitions. Model performance in lexical similarity evaluations rivals the window-based methods trained on traditional text corpora that are thousands of times larger in size.

On the other hand, the compact nature of the dependency-based co-occurrence also brings limitations to the model because the co-occurrence information is sparser than in window-based models. To address this problem, a novel co-training mechanism was proposed that improves embedding quality by making use of the information about the dependency contexts in the output embedding matrix, which is usually discarded in existing embedding models. The proposal was empirically shown to be effective especially when the size of the training data is small.

In the context of measuring lexical semantic similarity, many existing studies make use of the structures in semantic taxonomies. However, certain widely adopted structural features such as depth and density have peculiar distributions over the concepts in hand-crafted taxonomic knowledge graphs such as WordNet. I provided empirical evidence to demonstrate that, contrary to the intuitions that have motivated the use of these structural features in similarity metrics, depth and density in fact rarely correlate with human judgements of lexical similarity. This issue was subsequently addressed by refining the definition of depth and density in order to more accurately model their distribution over lexical concepts (corresponding to nodes in a semantic taxonomy). Specifically, the ordinal representation of depth was replaced by a cumulative definition, and information on semantic specificity was made inheritable in local sub-graphs in WordNet. The resulting definitions not only resulted in much improved alignment between the feature values and human judgements of similarity, but also improved evaluation results in lexical similarity with statistical significance by incorporating the new definitions into existing similarity measures that use the metrics of depth and density.

Meanwhile, many NLP applications require representation of longer sequences of text. Simple composition models such as the additive model are often quite effective and can yield highly competitive results. However, these models ignore even the most basic linguistic information such as word-order and thus bear little resemblance to human behaviour in natural language understanding. Consequently, it is of both theoretical and practical merit to explore more complex alternatives that are capable of modelling linguistic knowledge in the data. To this end, I first employed a recurrent neural network model with long short-term memory units for semantic composition. Analytically, I reasoned that the gating behaviour of the LSTM units can potentially benefit from incorporating syntactic knowledge of the input sequence. Empirically, when the proposed model was applied to the tasks of language modelling and near-synonym lexical choice, the additional information on syntax was shown to help achieve notably better performance in both tasks.

Lastly, a compositional similarity measure was proposed to assess the similarity between two sets of random events. The model is a general framework under which lexical similarity can be estimated with either vector space similarity using distributed representations or simple co-occurrence statistics of the individual events. Empirically, the similarity metrics work very well even with very crude probability estimations such as the Maximum Likelihood Estimation. With the help of lexicographic knowledge, including a monolingual dictionary and a lexical taxonomy, the proposed model was successfully applied to unsupervised word sense disambiguation (WSD) and achieved state-of-the-art results in two widely-adopted WSD benchmarks.

## 11.2 Future Work

### 11.2.1 Near-synonymy

#### Decoupling Polysemy from Near-synonymy

The key motivation for placing the lexical and compositional semantic models in the context of near-synonymy is the high degree of granularity of the near-synonym distinctions. In the existing literature, however, lexical choice data has been extracted from text corpora with simple string matching without much consideration of the semantic properties of the target words. For example, the target near-synonyms can often be polysemous. A few existing studies briefly mentioned the effect of polysemy on near-synonym lexical choice and fill-in-the-blank data collection. Edmonds (1997) claimed that the seven near-synonym sets used in his study (and repeatedly in later studies) were chosen because of their low degree of polysemy. Inkpen (2007) stated that the presence of polysemy might result in a higher baseline in lexical choice data. In either case, the claims are largely based on intuition without empirical support.

To address this issue, a starting point would be to perform word sense disambiguation on the lexical choice data. An immediate challenge is that despite the large amount of research effort spent on WSD, performance of available WSD systems is quite limited (Navigli, 2009). Adding to the challenge is the fact that many of the near-synonyms do not bear their dominant meanings in the synonym groups they belong to (e.g., *history* in its synonym set with *annals* and *chronicle*, Hayakawa, 1994). In other words, it is highly likely that the lexical choice data that is currently used is rather noisy and inaccurate in representing the semantic distinctions among near-synonyms. A systematic investigation is much needed to resolve this issue.

