University of Sussex

EVALUATING DISTRIBUTIONAL MODELS OF COMPOSITIONAL SEMANTICS

MIROSLAV MANOV BATCHKAROV

Submitted for the degree of Doctor of Philosophy
University of Sussex
September 2015

SUPERVISORS:
David Weir
Bill Keller

DECLARATION

I hereby declare that this thesis has not been and will not be submitted in whole or in part to another University for the award of any other degree.

Signature:

Miroslav Manov Batchkarov

UNIVERSITY OF SUSSEX
MIROSLAV MANOV BATCHKAROV
SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
TITLE: EVALUATING DISTRIBUTIONAL MODELS OF COMPOSITIONAL SEMANTICS

ABSTRACT

Distributional models (DMs) are a family of unsupervised algorithms that represent the meaning of words as vectors. They have been shown to capture interesting aspects of semantics. Recent work has sought to compose word vectors in order to model phrases and sentences. The most commonly used measure of a compositional DM's performance to date has been the degree to which it agrees with human-provided phrase similarity scores.

The contributions of this thesis are three-fold. First, I argue that existing intrinsic evaluations are unreliable as they make use of small and subjective gold-standard data sets and assume a notion of similarity that is independent of a particular application. Therefore, they do not necessarily measure how well a model performs in practice. I study four commonly used intrinsic datasets and demonstrate that all of them exhibit undesirable properties.

Second, I propose a novel framework within which to compare word- or phrase-level DMs in terms of their ability to support document classification. My approach couples a classifier to a DM and provides a setting where classification performance is sensitive to the quality of the DM.

Third, I present an empirical evaluation of several methods for building word representations and composing them within my framework. I find that the determining factor in building word representations is data quality rather than quantity; in some cases only a small amount of unlabelled data is required to reach peak performance. Neural algorithms for building single-word representations perform better than counting-based ones regardless of what composition is used, but simple composition algorithms can outperform more sophisticated competitors. Finally, I introduce a new algorithm for improving the quality of distributional thesauri using information from repeated runs of the same non-deterministic algorithm.

ACKNOWLEDGMENTS

I would like to acknowledge the following people:

- My supervisors David Weir, Bill Keller and Jeremy Reffin (who often ended up supervising me) for always being available for a chat and for almost literally holding my hand throughout my time at Sussex. I am especially grateful to David and Jeremy for schooling me in art of sniffing out suspicious results, structuring messy thoughts and playing squash. I hope to one day write as eloquently and play as aptly as these guys.

- Other DisCo team members as Sussex — Daoud Clarke, David Sheldrick, Julie Weeds and Roland Davis. They filled in the gaps in my understanding of distributional semantics and always had the patience to sit through earlier versions of this work.

- The many Disco project members at the Universities of Oxford, Cambridge, York, Edinburgh and Queen Mary London for talking about and thus forcing me to read up on subjects I would have never considered otherwise.

- Past and present members of the Text Analysis Group at Sussex — Aleksandar Savkov, Andy Robertson, David Sheldrick, David Spence, Hamish Morgan, Jack Pay, Joe Taylor, Matti Lyra, Roland Davis, Simon Wibberley and Thomas Kober.

This thesis would not have been possible without your help. Thank you!

Lastly and most importantly, a few personal words. I could not be more grateful to my parents and my sister. You have taught me the value of hard work and curiosity, without which I would not have managed to get to here. I would not be writing this if it wasn't for the opportunities you have given me. I hope to get many more chances to make you proud in the future.

I am also grateful to my better half Stana for (quite literally) holding my hand throughout this adventure. I surely would have had a meltdown without you. I do apologise for all those times when a social event turned into a PhD conversation — you have endured too many of those.

This work was funded by UK EPSRC project EP/IO37458/1 "A Unified Model of Compositional and Distributional Compositional Semantics: Theory and Applications".

CONTENTS

1

INTRODUCTION

Philosophers and cognitive scientists have long theorised about the nature of language and how it can be used to convey meaning. More recently, computers have offered the possibility of automating natural language processing. However, machines are limited in their ability to understand language expressions. As a result, interest in ways of building and reasoning with computer representations of natural language semantics has been increasing. Two major strands of work have been investigated in the past. Formal semantics originates in logic and makes heavy use of formal mathematical machinery. Focus is on building logical representations of sentences, reasoning over these and linking them to the real world. Distributional semantics assumes meaning stems from language itself and is defined by how a word or a phrase is used within a language (distributional hypothesis). Distributional models have traditionally focused on lexical representations; attempts to model syntactic structure are still comparatively new. Recent computational research has made heavy use of this philosophical line of work by translating it into a range of computer algorithms that have been very successful in modelling certain aspects of meaning. One of the main advantages of distributional algorithms is that they offer a way of acquiring semantic representations without human intervention, which makes them very appealing from an engineering point of view.

This thesis aims to re-think how distributional models of semantics (DMs) are evaluated. Currently the most commonly used measure of performance is the degree to which DMs agree with human-provided similarity scores between pairs of words or phrases. I will argue that such evaluations are unreliable because they do not necessarily measure the utility of different ways of instantiating DMs for downstream tasks. Another issue is the lack of clarity as to what "similarity" means. Intrinsic gold-standard data sets assume there exists a single notion of similarity that is independent of a particular application. Further, the word similarity task is hard for human annotators to do reliably, which casts doubt on the suitability of such data sets for the evaluation of semantic models.

To address these issues, I propose a novel framework within which to compare distributional models in terms of their ability to support document classification, a generic and challenging task of practical importance. I also present several case studies of the kinds of qualitative and quantitative understanding that can be obtained by evaluating current state-of-the-art distributional models in my framework. I specifically focus on a range of extensions to distributional models which attempt to compose distributional phrase representations out of word representations.

The rest of this chapter introduces formal and distributional semantics at a high level and explains the contributions of this thesis in more detail.

1.1 FORMAL SEMANTICS

The term "formal semantics" refers to a family of methods which originate in mathematical logic and linguistics and attempt to systematically study 1) how semantic representation can be associated with expressions in natural language, and 2) how these semantic representations can be used for drawing conclusions (Manning, 2005). In this section I briefly review some of the more influential approaches proposed to date. I take the liberty of conflating a vast body of work in order to emphasise some of the key properties and assumptions, and in order to contrast it with distributional semantics, the main subject of this thesis.

1.1.1 *Montague semantics*

Montague semantics is broadly in the tradition of Alfred Tarski and was popularised (and named after) Richard Montague (Montague, 1970a,b, 1973). Its key idea is that no "important theoretical difference exists between formal and natural languages" (Montague, 1970b). Montague semantics is denotational (or referential) in nature. The main function of language is to allow us to talk about possible states of affairs; meaning is grounded in, and originates from, the world.

Montague semantics is defined by several core principles. First, it is truth-conditional. This means the meaning of a statement is a description of what the world would have to be like for the statement to be true. The world therefore serves as the set of necessary and sufficient conditions for the statement to be true. For example, the statement

Domain	Model
John	j
Kathy	k
tall	$\{j\}$
likes	$\{(j,k),(k,j)\}$

Table 1.1: A simple domain

"Brighton is north of London" is true if and only if the entity referred to as "Brighton" is in a particular spacial relation to the entity named "London", and the relation is named "north of". Note that the sentence and the state of affairs in the world are separate. More formally, a statement can be *interpreted* in many possible worlds, and it is only true or false with respect to a particular possible world. This is referred to as "possible-world semantics".

Another key property of Montague semantics is that it is model-theoretic. A model is a abstract mathematical object that represents the particular state of affairs in the world, which is expressed in a *metalanguage*. Several alternative metalanguages have been proposed, such as first order logic (FOL), linear logic (Girard, 1987) or frames (Fillmore, 1982). Using FOL expressions has historically been particularly popular as it is well-understood and has a clear semantics. There are two kinds of valid expressions in FOL:

TERMS which intuitively correspond to domain objects. More formally, they are said to have a corresponding denotation in the metalanguage. Consider the simple domain given in Table 1.1. The model constants j and k denote the real-world individuals *John* and *Kathy*. The denotation of the property *tall* in the domain is the set of all metalanguage constants which possess that property (*John*, assuming that John is tall and Kathy is not). The denotation of relations such as *likes* is the set of ordered tuples of constants which are in that relation. The model in Table 1.1 states that John likes Kathy and Kathy likes John.

FORMULAS which express statements that can be true or false, as defined above.

The logical metalanguage serves as an intermediate step between language and the real world, as both can be described in it. Sentences in a natural language can be translated into a logical representation because they are not merely strings, but also have an internal *syn-*

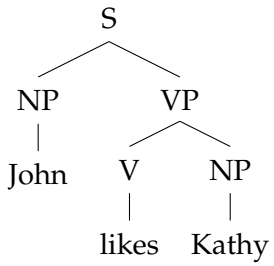


Figure 1.1: Constituency parse tree for "John likes Kathy"

tactic structure. Often the logical structure of a sentence mirrors its syntactic structure. Let us translate the sentence "John likes Kathy" into a logical form. First, we analyse the sentence syntactically (Figure 1.1). The translation of the word "John" is the constant j and the translation of "Kathy" is k. The transitive verb "to like" is seen as function that takes two arguments (constants), denoted in lambda calculus as $\lambda y.\lambda x.like(x,y)$. The semantic structure would be derived by two function applications as follows (using the notation of Blackburn and Bos (2005), where @ denotes function application):

$$\begin{aligned} \lambda y.\lambda x.like(x,y)@k &= \lambda x.like(x,k) \\ \lambda x.like(x,k)@j &= like(j,k) \end{aligned}$$

This bottom-up approach of constructing a representation for compound expressions is known as *composition*. The most cited formulation of the underlying *principle of compositionality* states that "The meaning of a compound expression is a function of the meanings of its parts and of the way they are syntactically combined." (Partee 1984, in Janssen, 2012b). Compositionality is one of the cornerstones of formal semantics. However, it is a somewhat contested idea as examples can be produced where the meaning of the whole is not a function merely of the parts and how they are combined, but also of a third component such as linguistic or extralinguistic context. I will return to the issues of composition in Chapter 2.2, where I will also discuss composition in a non-Montague setting.

For computational semantics, the use of a meaning representation language such as FOL has a number of benefits:

INFERENCE The rules of FOL allow us to infer new facts within the model. For example, $Mortal(Socrates)$ follows from $\forall x : Man(x) \implies Mortal(x)$ and $Man(Socrates)$

CONSISTENCY CHECKING Logical formulae that are satisfiable in at least one model are called consistent. For example, $short(John)$ is not true when interpreted in the model in Table 1.1, but may be true in some other models. In contrast, $short(John) \wedge \neg short(John)$ is not satisfiable in any model and is said to be inconsistent.

QUERYING Given a model and a FOL formula, is the formula satisfied in that particular model (Blackburn and Bos, 2005, p 20)? This task has important practical applications as it allows us to answer questions using the knowledge embodied in a model.

From a computational perspective, formal semantics is appealing because it offers a well-studied set of components, from translation of language into a logical form to querying and inference, which can be combined into an end-to-end system, e. g. for the purposes of question answering (Berant et al., 2013).

1.1.2 *Minimal recursion semantics*

Minimal recursion semantics (MRS) postulates that "the primary units of interest for computational semantics are elementary predications or EPs" (Copestake et al., 2005, p 2). An EP is "a single relation with its associated arguments (for instance, beyond(x, y))". The general principle is that recursion is avoided and an EP never contains other EPs, which creates a flat syntactic representation in the form of a list of EPs. The motivation for this is that the binary nature of the logical conjunction operator $\wedge$ creates undesirable spurious ambiguities. For example, "fierce black cat" can be represented in logical form both as $\text{cat(x)} \wedge (\text{black(x)} \wedge \text{fierce(x)})$ and $(\text{cat(x)} \wedge \text{black(x)}) \wedge \text{fierce(x)}$ (Copestake et al., 2005, p 2). In the flat MSR representation the ambiguity is removed by representing the phrase as a bag or predicates which are implicitly conjoined: cat(x), black(x), fierce(x). A similar flattening transformation is applied to the predicate representation of generalised quantifiers, which intentionally leaves their scope underspecified.

As Copestake et al. (2005) point out, " [t]he point of MRS is not that it contains any particular new insight into semantic representation, but rather that it integrates a range of techniques in a way that has proved to be very suitable for large, general-purpose grammars". Practical applications of the theory include parsing, language genera-

tion and machine translation (Copestake et al., 1995; Copestake, 1995; Carroll et al., 1999).

1.1.3 *Meaning-text theory*

Meaning-text theory (MTT) originated in the Soviet Union in the 1960s (Mel'čuk and Polguere, 1987; Mel'čuk, 1999; Milićević, 2006). It is a global model in that it studies together all aspects of natural language, from semantics to morphology and phonology. In contrast to mainstream Western approaches to semantics, which are predominantly analytic in nature, MTT focuses on synthesis of natural language utterances. This is because its main tenet is that language is above all used to express meaning, so any model of natural language must start with the semantics as its main object of interest. In that sense, MTT is a functional rather than a structural model of language. The study of paraphrases therefore plays a central part in MTT.

The main object of MTT is not sets of objects but rather the relations between them. The most important such relation is the many-to-many mapping between meanings and utterances. Although this correspondence is bidirectional, MTT concentrates on the passage from semantics to utterances (language generation) rather than vice versa. That process involves several intermediate representations. At the first level, semantics, utterances are represented as unordered semantic networks, which capture their predicate-argument structure. The next level of description is syntax, which is entirely modelled as a dependency tree rather than a constituency tree. For morphologically complex languages morphological dependencies may also be included to account for word declension. The final description is the phonological one, where an utterance is modelled as a sequence of phonemes. Representations are converted from one level to another using a rich set of rules based on introspection. However, no claim is made as to the psychological plausibility of such a model.

The lexicon plays a vital role in MTT, as any rule-based transformation from semantic to syntactic representations relies on it. Lexicographic work has therefore been very well represented in the area (Mel'čuk and Zholkovsky, 1984).

1.1.4 *Glue semantics*

Glue semantics (Dalrymple, 1999) is theory of syntax-semantics interface based on Lexical Function Grammar (Dalrymple , editor). Its starting point is that phrase structure trees are not necessarily appropriate for representing functional organisation. Therefore, glue semantics uses two levels of representations: *c-structure* and *f-structure*, and rules (projection functions) to convert from one to the other. The role of the former is to model the constituency structure of a sentence, while the latter represents its structural organisation and is reminiscent of syntactic dependencies. Semantic composition is "mainly determined by syntactic relations such as subject-of, object-of, modifier-of, and so on" (Dalrymple, 1999, p 8). The motivation for this is that these may be realised differently in the *c-structure* across languages, but are uniform in the *f-structure*.