### Exploiting Lexicographic Resources on Near-synonymy

Another promising direction is to devise novel ways of using lexicographic resources on near-synonymy and near-synonym distinctions, such as synonym dictionaries. Note that a synonym dictionary is different from a thesaurus, which focuses on the sameness of meaning and groups synonyms together to offer diversity in lexical choice. On the other hand, a synonym dictionary consists of detailed, hand-crafted lexicographic knowledge that aims at articulating the fine-grained differences among near-synonyms in the form of natural language. Synonym dictionaries are useful for humans to understand near-synonym distinctions. However, they are not extensively explored as a knowledge source for computational models in the existing literature. The few existing examples (e.g., that of Inkpen and Hirst 2006; Edmonds and Hirst 2002) rely heavily on manually created hierarchies and rules for assigning lexicographic properties to near-synonyms and their contexts.

As seen in Chapter 4 and Chapter 9, the general idea in using lexicographic knowledge is to establish an association between the distributional contexts of a target word and its lexicographic properties such as definition texts. In near-synonymy, however, several key assumptions are no longer valid in this framework. For example, near-synonym lexical choice focuses more on the contrastive meaning between a group of near-synonyms rather than the lexicographic meaning of any individual member in the group, for which definition texts in monolingual dictionaries used thus far in the thesis can be inadequate. Consider the dictionary definitions the word *error* and *mistake* from the *Merriam-Webster Dictionary*:

(8) ***error***: *an act or condition of ignorant or imprudent deviation from a code of behavior.*

(9) ***mistake***: *a wrong judgment.*

The definition texts are highly abstract and it is difficult even for a human reader to readily interpret the semantic difference between the two near-synonyms from them. In contrast, semantic differences are much more explicitly expressed in *synonym dictionaries* such as that of

Hayakawa (1994):

> (10) ***error, mistake:*** *... error often implies deviation from a standard or model, whereas mistake is preferred in the common situations of everyday life. ... Error is also used in a theological sense to mean sin, since sin is perceived as deviation from the moral standards or theological truths established by religion.*

In the above example, one (human reader, and hopefully also a computational model) can easily identify the differences between *error* and *mistake* with regard to specificity and topical applicability. When matching against distributional contexts of the respective near-synonyms, these description are computationally much more informative than the gloss texts found in monolingual dictionaries.

Many interesting distributional methods can be employed to achieve this goal. For example, a sequence-to-sequence model (Sutskever et al., 2014) can be used to generate distributional contexts from the encoding of the synonym dictionary glosses conditioned on a particular member of a near-synonym group. Since these models have previously been successfully applied to machine translation, the intuition here resembles the process of "translating" the abstract, lexicographic descriptions of near-synonym differences into concrete contexts of the corresponding near-synonyms. The generative nature of the decoder end in this proposal makes it easy to apply the model to solve NSLC problems. In addition, if the encoder and the decoder are reversed (e.g., generating descriptions for near-synonym differences given their distributional contexts), a model can be potentially trained to generate natural-language texts describing near-synonym differences from text corpora. Alternatively, an embedding model (e.g., similar to that of Hill et al., 2015a) can also be employed to embed the synonym dictionary glosses (and by extension, near-synonym differences) to predict the pair of near-synonyms being compared. The encoded lexicographic knowledge can then be applied in any near-synonymy-related application such as the lexical choice problem.

### 11.2.2 Lexical Representation Learning

The study introduced in Chapter 4 laid out a general framework for defining target-context association, and many extensions can be applied to the implementation adopted in this thesis. For example, when combining the heterogeneous knowledge sources in the relation-independent model (Section 4.1.2), the current implementation takes the top-$n$ relations and combines them with equal weights. If these weights are parameterized, it is possible not only to gain further improvement in performance, but also to quantify the contribution of the individual relations with regard to targets of different syntactic functions.

In addition, more evaluation tasks are needed to fully assess the strengths and weaknesses of the resulting embeddings. Currently, four lexical datasets and one phrasal similarity dataset are used in the evaluation. One of the criticisms of these datasets is that meanings of the target words are assessed out of context. Similar to the near-synonym lexical choice data, this can be a prominent issue especially when a target word has a high degree of polysemy. For similarity evaluation, there exist datasets that address this issue by providing a sentential context for each target word (Huang et al., 2012). Beyond lexical similarity, many other tasks can also be used for evaluation simply by using the embeddings as lexical representations (Pennington et al., 2014; Yogatama et al., 2014).