The rules of composition are expressed in a logical language, which serves as the glue (hence the name) for combining the meaning of words. The particular logic used in glue semantics is linear logic (Girard, 1987), which is a modification of classical Boolean logic where premises are viewed as resources and are consumed by the proof process. For instance, both A and B can be deduced from the premises A and $A \to B$. In linear logic, the premises are consumed by the deduction and the only fact that can be derived is B. This is appealing for linguistic purposes because the "semantic contributions [of words] are occurrences of information which are generated and used exactly once" (Dalrymple, 1999, p 15). Additionally, linear logic allows for an intuitive treatment of modification, where a modifier consumes an unmodified meaning and produces modified meaning.

In glue semantics, the meaning of a lexical entry is a formula in linear logic, called a "meaning constructor", which "specifies the "assembly instructions" for combining the meaning contributions of the syntactic arguments of the lexical entry to obtain the meaning contribution of the whole entry" (Dalrymple, 1999, p 14). Consider the following simplified example, which is due to (Dalrymple, 1999). The meaning constructor for the verb "greet" would be

$$\forall X, Y.\text{SUBJ} \rightsquigarrow X \otimes \text{OBJ} \rightsquigarrow Y \multimap f_\sigma \rightsquigarrow \mathit{greet}(X, Y)$$

which is interpreted as

- if the subject (SUBJ) has meaning ($\rightsquigarrow$) X

- and $(\otimes)$ the object OBJ has meaning Y
- then $(\multimap)$ "greet" has meaning $greet(X, Y)$

In the above formula, it is assumed that the syntactic arguments of a verb are linked to a semantic functions, i. e. that the subject of the verb (syntax) is the agent (semantics) and the object (syntax) is the patient (semantics). This information needs to be encoded in the lexicon.

1.1.5 *Lexical semantics*

The discussion above, as well as the remainder of this thesis, is predominantly concerned with the issue of composition. However, lexical semantics has also received significant attention in the past. In this section, I will briefly review several influential theories.

Frame semantics (Fillmore, 1982) states that the meaning of a word can only be understood in the context of a particular real-world situation, which is referred to as a frame. Formally, a frame is defined as "a system of concepts related in such a way that to understand any of them you have to understand the whole structure in which it fits; when one of the things in such a structure is introduced into a text [...] all of the others are automatically made available" (Fillmore, 1982, p. 111). For example, to understand the word *breakfast* "is to understand the practice in our culture of having three meals a day, at more or less conventionally established times of the day, and for one of these meals to be the one which is eaten early in the morning..." (Fillmore, 1982, p. 118). A word is said to *evoke* or *activate* a semantic frame. Words also serve a particular argument role in a frame, or fill a slot. For example, the word "sell" evokes the commercial transaction frame from the perspective of the seller, and "buy" activates it from the perspective of the buyer. Other words that evoke the same frame include "goods" and "money". In the sentence "Abby bought a car from Robin", "Abby" fills the buyer slot of the frame, "Robin" fills the seller slot and "car" fills the "goods" slot (Baker et al., 1998).

The Generative Lexicon presents theory of how "a core set of word senses [...] is used to generate a larger set of word senses when individual lexical items are combined with others in phrases and clauses" (Pustejovsky, 1995, p. 2). This contrast to the static view of word meaning, which Pustejovsky calls "sense enumerating lexicons", where each word is characterized by a predetermined num-

ber of word senses. The generative lexicon makes use of four levels of representation: Lexical Typing Structure, which "determines the ways in which a word is related to other words in a structured type system"; Argument Structure, which "encodes the conventional mapping from a word to a function"; Event Structure, "identifies the particular event type for a verb or a phrase", such as a State, Process or Transition; and Qualia Structure, which "defines the essential attributes of objects, events, and relations, associated with a lexical item". For example, the argument structure of the verb "build" is

$$\begin{aligned} ARG_1 &= animate\_individual \\ ARG_2 &= artefact \\ ARG_D &= material \end{aligned}$$

where ARG_D is the default argument, which participates in the logical expression, but is not necessarily expressed syntactically (Pustejovsky, 1995, pp. 63-66).

The qualia structure is further subdivided into four categories, modelling the origin, purpose and relations of an object to other objects (Pustejovsky, 1995, ch. 6).

1.2 DISTRIBUTIONAL LEXICAL SEMANTICS

An alternative approach to language semantics is based on the idea that one can discover the meaning of a word by inspecting its uses in a corpus of text. Consider the following example, due to Lazaridou et al. (2014):

> We saw a cute little *wampimuk* sleeping in the tree.

A single example of the unknown word "wampimuk" being used in context is sufficient for a reader to get an idea of what it might mean. With additional examples, one might be able to fully (or sufficiently well) characterise the meaning of "wampimuk". This idea has precursors in the later philosophy of Wittgenstein (*"meaning is use"* hypothesis) and the Firthian school of linguistics — "You shall know a word by the company it keeps" (Firth and Palmer, 1968). The *distributional hypothesis* (Harris, 1954) states that similar words tend to appear in similar contexts. Practical computer-based implementations date back to the work of Sparck-Jones (1965), Deerwester et al.

(1990) and Grefenstette (1994). The meaning of a word is thus the set of contexts it occurs in. If all possible contexts are ordered, each word can be described by a vector[1], enabling the use of a wide range of tools from linear algebra. Notably, the distance between the vectors corresponding to two words or phrases can be seen as a measure of similarity between them. Another important property is that this definition is effectively an algorithm for learning such representations from data[2]. Also, enumerating the contexts of a word does not require any annotated data or human intervention.

However, distributional semantics is not a silver bullet. Meaning is assumed to be grounded in language and not in real-world objects, which is a controversial view in philosophy and cognitive science. It is debatable if one could really know what "wampimuk" is without having seen, touched or smelled one. Additionally, word-level distributional models cannot be directly applied to longer phrases because phrases occur much less frequently, making it impossible to extract reliable co-occurrence statistics. This problem of how to build up distributed representations for phrases, sentences and documents out of word representation has seen increasing attention recently. Current approaches are reviewed in detail in Chapter 2. The framework proposed in this thesis can be used to evaluate both word-level and phrase level distributional representations.

1.3 THESIS STRUCTURE AND CONTRIBUTIONS

The rest of this thesis is structured as follows. Chapter 2 presents the background information necessary for reading this thesis. I begin by introducing the area of distributional compositional semantics and discussing the main open issues and trends in the literature. I describe several commonly used algorithms for learning distributional representations of single words from unlabelled text. Next, I outline recent attempts to perform composition in a distributional setting, i. e. to combine word representations into phrase representations. Emphasis is on algorithms that deal with noun phrase and verb phrase composition.

The second part of the chapter is concerned with how compositional distributional models can be evaluated. By far the most com-

1 The terms "vector" and "embedding" will henceforth be used interchangeably to refer to the distributional representation of words.

2 An example of how distributional representations can be derived from text and a detailed description of algorithms for doing so are shown in Chapter 2.

mon family of evaluation procedures currently are the so called intrinsic ones, which attempt to directly measure the "inherent" quality of a word or phrase representation. This often takes the form of computing the extent to which a computational model agrees with human-provided word or phrase similarity judgements. I argue that these gold-standard data sets are too small and subjective to provide a thorough measure of the utility of a distributional model for downstream natural language processing, and provide empirical evidence to support that claim. Further, such evaluations assume there exists a single gold-standard similarity score between a pair of words or phrases, which is independent of any particular application.

Chapter 3 presents an empirical assessment of four commonly used intrinsic data sets. I demonstrate all four evaluations exhibit undesirable properties such as being unable to detect if random noise is added to word vectors. This motivates the introduction of a novel procedure for assessing distributional models extrinsically through text categorisation. It applies the principle of feature expansion from information retrieval by training a statistical document classifier as usual, but at test time replacing all phrases with their nearest neighbours according to a distributional model. Intuitively, the classifier is only allowed to view the contents of a document through the prism of a distributional model. Classifier accuracy is therefore strongly dependent on the quality of the distributional model. This framework is independent of a particular distributional model and can be applied to both word-level or phrase-level models. It also allows us to evaluate models against a different notion of similarity by plugging in a different collection of documents for classification.

Chapter 4 presents an empirical comparison of several state-of-the-art composition algorithms in the proposed framework. I highlight the importance of training distributional models on clean unlabelled data that matches the domain where the model is being applied. I demonstrate simple methods for building phrase representations out of word representations match or exceed the utility of more sophisticated proposals when dealing with noun phrases and verb phrases. Finally, I present a simple procedure that uses information contained in multiple comparable distributional models to reorder and improve the nearest semantic neighbours of a noun phrase.

The contributions of this thesis are three-fold:

1. Empirical evidence of the disadvantages of intrinsic evaluations, which are still the predominant method for assessing distributional models;

2. A novel extrinsic evaluation framework based on document classification;

3. An evaluation of several popular distributional models in the proposed framework.

2

LEARNING WORD AND PHRASE REPRESENTATIONS

As discussed in Chapter 1, distributional models (DMs) are an attractive approach to natural language semantics as they can be learnt from unlabelled text without human supervision. This chapter introduces the area in more detail; it is divided into four parts.

Section 2.1 describes existing algorithms for building distributed representations of single words. I begin with an example of building word vectors using an early instantiation of the distributional hypothesis, which produces the so called counting vectors. Next, I formalise the notion of counting vectors and contrast them with the recently proposed family of neural-network algorithms.

Section 2.2 introduces composition, the process of combining word representations into phrase representations. I review four broad types of algorithms, based on element-wise operations, linear regression, recursive neural networks and category-theoretical grammar.

Section 2.3 discusses two broad types of methods of evaluating the quality of DMs. Intrinsic methods attempt to directly assess DMs, typically by measuring the degree to which they agree with human-provided word and phrase similarity scores. In contrast, extrinsic evaluations embed a DM in a practical task and look for improvement in performance in that task.

Finally, Section 2.4 discusses several open questions in distributional semantics, which represent important directions for future research.

Chapter 3 analyses four commonly used intrinsic evaluation data sets and demonstrates all four exhibit undesirable properties such as being unable to detect if random noise is added to word vectors. This motivates the introduction of a novel procedure for assessing distributional models extrinsically through text categorisation.

2.1 WORD REPRESENTATIONS

To illustrate the basic premise of distributional semantics, let us construct a vector representation for three words: "cat", "dog" and "orange". Suppose the following corpus of free text is provided:

did not think	*dog*	owners should
a filthy street	*dog*	ran towards
investigate recent	*dog*	bites
one even gave my	*dog*	a biscuit
a growl that only a	*dog*	would provide
accompanied by a	*dog*	on a leash
saw a thin little	*cat*	lapping milk
had a	*cat*	, a dusty little creature
she was called the	*cat*	because of her fierce manners
a nice fluffy	*cat*	ran my way
bought a tasty	orange	
Is this	*orange*	juice
	⋮	
an	*orange*	object hit the floor

The three words of interest will henceforth be referred to as *entries*; the contexts that an entry is used in are its *distributional features*. Features and entries are said to co-occur. For the sake of simplicity, this example focuses on the three entries above and uses all other content words occurring in the same sentence as features. Specifically, the features of the entry "dog" in the second sentence above are "filthy" and "street". The set of contexts for all entries can be laid out as a matrix (Table 2.1), where the rows correspond to entries, the columns correspond to features and each cell corresponds to how often a feature has occurred in the context of an entry. Each row represents the vector (distributional representation) for a particular entry, i. e. the vector for cat is $\overrightarrow{cat} = [9, 12, 5, 0, 1, 0, \ldots]$ and the one for dog is $\overrightarrow{dog} = [3, 4, 15, 0, 0, 0, \ldots]$[1]. When entries are viewed as vectors, a wide

1 Formally, this is the definition of a point in Euclidean space, which is equivalent to a vector whose initial point is the origin.

		Features						
		filthy	street	bite	tasty	buy	juicy	...
Entries	dog	3	4	15				...
	cat	9	12	5		1		...
	orange	1		2	4	7	25	...

Table 2.1: Three entries and some of their distributional features. The number is this table are hypothetical and do not come from the example corpus above.

range of tools from linear algebra can be leveraged. For instance, the Euclidean distance between two vectors v and w is defined as

$$d(\overrightarrow{v}, \overrightarrow{w}) = \sqrt{\sum_i (\overrightarrow{v}_i - \overrightarrow{w}_i)^2}$$

A number of other measures that exhibit desirable properties have been proposed — refer to Weeds (2003) for a comprehensive study. The Euclidean distance between "cat" and "dog", $cos(\overrightarrow{dog}, \overrightarrow{cat})$, is approximately 0.4, while $cos(\overrightarrow{dog}, \overrightarrow{orange}) \approx 0.92$. The information contained in the distributed representation of the three entries is sufficient to establish that "cat" is semantically closer to "dog" than to "orange".

2.1.1 *Counting distributional models*

The distributional representations above were built using a simple instantiation of a class of DMs that Baroni et al. (2014) termed "counting" algorithms, which were popularised by early work such as Sparck-Jones (1965), Redington et al. (1993), Schütze (1993) and Grefenstette (1994). These models count the number of times a feature appears in the context of a target entry. Features are typically taken to be other words or short phrases. A feature is usually considered to be in the context of an entry if the two occur within a distance of k words in the same sentence. These are known as "proximity" or "window"features[2]. For example, in the sentence "John likes black cats" the features of the entry "likes" would be the set $\{John, black\}$ if using a symmetric window of size 1 and $\{cats, John, black\}$ if using a window of size 2 and single words as features. Alternatively, adja-

2 Another popular alternative, called "dependency features", is discussed in Section 2.1.3

cent bigrams can be used as features, in which case the features of "likes" would be the set $\{black\_cats\}$.

After counting how many times each feature and entry co-occur, the resultant counts are typically weighted by a factor that reflects their informativeness. This reflects the intuition that not all contexts are equally important. For instance, given a large enough window, most entries would have common words such as "the" and "a" as features. Using the raw un-weighted counts would not account for entries and features co-occurring by chance and would distort the probability distribution of entries over possible contexts. Co-occurrences that are likely to have been observed by chance are not likely to be informative and are therefore given a lower weight, and vice versa. Common re-weighting schemes include (positive) pointwise mutual information, log-likelihood ratio and χ^2 (Evert, 2005). Pointwise mutual information is a measure of association between an entry e and a feature f that accounts for the probability that they co-occur by chance. It is defined as

$$PMI(e, f) = \log \frac{P(e, f)}{P(e)P(f)}$$

$P(e)$ is the prior probability of the entry e occurring, which is the ratio between the number of occurrences of e and the total number of entry occurrences. $P(f)$ is the prior probability of the feature f, which is calculated in a similar fashion. $P(e, f)$ is the joint probability of e and f. A commonly used correction, which has been argued to be superior (Evert, 2005; Bullinaria and Levy, 2007, 2012), is positive PMI, defined as

$$PPMI(e, f) = \begin{cases} 0 & \text{if } PMI(e, f) < 0 \\ PMI(e, f) & otherwise \end{cases}$$

The vectors that result from using a counting approach are interpretable, as each dimension corresponds to a particular feature. However, these vectors are also often very high-dimensional. Because the memory footprint and running time of common linear-algebraic algorithms are a function of dimensionality, raw count vectors are often impractical to work with. Dimensionality reduction (DR) techniques such as Singular Value Decomposition (SVD) or Non-negative Tensor Factorisation are commonly employed to address these practical issues (Turney and Pantel, 2010). A secondary benefit of DR is that the

resultant entry vectors are more robust to noise in the input data and thus perform better in evaluations. For example, a notable early application of SVD to NLP is the work of Deerwester et al. (1990), which performs SVD (over a term-document matrix) in an information retrieval system. A drawback of DR is that it makes the dimensions of the resultant vectors harder to interpret, as each dimension becomes a (potentially non-linear) combination of the original interpretable distributional features. Also, reduced representations are only an approximation of the original embeddings; some information is lost in the process.

2.1.2 *Neural distributional models*

Neural distributional models formalise the intuitions behind the counting approach by encoding desirable properties of the resultant word vectors explicitly as an objective function and using a neural network to optimise that function. These models address the rigidity of the counting approaches, which typically follow a count-reweight-reduce process, with various heuristics applied at each step in an attempt to endow the model with certain desirable properties. For example, PPMI and context selection remove or reduce the weight of non-informative contexts, SVD makes word vectors dense and low-dimensional, and length normalisation accounts for frequency effects. In contrast, neural models allow for these properties to be encoded explicitly in the objective function and learnt jointly. Neural models have become very popular in recent years. Thorough surveys can be found in Schmidhuber (2015) and LeCun et al. (2015).

In this section I focus on two concrete neural semantic models, called WORD2VEC (Mikolov et al., 2013a) and GLOVE (Pennington et al., 2014). These are selected because they are very efficient to train, perform well in a range of tasks and offer free high-quality implementations. The details of these models are given below.

WORD2VEC starts off with a randomly initialised vector of fixed dimensionality for each entry. Mikolov et al. present two different formalisations of the notion of context, called continuous bag-of-words (CBOW) and skip-gram (SG). CBOW maximises the probability of an entry given the context it was observed in. In contrast, the SG model

maximises the probability of the contexts an entry occurs in given the entry. Formally, the objective function SG maximises is

$$J_{SG} = \frac{1}{|T|} \sum_{t=1}^{|T|} \sum_{-c \leq j \leq c, c \neq 0} \log P(w_{t+j}|w_t)$$

where $|T|$ is the size of the training corpus, c is the size of the context window, and w_t is the t-th word in the corpus (the entry). The CBOW objective function is

$$J_{CBOW} = \frac{1}{|T|} \sum_{t=1}^{|T|} \sum_{-c \leq j \leq c, c \neq 0} \log P(w_t|w_{t+j})$$

In both SG and CBOW, the probability of a word w_1 given another word w_2 is defined in terms of their vector representations as

$$P(w_1|w_2) = \frac{\exp(\overrightarrow{w_1} \cdot \overrightarrow{w_2})}{\sum_k \exp(\overrightarrow{w_k} \cdot \overrightarrow{w_2})}$$

where k is an index into the list of entries contained in the model. Maximising this probability amounts to maximising the similarity (dot product) between an entry and the contexts it occurs in at the expense of contexts it does *not* occur in. Intuitively, a high ratio ensures that words that appear together in the corpus have more similar vectors than words that do not. Because the number of contexts an entry does *not* occur in can be very high, the sum in the denominator can range over a very large set, making learning impractical. To alleviate this problem Mikolov et al. use a technique known as negative sampling, where the summation is over a small random sample of all contexts. The end result is that if "cat" appears in the context of "food", then the vector of "food" is made more similar to the vector of "cat" (as measured by their dot product) than the vectors of several other randomly chosen words instead of all other words in language.

In its original formulation WORD2VEC is very similar to window-based counting models. In fact, its objective function has been shown to be equivalent to a counting-based approach with mutual information weighting (Levy and Goldberg, 2014b). Since it was first introduced, WORD2VEC has been extended in several ways. Yu and Dredze (2014) add a term to the SG objective function to encourage words that are close in WordNet (Miller, 1995) to have similar embeddings. Similarly, Passos et al. (2014) propose an extension that encourages the model to assign similar embeddings to words that ap-

pear in the same section of a gazetteer. Levy and Goldberg (2014a) train a WORD2VEC model that considers dependency features (Section 2.1.3). Tang et al. (2014), Labutov and Lipson (2013) and Faruqui et al. (2014) propose ways of adapting WORD2VEC embeddings to a particular domain, such as sentiment. Ling et al. (2015) add sensitivity to word order to WORD2VEC, which results in improved performance at part-of-speech tagging and dependency parsing.

WORD2VEC has been applied to a range of tasks. Fu et al. (2014) build semantic hierarchies by identifying hypernymy, using word vectors as input to a classifier. Mikolov et al. (2013b) learn a mapping between the vector spaces for a pair of languages and use it to expand the phrase table of a machine translation system. Kågebäck et al. (2014) use WORD2VEC embeddings to identify similar sentences and improve an existing extractive multi-document summarisation algorithm. The key intuition of that work is that semantically similar sentences do not contribute to the summary and should therefore be removed.

GLOVE (Pennington et al., 2014) seeks to address some of the limitations of WORD2VEC. Its objective function is designed around three considerations:

1. WORD2VEC only leverages information contained locally, within a small window around a target entry. The potentially useful information provided by the global co-occurrence count between features and entries is ignored.

2. WORD2VEC does not explicitly attempt to weight feature co-occurences by informativeness. Pennington et al. note that non-informative features w_u are expected to co-occur with approximately equal probability with words w_i and w_j for any choice of $u \neq i \neq j$. The ratio $P(w_u|w_i)/P(w_u|w_j)$ should therefore be close to 1.

3. The distinction between an entry and a feature is somewhat arbitrary, as any word can be both. The entry-feature matrix should therefore be invariant to transposition. This is trivially true for counting models without dimensionality reduction, but neural models learn a latent low-rank approximation of that matrix, which is not invariant to transposition. GLOVE addresses that problem by assigning two vectors to each word w. When modelled as an entry, w is assigned the vector $\overrightarrow{w}$, and when it is viewed as a feature its vector is $\overrightarrow{w'}$.

GLOVE's objective function, which satisfies these desiderata is

$$J_{\text{glove}} = \sum_{i,j=1}^{|W|} f(X_{ij})(w_i^T w_{j'} + b_i + b_{j'} - \log X_{ij})^2$$

where X_{ij} is the co-occurrence count of words i and j, b is a bias term, and f is a non-decreasing function such that $f(0) = 0$ and $f(x)$ is small for large values of x.

Note that both WORD2VEC and GLOVE are unsupervised, even though their cost functions are optimised via supervised training.

2.1.3 *Context*

In the discussion so far, I assumed a particular definition of what makes a feature. I used the so called proximity vectors, where two words are considered to co-occur if they appear in the same sentence. The advantage of this approach is that it requires little or no preprocessing of the input text, and is therefore fast in practice. However, words that appear in the same sentence are not necessarily closely related.

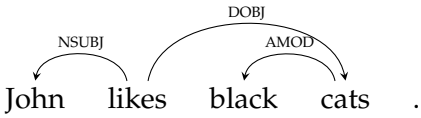


Figure 2.1: Dependency analysis of "John likes black cats".

Dependency features are a linguistically motivated alternative to proximity features. One would first use a dependency parser to analyse the syntactic structure of the sentence. Dependency analysis is a type of syntactic analysis that describes how pairs of words are related grammatically in a sentence, instead of how the sentence decomposes into sub-structures as in a constituency-based analysis. A dependency analysis of the example sentence "John likes black cats" according to the Stanford system (De Marneffe and Manning, 2008) is shown in Figure 2.1. The tree encodes the following facts:

- "John" is the nominal subject (NSUBJ) of "likes"
- "cats" is a direct object (DOBJ) of "likes"
- "black" is an adjectival (AMOD) modifier of "cats"

Dependency information can be used to extract more targeted features for the entry "likes", such as $\{has\_nsubj\_John, has\_dobj\_cats\}$. More precisely, dependency-based distributional features take into account that "black" is not directly grammatically related to "likes" even though the two words are adjacent in the corpus. Additional information can be extracted in the form of second-order dependency features (Weir et al., in press). For example, "likes" has a direct object which in turn has "black" as an adjectival modifier. The feature that captures that information would be *dobj_amod_black*.

Dependency features have been used both in counting (Lin, 1998; Erk and Padó, 2008; Grefenstette et al., 2011) and neural algorithms (Levy and Goldberg, 2014a). A practical consideration is that the corpus needs to be parsed, which is computationally intensive and cannot be done with perfect accuracy. A comprehensive overview of other issues relating to word-level distributional models can be found in Erk (2012) and Turney and Pantel (2010).

2.2 COMPOSITION

The distributional models considered so far are capable of learning a representation of the meaning of single words. However, a computational model of semantics needs to be applicable to phrases, sentences, paragraphs and ultimately documents. However, longer phrases occur much less frequently than single words regardless of the amount of text available, and many plausible phrases may not occur at all. Trivially extending the distributional models discussed so far to the phrase level is not feasible.

The *principle of compositionality* provides a possible way of addressing this limitation. Modern formulations are widely attributed to 19-th century German philosophers, most notably Frege (Janssen, 2012a). In its most commonly used form, the principle states that "[t]he meaning of a compound expression is a function of the meanings of its parts and of the way they are syntactically combined" (Partee 1984, in Janssen, 2012a). Composition is needed in order to model the meaning of the vast number of phrases and sentences that can be built out of a language's vocabulary.

I will now detail four families of compositional distributional algorithms (henceforth referred to as "composers"). I will return to the philosophical issues relating to composition in Section 2.4.

2.2.1 *Pointwise compositional models*

Mitchell and Lapata (2008) are the first to empirically evaluate and compare a number of different distributional composers. These include:

ADDITIVE The phrase vector is the pointwise sum of the vectors of the constituents: $\overrightarrow{p} = \alpha\overrightarrow{u} + \beta\overrightarrow{v}$

MULTIPLICATIVE The phrase vector is the pointwise product of the vectors of the constituents: $\overrightarrow{p} = \gamma\overrightarrow{u} \cdot \overrightarrow{v}$

In this thesis, and often in the literature, $\alpha = \beta = \gamma = 1$ for simplicity. Such pointwise models are simple to implement and perform well in practice (Mitchell and Lapata, 2008; Grefenstette et al., 2013). Since addition and multiplication are commutative, these composers are insensitive to word order, but are able to compose any phrase, sentence or document regardless of grammatical structure.

2.2.2 *Linear regression model*

The adjective linear map model of Baroni and Zamparelli (2010) focuses on adjective-noun compounds and provides an empirical method of exploiting the intuition that attributive adjectives are "functions from the meaning of a noun onto the meaning of a modified noun" (Baroni and Zamparelli, 2010, p 1183). The authors differentiate between two units of meaning: nouns and noun phrases. The latter consist of an adjectival modifier and a noun head. Vectors for both can be obtained from a corpus. These are referred to as "observed vectors" and are denoted below by $\overrightarrow{u}$ and $\overrightarrow{uv}$ for a noun and modified noun respectively. Adjectives are seen as functions from the former to the latter and are denoted as $f : \overrightarrow{u} \rightarrow \overrightarrow{uv}$. The authors assume the mapping f is linear and can be represented as an $n \times n$ matrix, which transforms n-dimensional noun vectors into n-dimensional noun phrase vectors. The values of the matrix A_u are specific to each adjective u and are empirically set to minimise the squared difference between the corpus-observed and the model-predicted noun phrase vectors. The matrix is learnt via linear partial least squares regression and is applied to a noun by means of matrix multiplication:

$$\widehat{uv} = A_u\overrightarrow{v}$$

where $\widehat{uv}$ is the composed (predicted) vector for the phrase uv.

The algorithm does not require human supervision, but involves a supervised learning step. Because corpus-observed vectors for bigrams are required, the model may have low coverage. Additionally, n^2 parameters have to be learnt from a small amount of data. Later work (Grefenstette et al., 2013; Paperno et al., 2014) attempts to extend this approach to other phrase types, such as verb-object and subject-verb-object.

In this thesis I use the 2010 formulation of the algorithm and also apply it to noun-noun compounds. I assume noun modifiers function similarly to adjectival modifiers, therefore a matrix representation for noun modifiers can be learnt similarly to how adjectival modifiers are learnt.

2.2.3 *Recursive neural network model*

In its basic form, an auto-encoder is a three-layer feed-forward neural network where the input and output layers are of size n and the hidden layer is of size $m \ll n$. Inputs are also used as targets and the goal of the network is to learn to reconstruct its inputs after they have passed through a small hidden layer. The activations of the hidden layer can be seen as a more compact representation of the original inputs. Auto-encoders operate in two stages — the inputs are first encoded into a compact representation, which is then decoded into a reconstruction. The parameters of the network are trained to minimise the difference between an input vector and its reconstruction. This is done in a supervised fashion via back-propagation, but no labelled data is required as the inputs also serve as targets.

Socher et al. (2011) propose a compositional model based on the recursive, bottom-up application of auto-encoder neural networks along the nodes of a constituency parse tree. The process is best explained with an example. I will first explain the process at a high level and will provide technical details later.

Suppose we wish to compose the sentence "$x_1\ x_2\ x_3$", whose parse tree is given by the blue nodes in Figure 2.2. A network is trained to compress a pair of n-dimensional word vectors into a single n-dimensional vector and to then reconstruct the original vectors. The encoding process is applied bottom-up to pairs of words in the parse tree that share an ancestor. For the tree in Figure 2.2, the vectors for x_2 and x_3 are first encoded (using some yet unspecified operation W_e)

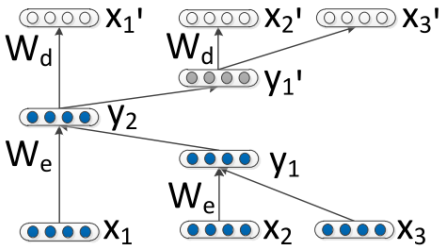


Figure 2.2: Unfolding recursive autoencoder, reprinted from Socher et al. (2011). Blue nodes denote input word embeddings and grey nodes are their reconstructions.

into a latent representation y_1, which is then combined with x_1 (again using W_e) to form y_2, the vector for the entire sentence. Note that at each encoding step two vectors of size n are composed into one while respecting the syntactic structure of the sentence.