### 11.2.3 Semantic Composition

With the proposed LSTM-RNN model, the motivation for including linguistic knowledge is to improve gating behaviour and help the model identify syntactically plausible sub-sequences. In addition to syntactic information, there are many other types of linguistic knowledge that can be applied by "teacher-forcing" the model to achieve the same objective, including semantic clustering and labelling of the context words, and constituent markers from chunking. More recently, Ghosh et al. (2016) demonstrated that LSTM performance can be improved by topical information from the global-level contexts. In addition, with access to this linguistic knowledge, the model might be simplified by introducing dependency among the gates, much like the *gated recurrent unit* (GRU) proposed by Chung et al. (2015).

Recent studies also suggest that it is possible to make explicit the memory recurrence in RNN models. Cheng et al. (2016), for example, replaced the recurrent state at each time-step with a weighted sum of the entire history of the states and showed that the modification improved model performance in both sentiment classification and natural language inference tasks. However, computational efficiency is significantly compromised due to the weight calculations, which can be a major limitation of the model especially for data with larger sequence lengths. Nonetheless, many possible improvements can be made to address the issue including simplification of the weight calculation, for example, by making linguistically motivated assumptions of independence between model parameters and the state activations of the model, which may yield significant reduction in computational complexity.

# Bibliography

Khaled Abdalgader and Andrew Skabar. Unsupervised similarity-based word sense disambiguation using context vectors and sentential word importance. *ACM Transactions on Speech and Language Processing*, 9(1):2:1–2:21, May 2012.

Eneko Agirre and German Rigau. Word sense disambiguation using conceptual density. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 16–22, 1996.

Hiyan Alshawi. Processing dictionary definitions with phrasal pattern hierarchies. *Computational Linguistics*, 13(3-4):195–202, 1987.

Collin Baker, Charles Fillmore, and John Lowe. The Berkeley FrameNet project. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 86–90, 1998.

Satanjeev Banerjee and Ted Pedersen. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Computational Linguistics and Intelligent Text Processing*, pages 136–145. Springer, 2002.

Satanjeev Banerjee and Ted Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 805–810, 2003.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014.

Marco Baroni and Roberto Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193, 2010.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2003.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010.

John Rupert Lyon-Bowes Bernard. *The Macquarie Thesaurus: The book of words*. Macquarie Library, 1986.

William Blacoe and Mirella Lapata. A comparison of vector-based representations for semantic composition. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556. Association for Computational Linguistics, 2012.

Danushka Bollegala, Takanori Maehara, Yuichi Yoshida, and Ken-ichi Kawarabayashi. Learning word representations from relational graphs. In *Proceedings of the 29th AAAI Conference on Aritificial Intelligence*, pages 2146–2152, 2015.

Julian Brooke, Tong Wang, and Graeme Hirst. Automatic acquisition of lexical formality. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 90–98. Association for Computational Linguistics, 2010.

Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

Ping Chen, Wei Ding, Chris Bowes, and David Brown. A fully unsupervised word sense disambiguation method using dependency knowledge. In *Proceedings of the The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 28–36, 2009.

Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *Computing Research Repository*, abs/1601.06733, 2016. URL `http://arxiv.org/abs/1601.06733`.

Jason Chiu and Eric Nichols. Named entity recognition with bidirectional LSTM-CNNs. *arXiv preprint arXiv:1511.08308*, 2015.

Martin Chodorow, Roy Byrd, and George Heidorn. Extracting semantic hierarchies from a large on-line dictionary. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pages 299–304, 1985.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Gated feedback recurrent neural networks. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2067–2075, 2015.

Kenneth Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143, 1988.

Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

Trevor Cohen, Roger Schvaneveldt, and Dominic Widdows. Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections. *Journal of biomedical informatics*, 43(2):240–256, 2010.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM, 2008.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

D. A. Cruse. *Lexical Semantics*. Cambridge University Press, 1986.

Ido Dagan, Lillian Lee, and Fernando Pereira. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 34(1):43–69, 1999.

Scott Deerwester, Susan Dumais, George Furnas, Thomas Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

Chrysanne DiMarco, Graeme Hirst, and Manfred Stede. The semantic and stylistic differentiation of synonyms and near-synonyms. *AAAI Spring Symposium on Building Lexicons for Machine Translation*, pages 114–121, 1993.

Philip Edmonds. Choosing the word most typical in context using a lexical co-occurrence network. In *Proceedings of the 35th annual meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 507–509, 1997.