Decoding is done similarly to encoding, but in reverse order. y_2 is first expanded into $x_1\prime$ and $y_1\prime$ (using a decoding operator W_d), which are the reconstructions of x_1 and y_1 respectively. $y_1\prime$ is then recursively decoded using W_d into $x_2\prime$ and $x_3\prime$, the reconstructions of x_2 and x_3. The aim of the algorithm is to learn W_e and W_d such that, for each $x \in \{x_1, x_2, x_3, y_1\}$, x is close to $x\prime$.

The technical details of the algorithm are as follows. First, nodes are not atomic units, but n-dimensional vectors, which are held static during training. Leaf nodes are word embeddings, and internal nodes correspond to phrase representations. W_e and W_d are matrices of size $n \times n$, which are applied to word vectors via standard matrix multiplication. Encoding is performed using the standard neural network technique:

$$\overrightarrow{y_1} = \tanh(W_e[\overrightarrow{x_2}; \overrightarrow{x_3}])$$

where $[\overrightarrow{x_2}; \overrightarrow{x_3}]$ denotes the concatenation of $\overrightarrow{x_2}$ and $\overrightarrow{x_3}$ and *tanh* is a non-linearity. Bias terms have been omitted for simplicity. Similarly, decoding is defined as

$$[\overrightarrow{x_1\prime}; \overrightarrow{y_1\prime}] = \tanh(W_d \overrightarrow{y_2})$$

The goal is to learn W_e and W_d such that the reconstruction error

$$\sum_{x \in \{x_1, x_2, x_3, y_1\}} \|x - x\prime\|^2$$

is minimised.

As with the work of Baroni and Zamparelli, training is supervised, but no data is required beyond word embeddings, which can be obtained using any of the algorithms in Section 2.1. Unlike Baroni and Zamparelli, the algorithm of Socher et al. is applicable to complete sentences of any syntactic structure. Note the same composition operation is used for all nodes in the parse tree instead of an adjective-specific linear map. The model is therefore less flexible[3] as it has two matrices available to store information about all complex pairwise interactions between words. On the other hand, it is easier to train because it has considerably fewer parameters to learn from data.

2.2.4 *Category-theoretical models*

Grefenstette and Sadrzadeh (2011) present a range of tensor-based algorithms for composing simple transitive sentences such as "person buys car" or "child likes fruit". All of these represent nouns as vectors, which is compatible with the rest of the models in this section. Similarly to the work of Baroni and Zamparelli (2010), words that take arguments are represented as higher-order tensors, and these representations are learnt from data. I consider four specific instantiations of the tensor framework — copy-object, copy-subject, Frobenius Addition and Frobenius Multiplication, which all represent verb phrases as vectors as opposed to matrices or tensors.

Copy-object composition is defined as

$$\overrightarrow{\text{Subj Verb Obj}} = \overrightarrow{Subj} \bigodot (\overline{Verb} \times \overrightarrow{Obj})$$

where $\overrightarrow{Subj}$ and $\overrightarrow{Obj}$ are the embeddings of the subject and object respectively, $\bigodot$ is pointwise multiplication and $\times$ is standard matrix multiplication. The composition process is agnostic to how word vectors are learnt. The verb representation $\overline{Verb_v}$ for a verb v is learnt as follows:

$$\overline{Verb_v} = \sum_{\langle s,v,o \rangle \in \mathcal{T}} \overrightarrow{s} \bigotimes \overrightarrow{o}$$

where $\mathcal{T}$ is the set of verb phrases (subject-verb-object tuples $\langle s\text{-}v\text{-}o \rangle$) for v which occur in a corpus, $\overrightarrow{s}$ and $\overrightarrow{o}$ are the unigram vectors corresponding to the subject s and object o and $\bigotimes$ is tensor multiplication (Milajevs et al., 2014, Section 3). For example, if the only uses

3 This is arguable, as Socher et al.'s model is non-linear and Baroni and Zamparelli's is linear.

of the verb "to buy" in a corpus are "person buy car" and "child buy toy", the representation of "buy" would be

$$\overline{buy} = \overrightarrow{person} \bigotimes \overrightarrow{car} + \overrightarrow{child} \bigotimes \overrightarrow{toy}$$

and the vector for the unobserved VP "person buy toy" would be

$$\overrightarrow{\text{person buy toy}} = \overrightarrow{person} \bigodot (\overline{buy} \times \overrightarrow{toy})$$

Copy-subject composition is defined as

$$\overrightarrow{\text{Subj Verb Obj}} = \overrightarrow{Obj} \bigodot (\overline{Verb}^T \times \overrightarrow{Subj})$$

Frobenius addition and multiplication are defined as the pointwise sum or product of copy-subject and copy-object respectively.

OTHER MODELS In this chapter I have only reviewed those general-purpose models that will be evaluated experimentally in Chapter 4. Several other compositional models specifically targeting document classification will be discussed in Chapter 3.6. Evaluating models by Thater et al. (2010); Giesbrecht (2010); Dinu and Lapata (2010); Paperno et al. (2014); Weeds et al. (2014b); Weir et al. (in press) is left to future work.

2.3 EVALUATING DISTRIBUTIONAL MODELS

This section reviews existing evaluation routines for distributional models, with an emphasis on compositional models. I group existing proposals into two broad classes, intrinsic and extrinsic, and review some of the fundamental properties of each. I will discuss the advantages and disadvantages of each evaluation in Chapter 3 and argue that intrinsic tasks are unreliable due to the subjective nature of the task and the small size of the gold-standard datasets used to date.

2.3.1 *Intrinsic evaluation*

Sparck-Jones and Galliers (1995, p 19) define intrinsic evaluation criteria as "those relating to a system's objective". I group existing intrinsic evaluations for distributional models into three categories, based on word similarity, context prediction and selectional preferences respectively. These are discussed separately below.

Word 1	Word 2	Sim
money	cash	91.5
tiger	cat	73.5
doctor	nurse	70.8
smart	stupid	58.1
five	month	33.8

Word 1	Word 2	Sim
car	automobile	100
cooking	rice	74
carrot	pepper	56
hawk	insect	42
dog	silver	18

Table 2.2: Examples of word pairs and their similarity from WS353 (left, Finkelstein et al., 2001) and MEN (right, Bruni et al., 2014), normalised to a scale from 1 to 100.

2.3.1.1 *Word and phrase similarity*

In the distributional literature, most intrinsic procedures attempt to assess the quality of a distributional model by correlating its output to human-provided word or phrase similarity annotations (Rubenstein and Goodenough, 1965; Miller and Charles, 1991; Finkelstein et al., 2001; Huang et al., 2012; Luong et al., 2013; Bruni et al., 2014; Hill et al., 2014). These evaluations are very direct in that they attempt to measure if models capture what humans perceive as the semantics of an utterance. Most datasets are at the word level, but some present ambiguous words that can be disambiguated by context — such tasks require some form of context representation, which is a central problem in composition. Such context-dependent evaluations are therefore commonly used to evaluate distributional models. Several examples of word similarity scores assigned by humans are shown in Table 2.2.

A family of data sets explicitly focus on the similarity of relations and ask questions such "London is to Britain as Paris is to what? (answer: France)". Some versions of the task offer a choice of candidate answers (Jurgens et al., 2012; Turney, 2012), while some are open-ended (Mikolov et al., 2013a). The data set of Mikolov et al. also contains a set of syntactic questions, where focus is on morphological rather than semantic relatedness. The questions are of the form "Kings is to king as houses it to what? (answer: house)".

At the phrase level, several evaluation data sets are based on the intuition that similar words or phrases are substitutable in context, which involves both finding synonyms and disambiguating the context. For instance, "bright" can be replaced by "talented" in the sentence "She is a bright student" and by "well-lit" in "The room is east-facing and is very bright in the morning" (McCarthy and Navigli, 2007). This task requires words with similar distributional representa-

tions to be limited to synonyms and hypernyms in order to preserve the truthfulness of the sentence. Also, unlike the word similarity data sets discussed above, performing well at this task requires a notion of context and therefore (likely) composition.

Mitchell and Lapata (2008) present a similar phrase-level intrinsic evaluation data set that also requires an understanding of context. The authors focus on intransitive verbs and their objects, and select examples where the meaning of a polysemous verb can be disambiguated by its object. For instance, in the context of "fire", "to glow" is closer in meaning to "burn" than it is to "beam". The opposite is the case in the sentence "The face glows". These requirements are quantified by asking humans to provide a similarity judgement between verbs in context. The similarity scores provided by a good compositional model are expected to be strongly correlated with the human judgements. Doing well at that task requires a grasp of composition and the ability to compute the similarity of short phrases. The data set was subsequently extended to transitive sentences (Grefenstette and Sadrzadeh, 2011).

Erk and Padó (2008) extract ambiguous target words and their context, and aim to rank a predefined set of possible paraphrases of the target word by appropriateness. For instance, in the sentence "By asking people who work there, I have since determined who he was." the best ranking of substitutes for "work" is "be employed" (4 votes), "labour" (1 vote), "toil" (0 votes) and "task" (0 votes).

Kartsaklis et al. (2012) and Polajnar and Clark (2014) build datasets of dictionary-like definitions such as "frequent = event happen many time short time" and "coach = teacher sport". The task is to compose those definitions in a way that makes the correct definition of a term more similar to the term than other definitions.

New word similarity data sets are still being published, with four clear trends:

SIZE early data sets, e.g. (Rubenstein and Goodenough, 1965; Miller and Charles, 1991) only contain several tens of pairs of words. More recent work, e.g. Hill et al. (2014) and Bruni et al. (2014), contain several thousand entries. The latter provide a more comprehensive test bed for distributional models.

CONTEXT In early work, human annotators saw single words out of context, which made their tasks particularly challenging. It was up to the subject to choose what sense of a word is intended.

For instance, the river sense of "bank" is similar[4] to "river", but the financial sense is not. More recent data sets, e.g. Luong et al. (2013), show annotators an entire sentence and highlight the target word, which provides some degree of disambiguation.

SIMILARITY VS RELATEDNESS Intrinsic data sets often contain a mix of semantic relations, such as synonymy ("car–automobile"), antonymy ("smart–stupid"), hypernymy ("cat–tiger"), co-hypernymy ("hawk–insect") and relatedness ("cooking–rice"), which are all annotated with a high similarity score. Hill et al. (2014) argue these qualitatively different relations should not be conflated. SIMLEX999 therefore contains only similar (and not related) words.

LONGER PHRASES Initial work was dominated by word-level data sets, e. g. (Rubenstein and Goodenough, 1965; Miller and Charles, 1991). More recent tasks are starting to consider noun phrases, verb phrases or entire sentences (McCarthy and Navigli, 2007; Mitchell and Lapata, 2008; Turney, 2012). While these data sets partly address the lack of context for the annotator, they are still narrow in scope and coverage — the data set of Mitchell and Lapata only contains 120 verb-object phrases. The issues with word-similarity data sets will be discussed in more detail in Chapter 3.1.

2.3.1.2 *Context prediction*

One theory as to what information phrase vectors should encode states that "the meaning of a string is a vector representing contexts in which that string occurs in a hypothetical infinite corpus" (Clarke, 2012). One should therefore be able to predict or infer those contexts from the vector. In other words, a compositional model should produce phrase representations which are consistent with the representations one would have obtained directly from an infinite corpus.

Baroni and Zamparelli (2010) instantiate this idea by requiring phrase representations derived through composition to be similar to corpus-

4 Under a particular interpretation of what "similar" means. The annotation guidelines for WS353 (Finkelstein et al., 2001) do not provide a definition at all and leave that decision to the annotator. The MEN guidelines (Bruni et al., 2014) also do not provide an explicit definition, but suggest that holonyms (e. g. "wheels-car") are more similar than topically related words (e. g. "dog-race"). In Chapter 3 I will argue that the tacit assumption that there exists a single notion of similarity, independent of what one is trying to achieve, is one of the main shortcomings of existing intrinsic data sets.

observed ones. For instance, the result of composing the vectors for "red" and "car", $\widehat{red\_car} = f(\overrightarrow{red}, \overrightarrow{car})$, and the observed vector for the phrase "red car", $\overrightarrow{red\_car}$, derived by treating all occurrences of "red car" in a corpus as a single token, should be very similar. This approach is attractive as it does not rely on information provided by human annotators. However, it does not reliably scale beyond bigrams because of the sparsity of observed n-grams. Also, it might incorrectly penalise over-predicted contexts, i. e. ones a compositional model (correctly) believes are plausible for a given word or phrase, but have never been observed in the corpus due to data sparsity.

2.3.1.3 *Selectional preferences*

Another approach to intrinsic evaluation is to verify that compositional models learn to respect the selectional preferences of their constituents. For instance, while the verb phrase "animal eats mountain" is grammatical, mountains do not possess the "can-be-eaten" semantic facet (Polajnar et al., 2014)[5]. The semantic representation of that verb phrase should be qualitatively different to that of "animal eats plant".

Similarly, Vecchi and Baroni (2011) use an intrinsic evaluation based on the notion of plausibility. The key idea is a phrase may not have occurred in an unlabelled corpus for two reasons. First, the phrase may be nonsensical, i. e. the head and argument may not match each other's selectional restrictions. An example of such a nonsensical phrase is "residential steak" (Vecchi and Baroni, 2011). Second, the phrase is not common enough or the corpus is too small. For example, the phrase "blue rose" may not appear in a corpus but is it perfectly plausible that a rose may be blue.

2.3.2 *Extrinsic evaluation*

Extrinsic evaluations incorporate distributional methods in a practical task and look for an improvement in performance at that task. Authors typically view distributional models as an extra source of information that improves their model, and the focus is by and large on performing better at the task. Often a single distributional model with default parameter values is utilised (e.g. Berant and Liang, 2014; Kiela and Bottou, 2014). There has been little work on comparing dif-

5 Perhaps with the exception of mountains of candy.

ferent distributional models in terms of their ability to improve performance at an external task.

Huang and Yates (2009) obtain considerably more accurate part-of-speech taggers and chunkers by linking infrequent words to frequent words through their distributional properties in a large unlabelled corpus. The authors present distributional models as a smoothing method that alleviates the sparsity problem faced by Conditional Random Fields (Lafferty et al., 2001) trained on little data. Results suggest that distributionally-augmented models require considerably less training data to reach the same levels of accuracy. However, no attempt is made to compare different ways of instantiating a distributional model.

Turian et al. (2010) improve their chunker/NER system through word-level distributional information. The authors experiment with a wide range of distributional models, but the differences are very small — reported F1 scores on the test set range from 94% to 95.15% for chunking and from 87.36% to 90.9% for NER (error bounds were not reported). With differences that small it is difficult to meaningfully compare the value of different distributional models or to gain qualitative insight into their properties. Also, all distributional models considered are word-level.

Weston et al. (2015) propose an extrinsic framework based on question answering and cast a broad range of high-level semantic problems within it. The tasks range from basic factoid question answering ("John is in the kitchen. Where is John?") to induction ("Lily is a swan. Lily is white. Greg is a swan. What colo[u]r is Greg?"). These tasks are not necessarily directly addressable by the distributional methods evaluated in this thesis, but they approach the problem of evaluation of general-purpose semantic models methodically by testing different aspects of a model within the same framework.

Milajevs et al. (2014) compare the effect of some of the parameters of distributional models on two large-scale tasks— dialogue act tagging and paraphrase identification. The former task is concerned with attaching a tag such as question or statement to each sentence in a conversation. The latter is about deciding if a pair of sentences with high lexical overlap are paraphrases or not.

Lastly, a number of authors have used composed phrase representations as input to another step in a pipeline. These models often are designed to use additional task-specific labelled data (unlike the models discussed in Chapter 2.2, which do not rely on labelled data).

Socher et al. (2013b) and Hermann and Blunsom (2013) evaluate their composers on the task of sentiment analysis. The models are trained in a similar fashion to Socher et al. (2011), but also make use of a dataset of phrases and sentences annotated for sentiment. A classifier is trained to predict the sentiment a phrase expresses given its distributional representation, which is learnt jointly with the parameters of the classifier. Socher et al. (2013a) present a model that jointly learns how to parse and to represent phrases as vectors. The common feature of those papers is that they learn a task-specific representation that is not necessarily a general-purpose semantic model, but often can be useful to some extent for other tasks.

2.4 DISCUSSION AND OPEN QUESTIONS

This section discusses several open questions in the distributional semantic literature. These include: learning of vectorial representations from linguistic resources other than free text; learning multiple context-dependent representations per word; using the same structures to model words, phrases and sentences; non-compositionality and extra-linguistic knowledge. While this thesis does not deal with these issues experimentally, they form an important part of the current discourse in the area and illustrate important directions for future research.

2.4.1 *Alternative representations*

DISTRIBUTED REPRESENTATIONS Erk (2012) differentiates between distributional and distributed representations. The former refer exclusively to models relying on the distributional hypothesis and representing words and phrases as a distribution over the contexts in which they may occur. In contrast, the latter is a more general term that encompasses a range of continuous representations that do not necessarily make use of the distributional hypothesis. One such work is Faruqui and Dyer (2015), where a word is seen as a distribution over components of discrete lexical resources such as WordNet (Miller, 1995) or FrameNet (Baker et al., 1998). The features of "adoration" might include the binary indicators "is_synonym_of_love" and "is_synonym_of_affair" (Faruqui and Dyer, 2015, Section 2), which are derived from WordNet synsets. Similarly, McRae et al. (2005) represent words as a distribution over human-interpretable properties

that relate to the real world. For instance, "airplane" has as features "has_wings", "is_fast" and "made_of_metal".

MULTIPLE REPRESENTATIONS Recent distributional models also break away from the traditional single-vector-per-word paradigm. For instance, Socher et al. (2012) and Paperno et al. (2014) model words as both a vector and a matrix. The vector captures the meaning of a word when used as an argument, and the matrix when it is used as a predicate. The use of matrices and matrix-vector product "formalize[s] argument slot saturation, operating on an argument vector representation through matrix by vector multiplication" (Paperno et al., 2014, Section 2.1). These richer representations allow for the modelling of the semantics of each word separately in each context and grammatical function it can be used in.

In a similar vein, Reddy et al. (2011), Huang et al. (2012), Bartunov et al. (2015) and Neelakantan et al. (2015) learn multiple vectors per word, each corresponding to a distinct word sense in a particular context. Baroni and Zamparelli (2010) assign a separate vector to each word depending on its part-of-speech tag. This is a computationally efficient way of achieving partial word sense disambiguation. For example, "to bank" and "a bank" would be represented separately, but the river and financial senses of "bank" when used as a noun are not differentiated. I also adopt this method in Chapter 4.

2.4.2 *Word, phrase and sentence space*

Within the family of distributional models, the company a word keeps is typically modelled as a vector. However, a number of other options have been proposed. Some authors argue that words that take arguments, such as adjectives and verbs, should be represented as higher-order tensors instead to better capture the complex interactions between functions and arguments (Baroni and Zamparelli, 2010; Socher et al., 2012; Grefenstette, 2013; Paperno et al., 2014). Others use trees (Erk and Padó, 2008; Weir et al., in press) or graphs (Hope, 2015).

Another open question is what the semantics of the sentence space should be. Clarke (2012) subscribes to a "context-theoretic" view, where the representation of a sentence is compatible with that of words and models its possible contexts in a hypothetical infinite corpus. Kiros et al. (2015) implicitly assume the same by building sentence vectors

that can be used to predict surrounding sentences. In contrast, Coecke et al. (2011, Section 4.1), Grefenstette et al. (2011, Section 2) and Clark (2013, Section 4) present a two-dimensional truth-theoretic (plausibility) space, which is spanned by TRUE and FALSE basis vectors. In that framework, sentences are not continuous entities but true or false (to a degree).

It is also unclear whether 1) phrases, sentences and words should live in the same vector space, and 2) words of different types (e. g. nouns, adjectives) should live in the same space.

Formal semanticists generally answer these question negatively. Different grammatical structures should not be comparable, because it is meaningless to compare the meanings of, for example, noun and verb phrases. Therefore, structures of different grammatical type should not live in the same vector space in a distributional framework. However, classes of comparable structures exist. In many grammatical formalisms structures are typed, and their types describe the kinds of interactions these structures can participate in. For example in category-theoretical grammar the type of a noun is N, and the type of an attributive adjective in English is N/N, i. e. an adjective is a function that takes a noun to the right and returns another noun-like object. Syntactic types explicitly define classes of comparable structures. For instance, composed noun phrases and nouns are both of type N and are therefore comparable. In distributional semantics, the closest counterpart is the work of Baroni and Zamparelli (2010) and Coecke et al. (2011), who model adjectives as matrices and nouns as vectors. As a result, these two atoms are not comparable, but their composition (a noun phrase represented as a vector) is comparable to nouns.

However, some authors view non-comparability of grammatically different structures as an undesirable side effect of using models where "predicate arity is encoded in the order of the corresponding tensor" (Paperno et al., 2014, Section 1.2). Later work by Baroni and Zamparelli's group takes a step towards addressing what they see as a limitation of their 2010 model by representing each word as both a vector and a set of matrices (Paperno et al., 2014). The motivation for this decision is that "the same or similar items that occur in different syntactic contexts are assigned different semantic types with incomparable representations" (Paperno et al., 2014, Section 1.2). The introduction of multiple representations for each word makes it pos-

sible to compare the meaning of a verb in transitive, intransitive and passive uses.

A large proportion of the neural network literature, e. g. Le and Mikolov (2014); Socher et al. (2011); Kusner et al. (2015) subscribes to the idea that all words, phrases and sentences should live in the same space. In practice, being able to compute the similarity between any pair of constituents, regardless of their grammatical structure, turns out to be beneficial. For instance, Socher et al. (2011) first compute a vector representation for each subtree of a constituency parse of a sentence and then use these vectors to compare every subtree from a target sentence to every subtree in a candidate paraphrase sentence. The model achieves state-of-the-art performance in paraphrase detection even though the word vectors it operates on can be improved (Section 4.4.3)[6].

It should be pointed out that while there are no explicit constraints on what types of words and phrases should be comparable, in practice neural distributional models often learn a soft and implicit notion of syntactic typing. For instance, Tables 4.16 and 4.17 show the nearest distributional neighbours of nouns are predominantly other nouns, and those of adjectives are mostly other adjectives, even though this constraint is not explicitly encoded into the objective function of the model.

This thesis argues for a compromise between the two extremes. On one hand, imposing a hard constraint on what grammatical types of words or phrases can be compared may result in loss of performance. Imagine a topic classification task where one is trying to differentiate between articles about war and articles about trade. Words and phrases that identify the first topic can be of various syntactic types, e. g. "a conflict", "to destroy", "destructive", "collateral damage", "to fire on civilians". Being able to compute a similarity score between these would undoubtedly be beneficial.

However, just because one can compute the similarity of any pair of phrases does not mean one should. In the war example above sim-

6 I hypothesise this is due to the pooling layer, which performs a pairwise comparison between all nodes in a sentence's parse tree. Such approaches are driven by a desire to do well at a task rather than by some underlying intuition about the nature of language. This is indicative of a more general trend in the neural embedding literature, where little to no linguistic preprocessing is done and data quantity is favoured over data quality. Papers in the area typically rely on having large amounts of unprocessed data as opposed to small(er) amounts of processed data. For example, entries and features are commonly stemmed or lemmatised in the counting-based representation literature to reduce data sparsity. In contrast, this is seldom done in the neural embedding literature.

ilarity scores are computed and make sense only in the context of a task. How similar is the noun "dog" to the adjective "purple" to the number "seven" outside of a particular context? The issue is it is unclear what steps one goes through to reach a conclusion. A possible explanation is that one thinks of a "purple dog" and the plausibility of that phrase. Another is that one considers the selectional preferences of both words and decides if they can be composed in any way. A third option is one considers the ontological relationship between words. For instance, "cat" is similar to "dog" as both are types of animal, and "racket" is similar to "squash" as the competitive sport is played with rackets. However, these intuitions do not hold when comparing phrases of different syntactic types or at the phrase level. It is unclear how one determines if "very" or "much faster" is similar to "dog".

2.4.3 *Non-compositional and extra-linguistic meaning*

In both formal and distributional settings, it is well-known that compositionality does not hold universally (Saeed, 2009). For example, in the idiom "spill the beans" the verb "spill" takes on a new sense of "reveal" and beans maps onto "meaning". "Given that these senses [...] are not readily available at the simplex level other than in the context of this particular [multi-word expression], it seems fallacious to talk about them composing together to form the semantics of the idiom." (Baldwin et al., 2003, p 1). The issue is complicated even further by the fact that phrases can be used both literally and metaphorically. While it is a less frequent use, "spill the beans" may literally refer to slipping and dropping a can of beans on the floor.

Often the distinction between compositional and non-compositional use is not clear cut. Phrases are typically thought of as lying in a "continuum of compositionality" (McCarthy et al., 2003). Baldwin et al. (2003) identify three degrees of non-compositionality based on the degree to which phrases admit syntactic variation. This view is so widely held that a shared task was held in 2011 at CoNLL, where phrases were annotated for degree of compositionality on a hundred-point scale. Some examples of the data for the shared task are shown in Table 2.3.

As Schubert (1986) points out, even though language is a system governed by a set of intrinsic rules, it is "by no means free from external influence". A language is used by humans, who are embod-

Phrase	DoC	Phrase	DoC
reinvent wheel	5%	small print	58%
raise bar	9%	left hand	76%
lose sight	19%	find way	91%

Table 2.3: Degree of compositionality (DoC) of several phrases from the 2011 CoNLL shared task (Biemann and Giesbrecht, 2011). The higher the score, the more literal the meaning of the phrase is.

ied agents and experience the world through a range of senses. It is natural that some of the language we use will refer to experiences related to phenomena outside of language. In other words, it is likely that a distributional model that defines meaning as use is unable to fully model aspects of semantics that relate to our other senses. The research community has recently started to recognise this and has attempted to bridge language and visual (Frome et al., 2013; Bruni et al., 2014; Kiela and Bottou, 2014; Kiros et al., 2014), olfactory (Kiela et al., 2015) and auditory (Kiela and Clark, 2015) experiences.

CHAPTER SUMMARY

This chapter introduced the background material which will be referred to throughout the rest of the thesis. I reviewed existing methods for building word and phrase representations. At the word level, I detailed two families of algorithms: counting-based, which count feature-entry co-occurrences, re-weight them and perform dimensionality reduction, and neural algorithms, which define an explicit cost function and learn vector representations that optimise it. I discussed four types of compositional algorithms, based on pointwise vector operations, linear regression, recursive neural networks and category theory respectively.

I surveyed a range of methods that have been used to evaluate distributional models. The most popular types of assessments are the so-called intrinsic ones, which typically correlate a DM's prediction to human-provided similarity scores. In contrast, extrinsic methods embed DMs in an external task and assess their utility at that task. Currently, intrinsic data sets are by far the more popular choice. In the next chapter, I will present theoretical and empirical arguments against intrinsic evaluation, and will introduce a novel extrinsic procedure.

3

A FRAMEWORK FOR EXTRINSIC EVALUATION

The previous chapter outlined current approaches to learning distributional word representations and composing them into phrase representations. I also discussed how distributional models (DMs) have traditionally been evaluated. This chapter will critique existing intrinsic evaluations and will propose a novel extrinsic evaluation.

Section 3.1 argues that existing intrinsic evaluations are unreliable as they make use of small gold-standard data sets and assume there exists a single notion of similarity that is independent of a particular application. I consider four commonly used word similarity datasets and show empirically that inter-annotator agreement in these is considerably lower than in extrinsic tasks. Further, three of the four datasets fail to reliably detect the presence of random noise in word vectors.

Section 3.3 proposes a novel extrinsic framework based on distributional feature expansion applied to document classification (DC). In that framework, a document classifier is trained with a bag-of-phrases representation. At test time, all document features are replaced with their nearest neighbour according to a distributional model. Intuitively, a classifier is not allowed to access the actual contents of a test document, but can only view it through the prism of a DM. DMs that better capture the meaning of words and phrases in the context of a particular classification dataset provide better replacements. The accuracy of the classifier when a particular DM is used provides a direct measure of the DM's quality. Chapters 3.4 and 3.5 show a detailed example and discuss some aspects of the procedure that require careful consideration.

Section 3.6 discusses alternative ways of utilising distributional models for document classification. These present competing ways of instantiating an evaluation through document classification.

Chapter 4 will present an evaluation of the methods for building word representations and composing them introduced in Chapter 2 in the framework described below.