Philip Edmonds and Scott Cotton. Senseval-2: Overview. In *Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, 2001.

Philip Edmonds and Graeme Hirst. Near-synonymy and lexical choice. *Computational Linguistics*, 28(2):105–144, 2002.

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*, pages 897–906. Association for Computational Linguistics, 2008.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah Smith. Retrofitting word vectors to semantic lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, 2015.

Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.

David Fernández-Amorós, Julio Gonzalo, and Felisa Verdejo. The UNED systems at Senseval-2. In *The Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 75–78, 2001.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: the concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, pages 406–414, 2001.

Mark Alan Finlayson. Java libraries for accessing the Princeton WordNet: Comparison and evaluation. In *Proceedings of the 7th Global Wordnet Conference*, Tartu, Estonia, 2014.

Evgeniy Gabrilovich and Shaul Markovitch. Wikipedia-based semantic interpretation for natural language processing. *Journal of Artificial Intelligence Research*, pages 443–498, 2009.

William Gale, Kenneth Church, and David Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26(5-6):415–439, 1992.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB: The paraphrase database. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, 2013.

Helena Hong Gao. A specification system for measuring relationship among near-synonyms of physical action verbs. In *Second Workshop on Chinese Lexical Semantics*, 2001.

Felix Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 2000.

Samuel Gershman and Josh Tenenbaum. Phrase similarity in humans and machines. In *Proceedings of the 37th Annual Cognitive Science Society Meeting*, 2015.

Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry Heck. Contextual LSTM (CLSTM) models for large scale NLP tasks. *CoRR*, abs/1602.06291, 2016.

Gene Golub and Charles Van Loan. *Matrix Computations*. Johns Hopkins Univeristy Press, 2012.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.

Emiliano Guevara. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 33–37, 2010.

Zellig Harris. Distributional structure. *Word*, 10(23):146–162, 1954.

Samuel Hayakawa. *Choose the Right Word: A Contemporary Guide to Selecting the Precise Word for Every Situation*. Collins, 1994.

Karl Moritz Hermann and Phil Blunsom. The role of syntax in vector space models of compositional semantics. In *ACL (1)*, pages 894–904. Citeseer, 2013.

Felix Hill, Kyunghyun Cho, Sebastien Jean, Coline Devin, and Yoshua Bengio. Embedding word similarity with neural machine translation. *arXiv preprint arXiv:1412.6448*, 2014.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. Learning to understand phrases by embedding the dictionary. *arXiv preprint arXiv:1504.00548*, 2015a.

Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(665–695), 2015b.

Graeme Hirst. *Semantic interpretation and the resolution of ambiguity*. Studies in natural language processing. Cambridge University Press, Cambridge, England, 1987. <a href=http://www.amazon.com/exec/obidos/tg/detail/-/052142898X/>Buy at Amazon.com</a>.

Graeme Hirst. Near-synonymy and the structure of lexical knowledge. In *Working notes, AAAI Symposium on Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity*, pages 51–56, 1995.

Graeme Hirst and Alexander Budanitsky. Correcting real-word spelling errors by restoring lexical cohesion. *Natural Language Engineering*, 11(01):87–111, 2005.

Graeme Hirst and David St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 305–332. 1998.

Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

Florentina Hristea. Recent advances concerning the usage of the Naïve Bayes model in unsupervised word sense disambiguation. *International Review on Computers & Software*, 4(1): 58–67, 2009.

Eric Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 873–882, 2012.

Diana Inkpen. A statistical model for near-synonym choice. *ACM Transactions on Speech and Language Processing*, 4(1):1–17, 2007.

Diana Inkpen and Graeme Hirst. Acquiring collocations for lexical choice between near-synonyms. In *SIGLEX Workshop on Unsupervised Lexical Acquisition, the 40th Annual Meeting of the Association for Computational Linguistics*, pages 67–76, 2002.

Diana Inkpen and Graeme Hirst. Building and using a lexical knowledge-base of near-synonym differences. *Computational Linguistics*, 32(2):223–262, 2006.

Ozan İrsoy and Claire Cardie. Modeling compositionality with multiplicative recurrent neural networks. *arXiv preprint arXiv:1412.6577*, 2014.

Aminul Islam and Diana Inkpen. Second order co-occurrence pmi for determining the semantic similarity of words. In *Proceedings of the International Conference on Language Resources and Evaluation, Genoa, Italy*, pages 1033–1038, 2006.