3.1 ISSUES WITH INTRINSIC EVALUATION

3.1.1 *Size of data set*

One of the key issues with current intrinsic evaluation data sets is their small size. This is a by-product of how laborious such resources are to curate manually. Current data sets contain from 30 to 3000 word pairs (Table 3.1). Moreover, they only feature a "tidy" subset of all naturally occurring words and phrases, free of spelling variation, domain-specific terminology and named entities. The focus is predominantly on relatively high-frequency words.

Data set	Size	Unit
Miller and Charles (1991)	30	word pairs
Rubenstein and Goodenough (1965)	65	word pairs
Landauer and Dumais (1997)	80	multiple-choice questions
Finkelstein et al. (2001)	353	word pairs
Hill et al. (2014)	999	word pairs
Huang et al. (2012)	2003	word pairs
Luong et al. (2013)	2034	word pairs
Bruni et al. (2014)	3000	word pairs

Table 3.1: Sizes of popular word similarity data sets.

In contrast, typical distributional models "in the wild" have entries for all words in an unlabelled corpus (possibly with an entry frequency filter). The resulting vocabulary can be in the order of tens of thousands of words. For instance, the 2010 copy of English Wikipedia used in the next chapter contains $69K$ words that occur more than 100 times. In such a large distributional model, it is not at all difficult to find an entry whose neighbours appear to be unreasonable. Table 3.2 shows the top neighbours of several randomly selected unigram entries in a WORD2VEC model. Note that some of the entries are common English words ("performer"), some are less common ("pawnbroker", "godmother"), and some appear to be proper names (e. g. "amal", "menuhin", "roden").

Intrinsic evaluation data sets do not always feature rare entries, so the extent to which the model is sensible cannot be quantified fully. For practical applications, users need to understand the entire DM, not just the small fraction of it covered by an intrinsic evaluation. An analogy to software engineering describes the issue well. Good

quality unit tests have high coverage, i. e. they exercise all possible paths through a program. Poor tests only cover the basic use cases and fail to reveal corner cases which inevitably occur in practice.

Entry	Neighbour 1	Neighbour 2	Neighbour 3
godmother	betrothed	lover	spinster
performer	musician	artist	entertainer
confiscation	expropriation	forfeiture	taking
amal	al-amin	mukhtar	nasrallah
alms	penitent	passer-by	sainthood
pawnbroker	thrifty	bookie	yusuke
plas	cupar	vanden	bodoni
menuhin	yehudi	caryl	barenboim
parent	mother	family	live-in
roden	coy	kempthorne	orme

Table 3.2: Randomly selected entries from a distributional model and their nearest neighbours.

3.1.2 *Definition of Similarity*

The notion of similarity is challenging to define precisely. The literature typically takes a what I call a "causal" definition, where reasoning is from the world towards language. Words are seen as similar because the concepts they refer to have a relationship in the real world. For instance, "car" and "automobile" are similar as they often denote the same physical object; "car" and "wheel" are similar because real-world cars have wheels. In that frame of mind, words are inherently similar or dissimilar; they can be deemed similar because they are synonyms, antonyms, hypernyms, hyponyms, co-hyponyms, meronyms, holonyms or topically related. However, with the exception of SimLex-999 (Hill et al., 2014), none of the data sets discussed in this thesis attempt to differentiate between similarity and relatedness, or between different types of relatedness. In other words, intrinsic evaluations typically assume that there exists a single gold-standard similarity score between a pair of words or phrases. That score does not usually reflect the reason two words are similar. As we saw in Table 2.2, intrinsic data sets contain a mix of semantic relations such as synonymy ("car–automobile"), antonymy ("smart–stupid"), hypernymy ("cat–tiger"), co-hypernymy ("hawk–insect") and relatedness

("cooking–rice"). While there has been some work attempting to differentiate between those different types (e.g. Weeds et al., 2014a), the problem is still largely unsolved.

In this thesis, I argue for an operational definition of similarity, conditioned on and suited to a particular task. For example, "good acting" and "cluttered set" are highly dissimilar in terms of the sentiment they express towards a theatrical play. However, they are very similar in the context of detecting, in a multi-topic article stream, news items related to the theatre, as both phrases are highly indicative of theatre-related content. Lexemes are not inherently similar or dissimilar, but only in the context of a particular task[1]. This argument parallels that of Von Luxburg et al. (2012), who argue that "[d]epending on the use to which a clustering is to be put, the same clustering can either be helpful or useless". The quality of an unsupervised algorithm can only be assessed with reference to a particular application. Optimising a DM's correlation with a particular gold standard should therefore be viewed as only one of many possible tasks. For evaluation purposes, we require tasks which are not preloaded with assumptions about what is and is not correct.

3.1.3 *Relation between intrinsic performance and practical utility*

To date, distributional algorithms have mostly been evaluated intrinsically by measuring the degree to which they agree with human-provided word or phrase similarity scores. Such evaluations, however, do not necessarily measure how well a DM captures the kind of information needed for practical applications. Socher et al. (2012, Section 2.7) argue that "even with good correlation the question remains how these models would perform on downstream NLP tasks".

Even if one could reliably differentiate between the different types of similarity conflated by the word similarity task, it is still hard to come up with anything but basic intuitions as to how that information can be effectively used in practice. For instance, a model of hypernymy would probably be useful in textual entailment as "The cat ran across the road" implies "The animal ran across the road", but not vice versa (Weeds et al., 2014a). Being able to differentiate between synonyms and antonyms would probably be useful in information

1 The term context can be understood more generally than just language processing tasks. It can include social context, the background knowledge a person possesses or their sensory experiences. Humans often take context for granted because we use language exclusively in one of those types of context.

retrieval. A user searching for "white paint" would probably not be interested in results for "green paint" or "white paintbrush". However, these are only heuristics. In fact, part of the appeal of end-to-end neural distributional models is that they learn what is needed to solve a task without explicitly being told how to do so.

In contrast, in a framework where similarity is defined functionally (with respect to utility for a particular task), one does not need design a hierarchy of relatedness types. The definition of similarity is determined by the application domain and can be guided by labelled data, if available.

3.1.4 *Subjectivity and task difficultly*

When human judges annotate word pairs for similarity, the distinctions in meaning they are asked to make are often very subtle, especially in the absence of context. For instance, the similarity scores provided by 13 annotators for the pair "tiger–cat" range from 50% to 90% in WS353 (Finkelstein et al., 2001). This results in low inter-annotator agreement even between native speakers. In this section I compare the agreement score for the first 13 annotators of WS353 and the two authors of MEN (Bruni et al., 2014) to typical agreements reported in the document classification literature. Results suggest the word similarity task is considerably more challenging than document labelling tasks.

Comparing inter-annotator agreement scores for word similarity and document labelling tasks directly is not possible. Labels in the former are on an ordinal scale, so agreement is measured using Spearman's rho (ρ). In contrast, the labels used in the latter task are categorical; agreement is typically measured using Kohen's kappa (κ). To circumvent this issue I convert word similarity scores to discrete labels by placing the continuous scores into equally sized bins. For example, the range of similarity scores in WS353 is $[0, 10]$, and the bin boundaries are at $[0, 5, 10]$ when using two bins and at $[0, 3.33, 6.66, 10]$ when using three bins. The three-item continuous labelling $[2.1, 5.8, 7.9]$ is converted to $[A, B, B]$ when using two bins and to $[A, B, C]$ when using three bins.

This conversion process suffers from two drawbacks. First, order information is lost, so misplacing an item in bin A instead of in bin B is considered as severe an error as misplacing an item from bin A into bin F. This is less of an issue when the bin count is small.

Second, the number of bins is a free parameter ranging between 1 (all items in the same bin) and 7 or 10 (all items in original bins)[2]. κ is a decreasing function of the number of bins because it becomes harder for annotators to agree when there are a large number of bins to choose from. In this analysis I remain agnostic as to how many bins should be used and experiment with values between 2 and 5.

The inter-annotator agreement of WS353 and MEN (converted to Kohen's κ) is shown in Figure 3.1. Because κ is only applicable when there are exactly two annotators, I report an average κ over all pairwise comparisons[3]. A κ score can be computed between each of the 91 pairs of judges, or between each judge and the mean across all judges (as in Hill et al. (2014, Section 2.3)). These values are referred to as "pairwise" and "to mean" respectively[4]. Mean agreement ranges from $\kappa = 0.21$ to $\kappa = 0.62$[5].

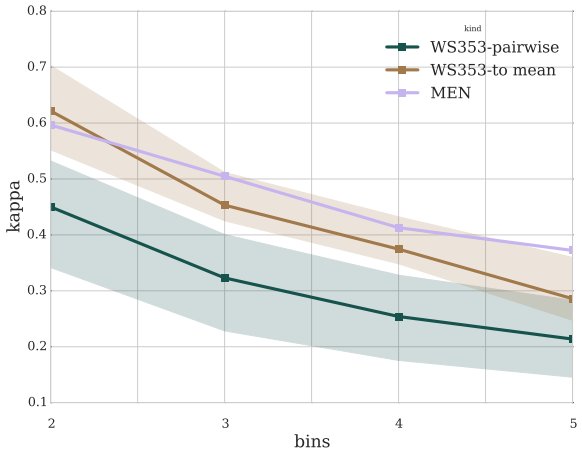


Figure 3.1: Inter-annotator agreement of WS353, measured in Kohen's κ. Shaded area shows the 68-th percentile. A confidence interval is not shown for MEN as only the annotation of a single pair of raters are available.

For comparison, Kim and Hovy (2004) report $\kappa = 0.91$ for a sentence-level binary sentiment annotation task. Gamon et al. (2005) report a pairwise κ ranging from 0.7 to 0.8 for a three-way sentence-level sen-

2 WS353 was annotated on a 10-point scale, whereas MEN used a 7-point scale.

3 Averaging is only needed for WS353, which has been annotated by (at least) 13 judges. MEN only provides full annotations for two judges.

4 Computing agreement scores between a rater R and the mean of all annotators is slightly disingenuous as the mean includes the annotations of R, so there is an element of measuring agreement with oneself. This artificially inflates the agreement score.

5 As a sanity check, I also compute inter-annotator agreement measured using Fisher-corrected Spearman correlation. For WS353 agreement is $\rho = 0.58$ for the "pairwise" case and $\rho = 0.74$ for "to mean", which is in line with the numbers reported by Hill et al. (2014). For MEN that figure is $\rho = 0.68$

timent task. McCormick et al. (2008) report $\kappa = 0.84$ for a five-way classification, where the task is to guess a patient's smoking status (e.g. smoker, past smoker, etc) based on a discharge summary written by their physician. Wilson et al. (2005) report $\kappa = 0.72$ for a four-class task where short expressions are annotated for sentiment. These are up to a few words long and are shown in the context of a sentence. Agreement rose to $\kappa = 0.84$ when phrases that annotators marked as "unsure" were removed from the data set. All these κ scores are considerably higher than those achieved by WS353 and MEN. These results challenge the value of intrinsic data sets as a gold standard.

It is impossible to calculate agreement for a number of other classification data sets, e. g. those where users volunteer a product review and therefore each review is only annotated by a single person. One such example is the popular data set of Pang et al. (2002).

3.1.5 *Sensitivity to Noise*

A desirable property of any evaluation framework for word vectors is that randomly generated word vectors result in random performance. In this section I investigate if this is indeed the case for four word similarity data sets— RG (Rubenstein and Goodenough, 1965), MC (Miller and Charles, 1991), WS353 and MEN. A random model is expected to achieve a correlation of $\rho = 0$ with the human-provided intrinsic similarity scores.

A WORD2VEC model is trained on either of two equally large unlabelled corpora— all of GIGAWORD or 15% of WIKIPEDIA (Section 4.1). Uniform random noise $\mathcal{U}(-n, n)$ is added to all non-zero elements of all word vectors, where n ranges from 0 to 3. It should be noted the vectors used in this section are dense, so most elements are non-zero.

Performance at the task is measured in terms of Pearson ρ to the gold standard similarities. A distributional model may contain no vector for some of the entries in an intrinsic data set. Low type coverage is typical when small amounts of unlabelled data are used for vector training. To account for this, I consider two different versions of this assessment:

RELAXED out-of-vocabulary (OOV) words are ignored. This setting may provide an unfair advantage to models trained on less data, because their poor coverage is forgiven.

STRICT OOV entries are assigned a random similarity score.

The relaxed and strict evaluations only differ considerably when very small amounts of unlabelled data are used (< 10% of WIKIPEDIA); this is due to coverage issues. Coverage of the intrinsic evaluation corpora ranges from 50% for RG and 85% for WS353 at 1% of WIKIPEDIA and increases rapidly to almost 100% at 15% of WIKIPEDIA for all intrinsic datasets.

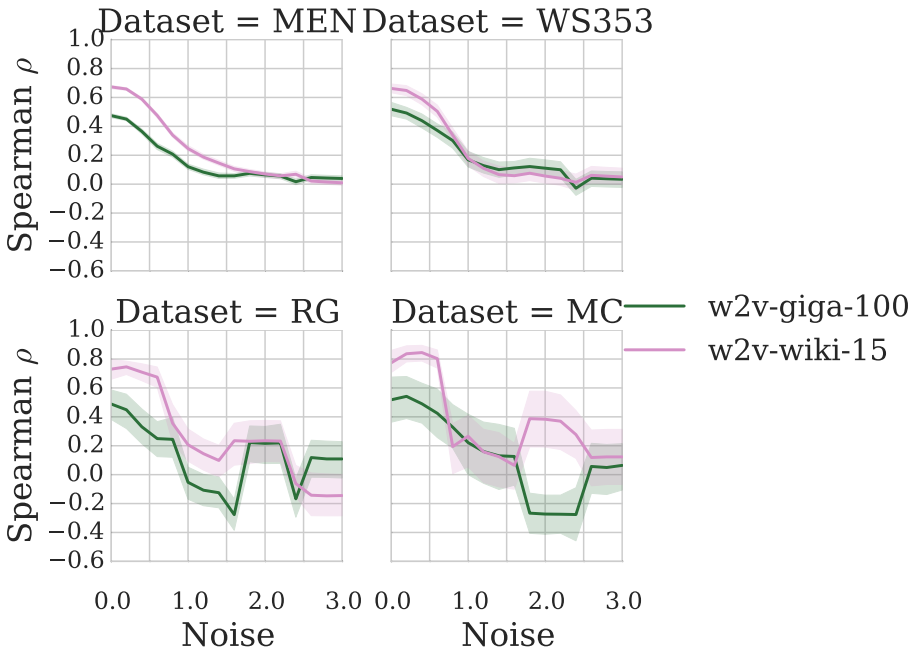


Figure 3.2: Noise test with intrinsic evaluations (relaxed version). Shaded region shows the 68-th percent interval centered at the mean, estimated via bootstrapping (Chapter 4.1.4). Results for the strict version are similar.

Results for the four data sets are shown in Figure 3.2. The two smaller datasets, MC and RG, do not sufficiently capture the degradation of vector quality as noise is added because ρ does not decrease monotonically with n. The variance of the measurements is also very high. WS353 and MEN's performance is satisfactory, with tighter error bars and correlation tending to zero as noise is added.

However, WS353 also exhibits undesirable behaviour upon closer inspection. Figure 3.3 shows the mean p-value of Spearman's ρ across all bootstrap repetitions. P-values for both unlabelled data sets and all labelled data sets increase rapidly after noise exceeds 0.8, meaning that the correlation scores in Figure 3.2 may have been observed by chance. The only labelled data set that performs reasonably is MEN. These results challenge the statistical power of intrinsic data sets.

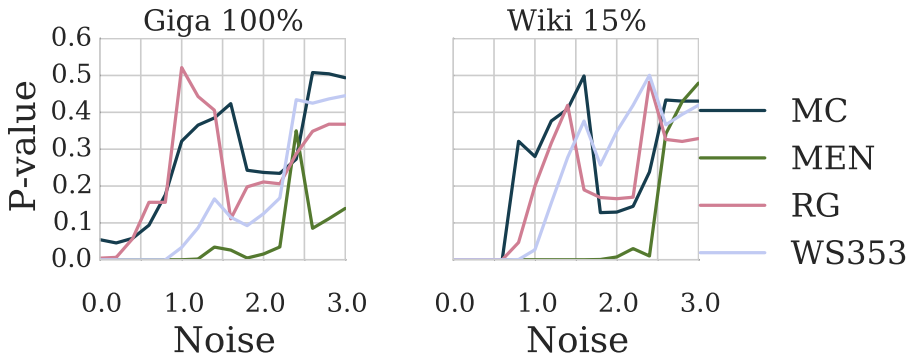


Figure 3.3: Mean p-values across all bootstrap repetitions for Spearman's correlations in Figure 3.2.

3.1.5.1 *Outlook*

As we have seen above, intrinsic evaluations suffer from a range of problems. Word similarity data sets assume the existence of a single gold-standard similarity score between words, detached from a particular application. Performance at such intrinsic evaluations does not necessarily correlate with practical utility. All four data sets above are small in size. As a result, MC, RG and WS353 fail to reliably detect the presence of random noise in word vectors. More fundamentally, the word similarity task is hard for human annotators to do reliably, which casts doubt on the suitability of such data sets for the evaluation of distributional models.

Despite this, intrinsic data sets are still the predominant paradigm in the literature. There are few recent comparative evaluations of distributional compositional models that use extrinsic rather than intrinsic evaluations. This appears to be mostly for historical reasons, as the end goal of a considerable amount of work in the 1990s was in fact to learn thesaurus information (Lin, 1998). Intrinsic evaluations were the first to emerge in the early days of distributional models and have effectively become the norm[6]. The use of extrinsic evaluation has been on the rise recently. This has been partly driven by companies like Google and Facebook setting up research programs in distributional semantics. The interest of those groups lies in doing well at business-critical tasks. Distributional semantics therefore became an important commercial tool rather than an academic undertaking, and as a result evaluation has shifted towards task-based

6 In more than one talk I have attended, prominent researchers have verbally expressed the opinion that one is virtually required to include an intrinsic evaluation of their method to get a paper accepted.

approaches. However, this trend does not seem to affect intrinsic evaluation methods, which are still extremely common. To the best of my knowledge, this thesis is the first work to openly criticise intrinsic evaluation and present empirical evidence of its disadvantages.

3.2 INTRODUCTION TO DOCUMENT CLASSIFICATION

To address the limitations of intrinsic evaluations, I set out to design an extrinsic evaluation framework that meets the following desiderata:

1. It can be easily adapted to a variety of different notions of similarity.
2. It is agnostic to the method used to build word and phrase representations.
3. It is sensitive to subtle differences in the quality of DMs; a random DM results in random performance.

The main contribution of this thesis is an instantiation of such a framework through document classification (DC). Note that improving on the state-of-the-art in DC is not an objective of this work. Instead, the aim is to compare different methods of building phrase representations and their suitability for use in DC. To test my proposal I focus exclusively on models for composing noun-noun and adjective-noun compounds (henceforth collectively referred to as noun phrases or NPs), and subject-verb-object compounds (verb phrases or VPs).

The rest of this section introduces the problem of document classification and outlines why the task is suitable for evaluating distributional models. I will only provide an overview of the properties that make DC appealing for distributional models. A thorough review of the DC literature can be found in Sebastiani (2002).

Document classification is the task of assigning text documents to a set of predefined categories (also called classes or labels). The term "document" is a generalisation and may be used to refer to anything from short phrases to web pages containing thousands of words. Typical modern classification systems are statistical and supervised — they make use of a *labelled* collection of documents to learn patterns that correlate with a particular label. Some of the factors that make DC a challenging problem in general are:

SHORTAGE OF TRAINING DATA Many real-world DC data sets are not sufficiently large for a classifier to reliably learn to differentiate between the classes.

CONTENT DRIFT The language used to describe a category may change over time. The performance of classifiers often degrades with time in online-learning scenarios, often within the space of several days.

CLASS IMBALANCE There can be many classes in a data set, some of which may be severely under-represented. A classifier that performs well on the more frequent classes may fail at the less common ones.

MULTI-LABEL CONTENT In some classification corpora a document can belong to multiple classes. This may be because the classes overlap semantically or because the document is large and refers to multiple distinct topics.

Document classification possesses several key properties that make it a good test bed for distributional semantics. These are:

VARIETY DC data sets come in a range of shapes and sizes. Different kinds of knowledge are required to do well in each. The classes a system is expected to distinguish can be defined by variation in topic (Hersh et al., 1994; Lewis et al., 2004), sentiment (Pang et al., 2002), emotion (Wicentowski and Sydes, 2012), presence of irony or sarcasm (González-Ibáñez et al., 2011), relevance to a business (Lyra et al., 2013), etc. The semantic distinctions between the classes in a labelled dataset define a particular notion of similarity that is needed to classify that corpus well. Therefore, a DM can be tested against multiple types of similarity by using a different classification dataset.

BACKGROUND KNOWLEDGE DC often requires knowledge that is not necessarily contained in the training set. Suppose one is interested in recognising documents relevant to a particular topic, such as "University of Sussex's performance at the 2015 REF". A typical DC system requires a large amount of relevant documents in order to correctly identify words and phrases that distinguish relevant from irrelevant documents. In contrast, a human would be able to achieve better performance using only a brief description of what "University of Sussex" and "REF" are. Humans are able to transfer their existing knowledge of the world and the English language to the task at

hand instead of learning from scratch each time. This is interesting because DMs have recently shown promising performance in encoding information about the world, such as ontological relations (Weeds et al., 2014a), physical properties of objects (Fagarasan et al., 2015) or factual knowledge (Gupta et al., 2015).

SIZE Unlike intrinsic data sets, it is possible to collect large and coherent labelled document corpora with relatively little effort. Annotating a document for topic or sentiment does not require special linguistic training as humans can naturally identify such differences. Often, explicit annotation is not required at all. For example, the AMAZON corpus (Section 4.1.1) was built automatically by collecting customer reviews that pertain to a particular product category. The labelled document corpus is a by-product of a separate annotation effort (creating a product ontology and matching products to categories in the ontology) and is essentially free for DC purposes. In other cases users effectively volunteer the annotation effort by providing a numerical rating for a product alongside their review or by posting a comment in a sub-category of a web forum. For instance, the TECHTC corpora (Davidov et al., 2004) were built by sampling messages from different sub-forums in a hierarchically organised message board. In contrast, building even a small resource such as MEN required a concentrated effort. Large expertly annotated resources like the Penn Treebank (Marcus et al., 1993) took tens of person-years of work by trained linguists.

There are, of course, some drawbacks to automated data collection procedures. The category labels assigned may be inconsistent or noisy. Web forum communities are often not heavily moderated and discussion of secondary issues appear frequently. For instance, users of car forums sometimes discuss audio equipment or their music preferences, which are not strictly related to cars. Also, topic boundaries are sometimes difficult to define precisely, resulting in some disagreement between annotations. Document-level annotation is not always the most appropriate level of analysis. A film review may express a positive sentiment towards one actor's performance and a negative sentiment towards the musical score. In that domain a phrase or sentence treatment may be more appropriate. This has been reflected in recent work (Socher et al., 2013b) where fragments of sentences are annotated instead of entire documents. This process still does not

require expert annotators, but is significantly more laborious than scraping web pages automatically.

3.3 PROPOSED METHOD

Traditionally, generative classifiers such as Naïve Bayes classify documents based on the prior probability of the class and what has been learned about the documents' features[7] from the labelled data. Features are surface forms, and the link between test-time and train-time features is based purely on these surface forms and not on their semantics. In order to provide a way of evaluating DMs, I break that direct link by *making knowledge of learned features accessible only through a DM*.

In the context of this work a DM is expressed as a thesaurus, which contains a list of the nearest distributional neighbours of each entry. Word vectors are extracted from a large unlabelled corpus and subsequently composed into phrase vectors. The thesaurus only contains entries for NPs and VPs found in the testing section of the labelled corpus (targets). Two constraints are placed on what can be a neighbour of a target. First, neighbours are limited to those NPs, VPs, adjectives, nouns and verbs that appear in the training section of the labelled corpus. Second, neighbours may not overlap lexically with the target, ensuring that it is not possible for a DM to succeed merely by operating on surface forms. A thesaurus constructed in this manner provides a mapping from a feature seen at test time (the target NP or VP) to features that the classifier knows about (its neighbours).

Document classifiers are trained as usual using a bag-of-features representation. At test time, however, every document feature is replaced with its k nearest neighbours from the thesaurus, where the value of k is a free parameter. An example is provided in Section 3.4. Classifiers are not allowed to access the actual contents of a test document, but can only view it through the prism of a DM. Neighbours thus act as proxies for the document feature they replace. DMs that better capture the meaning of words and phrases in the context of a particular classification dataset provide better replacements for target

7 In the DC literature, the features of a document are the words and phrases that occur in it. A document is traditionally viewed as a distribution over these ("bag-of-words" representation). Document features can in turn have a distributional representation, as described in Chapter 2. The words and phrases a document feature co-occurs with in an unlabelled corpus are its distributional features. Where there is no ambiguity I will use the term "feature" to refer to document features.

NPs or VPs. The accuracy of the classifier when a particular DM is used provides a direct measure of the DM's quality. Note the objective is *not* to improve on the state-of-the-art in classification. Instead, the task is used to measure the quality of the DM, focussing here on comparing different methods of compositionally building NP and VP representations.

In extracting the set of features contained in a document, I do not perform any feature reweighting (e.g. TF-IDF) or selection (e.g. top-k with Mutual Information scoring) on document vectors. However, I discard all document features that do not have a distributional representation. Such features can neither be replaced nor be replacements for other features and therefore do not affect the performance of the classifier.

This evaluation framework allows a DM to be tested against multiple types of similarity by using a different classification dataset. This is because the semantic distinctions between classes in a labelled dataset define a particular notion of similarity that is needed to perform well at that corpus.

The framework is also agnostic to how word and phrase representations are constructed or what these representations are, as long as the nearest neighbours of an entry can be computed. For practical reasons the rest of this thesis considers only DMs that model both words and phrases as vectors, because their similarity can be calculated efficiently using a range of well-studied measures (Weeds, 2003). However, the evaluation framework does not impose any constraints on how document features are represented. For instance, models where adjectives are modelled as matrices can be evaluated within the same framework by adapting the similarity measure. This can be done in a number of ways, the simplest of which is to linearise matrices to vectors (Baroni and Zamparelli, 2010). Similarly, models where phrases are stored as trees can be evaluated by converting trees to vectors (Weeds et al., 2014a).

Further, additional constraints can be trivially incorporated into the framework. For example, Chapter 2.4.2 discussed whether the meaning of words and phrases of different grammatical type should be comparable. The models evaluated in this thesis theoretically allow one to compute a similarity score between verb phrases and noun phrases (although in practice these may have very different vectors and are therefore seldom each other's top neighbours). The framework allows for explicit syntactic typing to be added in by limiting

the potential neighbours of an entry to only those words and phrases that match its grammatical type. The framework is agnostic as to what grammatical formalism is used to assign types.

Lastly, the framework is can be transparent and interpretable. While in theory the framework can be based on any classification algorithm, in this thesis I use a Multinomial a Naïve Bayes classifier (Metsis et al., 2006). It is easy to reason why this classifier has made a particular decision. Its class-conditional probabilities can be traced back to document features and their replacements. There is only a single "layer" of interaction inserted between a DM and how it affects a classifier (the replacements).

3.4 EXAMPLE

Suppose a Naïve Bayes (NB) classifier is trained to distinguish between articles about seismology and articles about zoology, and no document can be about both topics. For simplicity, assume these are the only two labels that the classifier will ever encounter (i. e. it will never be shown sports articles), and the topics do not change over time. Further, suppose the top two documents in Table 3.3 have been provided for training and the bottom two for testing.

Label	Document
Seismology	Vesuvius's violent eruption sends hot lava into the sky.
Zoology	Cat food is for small felines.
?	Reports of a violent eruption spread around the world.

Table 3.3: An example training (top) and testing (bottom) classification corpus

Documents are represented as a bag of words with adjectives, nouns and NPs used as features at train time and NP features only at test time. Table 3.4 shows the features extracted from the document collection in Table 3.3.

During training, the classifier will learn a high value for the conditional probability of `violent_eruption` occurring in a document whose class is "Seismology", because `violent_eruption` is a distinct and informative document feature that is unlikely to appear often in documents related to zoology. What would the decision of the clas-

Label	Features
Seismology	{vesuvius, eruption, violent_eruption, lava, hot_lava}
Zoology	{cat, food, cat_food, feline, small_feline}
?	{violent_eruption}

Table 3.4: Document features extracted from corpus in Table 3.3. Note only NP features are extracted from the test corpus.

Neighbour 1	Neighbour 2	Neighbour 3
volcano	hot lava	food

Table 3.5: Thesaurus entry for "violent eruption"

sifier be when it is presented with the last document in Table 3.3? In standard document classification decoding, violent_eruption is an exact match for a known training feature and the document is assigned the "Seismology" label.