Aminul Islam and Diana Inkpen. Near-synonym choice using a 5-gram language model. *Research in Computing Sciences*, 46:41–52, 2010.

Mario Jarmasz and Stan Szpakowicz. Roget's thesaurus and semantic similarity. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 212–219, 2003.

Jay Jiang and David Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, pages 19–33, 1997.

William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014.

Pentti Kanerva, Jan Kristofersson, and Anders Holst. Random indexing of text samples for latent semantic analysis. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, pages 103–108, 2000.

Jussi Karlgren and Magnus Sahlgren. From words to understanding. *Foundations of Real-World Intelligence*, pages 294–308, 2001.

Adam Kilgarriff and Joseph Rosenzweig. Framework and results for English Senseval. *Computers and the Humanities*, 34(1-2):15–48, 2000.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Proceedings of Advances in Neural Information Processing Systems*, pages 3276–3284, 2015.

Hideki Kozima and Akira Ito. Context-sensitive measurement of word distance by adaptive scaling of a semantic space. *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 111–124, 1997.

George Lakoff. Linguistic gestalts. In *Papers from the Regional Meeting. Chicago Linguistic Society, Chicago, Ill.*, volume 13, pages 236–287, 1977.

Thomas Landauer and Susan Dumais. A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.

Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1188–1196, 2014.

Claudia Leacock and Martin Chodorow. Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283, 1998.

Alessandro Lenci. Distributional semantics in linguistic and cognitive research. *Italian journal of linguistics*, 20(1):1–31, 2008.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, pages 24–26, 1986.

Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 302–308, 2014.

Wentian Li. Random texts exhibit Zipf's-law-like word frequency distribution. *IEEE Transactions on Information Theory*, 38(6):1842–1845, 1992.

Yuhua Li, David McLean, Zuhair Bandar, James O'Shea, and Keeley Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering*, 18(8):1138–1150, 2006.

Dekang Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 768–774, 1998.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. Identifying synonyms among distributionally similar words. In *Proceedings of International Joint Conference of Artificial Intelligence*, pages 1492–1493, 2003.

Kenneth Litkowski. Sense information for disambiguation: Confluence of supervised and unsupervised methods. In *Proceedings of the Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 47–53, 2002.

David Luenberger. *Introduction to Dynamic Systems: Theory, Models, and Applications*. Wiley, 1979.

Rila Mandala, Takenobu Tokunaga, and Hozumi Tanaka. Combining multiple evidence from different types of thesaurus for query expansion. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–197, 1999.

Mitchell Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Diana McCarthy and Roberto Navigli. The English lexical substitution task. *Language Resources and Evaluation*, 43(2):139–159, 2009.

Rada Mihalcea, Paul Tarau, and Elizabeth Figa. PageRank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1126–1132, 2004.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Workshop Proceedings of the International Conference on Learning Representations*, 2013.

George Miller and Walter Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28, 1991.

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1781–1796, 2012.

Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 236–244, 2008.

Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010.

Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th International Conference on Machine Learning*, pages 641–648, 2007.

Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems*, pages 1081–1088, 2009.

Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of International Conference on Machine Learning*, 2012.

Saif Mohammad and Graeme Hirst. Distributional measures of concept-distance: a task-oriented evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 35–43, 2006.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 95–100. Association for Computational Linguistics, 2012.

Roberto Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69, 2009.

Shixiao Ouyang, Helena Hong Gao, and Soo Ngee Koh. Developing a computer-facilitated tool for acquiring near-synonyms in Chinese and English. In *Proceedings of the 8th International Conference on Computational Semantics*, pages 316–319, 2009.

Siddharth Patwardhan and Ted Pedersen. Using WordNet-based context vectors to estimate the semantic relatedness of concepts. *Proceedings of the EACL Workshop Making Sense of Sense — Bringing Computational Linguistics and Psycholinguistics Together*, 1501:1–8, 2006.

Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 241–257, 2003.

Ted Pedersen. A simple approach to building ensembles of Naive Bayesian classifiers for word sense disambiguation. In *Proceedings of the 1st Conference of North American Chapter of the Association for Computational Linguistics*, pages 63–69, 2000.

Ted Pedersen and Rebecca Bruce. Knowledge lean word-sense disambiguation. In *Proceedings of the 10th Annual Conference on Innovative Applications of Artificial Intelligence*, pages 800–805, 1998.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of English words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 183–190, 1993.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. SemEval-2007 task 17: English lexical sample, SRL and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 87–92, 2007.

Willard Van Orman Quine. *Ontological relativity and other essays*. Number 1. Columbia University Press, 1969.

Roy Rada, Hafedh Mili, Ellen Bicknell, and Maria Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 19(1):17–30, 1989.

Reinhard Rapp. The automatic generation of thesauri of related words for English, French, German, and Russian. *International Journal of Speech Technology*, 11(3):147–156, 2008.

Ariel Raviv, Shaul Markovitch, and Sotirios-Efstathios Maneas. Concept-based approach to word sense disambiguation. In *Proceedings of the 26th Conference on Artificial Intelligence*, pages 807–813, 2012.

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

Philip Resnik. Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11(11):95–130, 1999.

Ray Richardson and Alan Smeaton. Using wordnet as a knowledge base for measuring semantic similarity between words. Technical report, School of Computer Applications, Dublin City University, 1994.

Peter Roget. *Roget's Thesaurus of English Words and Phrases*. TY Crowell Co., New York, NY, 1911.

JV Rosenhead. The advanced theory of statistics: Volume 2: Inference and relationship. *Journal of the Operational Research Society*, 14(1):97–98, 1963.

Herbert Rubenstein and John Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, 1965.

Magnus Sahlgren. An introduction to random indexing. In *Proceedings of Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering*, 2005.

Magnus Sahlgren and Rickard Cöster. Using bag-of-concepts to improve the performance of support vector machines in text categorization. In *Proceedings of the 20th international conference on Computational Linguistics*, page 487. Association for Computational Linguistics, 2004.

Magnus Sahlgren and Jussi Karlgren. Automatic bilingual lexicon acquisition using random indexing of parallel corpora. *Natural Language Engineering*, 11(03):327–341, 2005.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211, 2012.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Philip J Stone, Dexter C Dunphy, and Marshall S Smith. The general inquirer: A computer approach to content analysis. 1966.

Michael Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In *Proceedings of the 2n International Conference on Information and Knowledge Management*, pages 67–74, 1993.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1139–1147, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

Kaisheng Tai, Richard Socher, and Christopher Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 1556–1566, 2015.

Wilson L Taylor. Cloze procedure: a new tool for measuring readability. *Journalism and Mass Communication Quarterly*, 30(4):415, 1953.

David Tugwell and Adam Kilgarriff. Wasp-bench: a lexicographic tool supporting word sense disambiguation. In *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 151–154, 2001.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, 2010.

Peter Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning*, pages 491–502, 2001.

Florentina Vasilescu, Philippe Langlais, and Guy Lapalme. Evaluating variants of the Lesk approach for disambiguating words. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 633–636, 2004.

Tong Wang and Graeme Hirst. Near-synonym lexical choice in latent semantic space. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1182–1190, 2010.

Tong Wang and Graeme Hirst. Refining the notions of depth and density in WordNet-based semantic similarity measures. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1003–1011. Association for Computational Linguistics, 2011.

Tong Wang and Graeme Hirst. Exploring patterns in dictionary definitions for synonym extraction. *Natural Language Engineering*, 18(03):313–342, 2012.

Tong Wang and Graeme Hirst. Applying a naive Bayes similarity measure to word sense disambiguation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 531–537, 2014.

Tong Wang, Abdel-rahman Mohamed, and Graeme Hirst. Learning lexical embeddings with syntactic and lexicographic knowledge. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 458–463, 2015.

Noah Webster. *Webster's seventh new collegiate dictionary*. G. & C. Merriam Comp., 1971.

Yorick Wilks, Dan Fass, Cheng-Ming Guo, James McDonald, Tony Plate, and Brian Slator. Providing machine tractable dictionary tools. *Machine Translation*, 5(2):99–154, 1990.

Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 133–138, 1994.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196, 1995.

Dani Yogatama, Manaal Faruqui, Chris Dyer, and Noah Smith. Learning word representations with hierarchical sparse coding. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 87–96, 2014.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

Yinggong Zhao, Shujian Huang, Xinyu Dai, Jianbing Zhang, and Jiajun Chen. Learning word embeddings from dependency relations. In *Proceedings of 2014 International Conference on Asian Language Processing*, pages 123–127, 2014.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015.

Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. Long short-term memory over recursive structures. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1604–1612, 2015.