The classification decision can be seen in two ways. In a sense, it is logical to classify the new document as being about volcanoes because it contains a feature highly indicative of seismology. On the other hand, no knowledge of the *meaning* of violent_eruption is needed, only that it often occurs in volcano documents. As far as the classifier is concerned, violent_eruption could as well have read "wampimuk" — the meaning of a feature is unimportant as long as it is correlated with one of the classes in the labelled corpus[8]. Of course, this is not to say a DM truly "knows" the meaning of words, but rather than a distributed representation can to some extent capture that meaning.

In my framework, the distributional representation of violent_eruption is used to build a thesaurus of words and phrases that are similar to it (Table 3.5). This contains all train-time features sorted by decreasing similarity. Using that thesaurus, the violent_eruption is replaced by its nearest neighbour volcano. The classifier is effectively told that observing violent_eruption in a document is equivalent to encountering volcano. Because of the small size of the training data the third neighbour is poor.

The classifier is able to make a correct decision because the underlying thesaurus happens to contain a sensible replacement

8 Under this interpretation all knowledge the classifier relies must come from the labelled data, which can be limited in size.

for `violent_eruption` in the context of the particular classification corpus. It would not have done as well if the top neighbour of `violent_eruption` were `feline`. The observation that classifier performance is sensitive to the quality of the provided thesaurus forms the basis of my proposal.

3.5 OTHER CONSIDERATIONS

Several aspects of the framework require careful consideration. Firstly, the main goal of this work is to provide a consistent framework of evaluation that puts all competing methods on an equal footing. However, the composers reviewed in Chapter 2.2 differ in the kinds of phrases they can handle. For instance, the work of Baroni and Zamparelli (2010) is only applicable to adjective-noun compounds, whereas the additive model can be applied to any phrase regardless of grammar. To ensure a fair comparison between all composers, I focus on their smallest common denominator. I experiment with two grammatical types of features. Noun phrases (NPs) consist of adjective-noun (AN) and noun-noun (NN) compounds, e. g. "large cat" or "cat food". Transitive sentences consist of transitive verbs and their subject and object, e. g. "dog chases man" (Section 4.1.3). Recall that here I refer to these as verb phrases (VPs) or subject-verb-objects (SVOs). However, the framework is applicable to document features of any grammatical type.

I build a thesaurus that contains an entry for each NP and VP in a labelled classification corpus. Vector representations for these are extracted from a large unlabelled corpus. The neighbours of an NP entry are all adjectives, nouns and other NPs in the labelled corpus, ranked by similarity to the entry as determined by a particular model of composition. Similarly, the neighbours of a VP entry are nouns, adjectives, verbs and other VPs. The thesaurus does not contain any entries for nouns, adjectives or verbs. As a result, a NP or VP can be replaced by a unigram at test time, but not vice versa. The purpose of this is to focus on the quality of *composition* as opposed to the quality of the unigram representations used, which has been studied extensively (albeit largely using intrinsic evaluations) in the past. However, the framework can be used to evaluate both word-level and phrase-level distributional models.

Secondly, I do not allow an entry and its neighbours to have any unigrams in common. For instance, "black cat"' cannot be a neigh-

bour of either "small cat" or "black trousers". This is referred to as the "lexical overlap constraint". The aim is to penalise pseudo-composers that do not actually make use of distributional representations but merely operate on surface forms, which would otherwise do very well in my evaluation. An algorithm that generates neighbours for an NP entry by randomly choosing other NPs that share the same head would score well in a topical DC task as the head of an NP is a strong indicator of its topic. Consider the task from Chapter 3.4. Any cat would be a good replacement for "black cat" in the context of telling animal-related documents from volcano-related ones. The lexical overlap constraint discourages such strategies. As a side effect, classification accuracy may suffer. However, the objective of this work is not to improve on the state-of-the-art in document classification but rather to provide a platform that allows researchers to compare different methods of building NP and VP representations and their suitability for document classification.

Replacing neighbours at test time requires one to compute the similarity between a target entry and a set of candidate neighbours. If the candidates are all possible phrases in a natural language the computational cost of the method would be prohibitive. However, the nature of my proposal permits certain optimisations which make my algorithm feasible in practice. I only search for neighbours of the document features that occur in the *testing* section of the *labelled* corpus. Also, the pool of candidates that are ranked according to their similarity to a target document feature only consists of features contained in the *training* section of the *labelled* corpus. The results of any nearest neighbour query are also cached to speed up retrieval.

Nearest neighbour search can be done in $O(dN \log N)$ using a KD-tree, where d is the dimensionality of the vector representation and N is the size of the candidate pool (Friedman et al., 1977). The use of KD-trees is essential to being able to run the large number of experiments in Chapter 4. However, they require a metric which satisfies the triangle inequality. Widely-used measures of similarity such as cosine distance, which have been shown to be appropriate for measuring semantic similarity (Lapesa and Evert, 2014; Bullinaria and Levy, 2007), are therefore not applicable here. Instead, I use Euclidean distance throughout this thesis.

3.6 RELATED WORK

The previous sections introduced a new framework for extrinsic evaluation of distributional models of semantics based on feature expansion in document classification. Several other extrinsic methods, focusing on tasks such as part-of-speech tagging, chunking and named entity recognition, were reviewed in Chapter 2.3.2. In this section I will discuss extrinsic evaluations specifically looking at document classification.

Standard document classifiers such as Naïve Bayes expect a single vector per document as input. This is at odds with distributional models, which typically output a vector per word or phrase. This section describes several algorithms that aim to represent an entire document as a single vector while preserving the distributional properties of its contents. These can be viewed as alternatives to the framework described above.

One family of algorithms do away with the idea of a single vector per word and attempt to learn a document vector instead. These proposals typically originate in the topic modelling and deep learning literature. Both fields are very active, with new papers being published almost every week. This section will only discuss several representative algorithms — thorough and up-to-date surveys are available in Schmidhuber (2015) and Blei (2012).

Topic models represent documents as a probability distribution over a fixed number of latent "topics", which in turn are distributions over words (Blei et al., 2003). Extensions have been proposed that can induce topic hierarchies (Blei et al., 2004), incorporate additional information into the topic distribution (McAuliffe and Blei, 2008; Andrzejewski et al., 2009; Steyvers, 2010) or allow for topics to evolve over time (Blei and Lafferty, 2006). It is worth pointing out that evaluation has become an important issue in the topic modelling community. Traditionally, topic models have been evaluated by measuring how well a trained model fits a held-out set. However, Blei (2012, p 83) points out that "there is no technical reason to suppose that held-out accuracy corresponds to better organization or easier interpretation [when topic models are used to organise, summarise, and help users explore large corpora]". One of the open questions in topic modelling is therefore to "develop evaluation methods that match how the algorithms are used" (ibid). This general argument is one of key theses of this work.

Le and Mikolov (2014) introduce an unsupervised method that extends WORD2VEC (Mikolov et al., 2013a) by considering a context window that spans longer pieces of text such as paragraphs or documents. Both words and paragraphs are modelled as fixed-length dense vectors, initialised randomly. The vector representations for several words and the paragraph they belong to are concatenated and trained to predict the next word in the paragraph. For instance, given the vectors for "the cat sat on the" and the vector for that paragraph, the task is to predict the next word, e. g. "mat". In this framework, word vectors contribute local information and the paragraph vector supplies global information. Kiros et al. (2015) present a related model, where a recursive neural network is applied sequentially to the word vectors of each sentence. Given a sentence, the task is to predict the preceding and following sentences in a novel. The underlying assumption is similar to the distributional hypothesis: sentences that co-occur in the same book should have similar representations.

Kim (2014) presents a convolutional neural network model for sentence modelling. The model is trained in a supervised fashion using pre-trained word embeddings, which can be obtained without supervision. The convolutional filter is applied to a window of several words in a sentence. The final layer is a softmax, meaning output is a probability distribution over labels. The model allows embeddings to be modified during training, essentially adapting them to the particular task. For instance, without task adaptation the neighbours of "good" are "great" and "bad", and with adaptation to a sentiment data set they become "nice" and "decent". This illustrates an important trend in the neural network literature — additional labelled data can be leveraged to improve performance at specific tasks. Zhang et al. (2015) present a similar model that does away with the notion of a word and can be trained directly on characters. Provided with a large labelled data set (in the order of several hundred thousand documents), the model can achieve good performance without any information about words or the syntactic structure of sentences.

Another family of algorithms keeps the idea of mapping each word to a vector and seeks ways of reducing multiple word vectors to a single document vector. My framework is one instance of such an approach. A popular method in the literature is to add or average all the vectors of words contained in the document (Klementiev et al., 2012; Lebret and Collobert, 2014). A variation of the same theme is to concatenate instead of add, but care needs to be taken to ensure the rep-

resentation of all documents is of the same dimensionality regardless of document length. Averaging and addition are insensitive to word order and grammar and treat all words the same despite evidence in the literature that function words should be treated differently to content words (Sadrzadeh et al., 2013, 2014). However, it is probably suitable for topic-based DC as including information about all words in a document can characterise the topic (bag-of-words models tend to work very well for such tasks). On the other hand, ignoring syntax and grammar will inevitably fail to capture important relations in the text such as negation. Such techniques would be expected to do poorly at sentiment analysis.

An alternative is to model a document as a bag of vectors (Kriegel and Schubert, 2004). The authors aim to capture different aspects of the semantics of the object being modelled (websites, 3D objects, etc). A training example (an entire website) contains multiple sentences (web pages), each of which is reduced to a single vector using a standard bag of words approach. In a distributional semantic context, a document could be represented as a bag of the distributed representations of the words it contains. The distances between each pair of documents is computed using a sum of minimum distances measure (Eiter and Mannila, 1997), and used in a k nearest neighbours (kNN) classifier. The disadvantages of this strategy include its running time (quadratic in the number of pages per site) and the fact that it is limited to kNN classification.

A recent instantiation of Kriegel and Schubert's idea is word-mover distance (Kusner et al., 2015). Documents are modelled as a bag-of-words, and each word is embedded in a distributional space. The distance between two documents is the sum of the total distance all words in one document need to "travel" to match a word in the other document. When the distance between each pair of documents has been computed, a kNN classifier can be used. The algorithm is attractive because it does not depend on a particular distributional model, has no hyper-parameters and is interpretable. A naïve implementation would be slow as it would involve computing the distance between each pair of words in two given documents, and between each pair of documents. Kusner et al. present a set of heuristics that allow the number of word-to-word and document-to-document comparisons to be reduced significantly, resulting in up to 100x speed-ups compared to the naïve version.

CHAPTER SUMMARY

This chapter argued that existing intrinsic evaluations are unreliable as they make use of small gold-standard data sets and assume there exists a single notion of similarity that is independent of a particular application. Performance at such intrinsic evaluations does not necessarily correlate with practical utility. I demonstrated empirically that the commonly used MC, RG and WS353 data sets fail to reliably detect the presence of random noise in word vectors.

I showed that the word similarity task is inherently harder for human annotators than a range of document-level annotation tasks. Motivated by this result, I introduced a novel extrinsic evaluation framework for distributional models based on distributional feature expansion applied to document classification. In that framework, all test-time document features are replaced with their nearest neighbour according to a distributional model. The accuracy of the classifier when a particular DM is used provides a direct measure of the DM's quality.

The framework is agnostic to how word and phrase embeddings are built; it can be used with words and phrases represented as vectors, matrices or trees. Compositional algorithms that operate on any grammatical structure can be evaluated within it.

The next chapter presents an evaluation of counting distributional models, WORD2VEC and GLOVE within my framework. I focus on the problem of noun phrase and verb phrase composition.

4

APPLYING THE FRAMEWORK

This chapter presents an evaluation of the methods for building word representations and composing them introduced in Chapter 2 in the framework introduced in Chapter 3. Specifically, the following questions are investigated:

- Which combination of word embedding algorithm and composition performs best for noun phrases and verb phrases?
- What is the impact of quantity and quality (domain and cleanliness) of unlabelled data in determining the success of a distributional model?
- Can the differences between multiple identical models trained on similar data be used to improve performance?
- Would the findings of this chapter hold if the parameters of the evaluation framework were set differently?

Results show that the determining factor in building word representations is data quality rather than quantity. In some cases only a small amount of unlabelled data is required to reach peak performance. Neural word representation algorithms perform better than counting-based ones regardless of what composition is used; simple additive composition algorithms consistently outperform more sophisticated competitors. These findings are consistent across a range of parameter settings for the evaluation framework.

I also consider a different way of encoding documents as vectors, which is based on vector quantization. The method achieves comparable accuracy with a significantly more compact document representation.

Finally, I introduce a new algorithm for improving the quality of distributional thesauri using information from multiple identical distributional models trained on different samples of the same data set.

Code for all experiments is available at `tinyurl.com/batchkarov`.

4.1 EXPERIMENT DETAILS

This chapter presents the results of a series of experiments, which were designed to address the research questions above. At a high level, the structure of an experiment is as follows. Word embeddings are learnt using one of four algorithms from one of three unlabelled corpora. Word vectors are composed into phrase vectors using one of ten compositional algorithms, which are applicable to different grammatical structures. Word and phrase vectors then become entries in a distributional thesaurus, which contains the nearest neighbours of each entry. Thesauri are evaluated at one of three labelled classification corpora using the framework introduced in Chapter 3. The rest of this section provides more details about each of the stages outlined above.

Distributional thesauri are built as follows. For each combination of word embedding algorithm A_u and compositional algorithm A_c:

1. Extract the sets E_U, E_{NP} and E_{VP} of all unigrams, NPs and VPs that occur in a given *labelled* corpus (Section 4.1.1). E_U only contains nouns, adjectives and verbs.

2. Using an *unlabelled* corpus and A_u, obtain a vector for each unigram in E_U (Section 4.1.2).

3. Use A_c to obtain a composed vector for each entry in $E_{NP} \cup E_{VP}$ (Section 4.1.3).

4. Use composed vectors to build a thesaurus, whose entries are $E_{NP} \cup E_{VP}$. The neighbours of each entry are $E_{NP} \cup E_{VP} \cup E_U$; these are sorted by Euclidean distance to the entry. Euclidean distance is used here instead of more popular alternatives such as cosine distance because it is a proper metric and satisfies the triangle inequality. It is therefore suitable for use with KD Trees, which provide an efficient way of computing the nearest neighbours of an entry (Section 3.5).

Each thesaurus is modified in two ways for a particular classification experiment:

1. Thesaurus entries not contained in the testing section of the labelled corpus are removed. These are not needed because the thesaurus is only queried for those entries that appear in the test set.

2. Neighbours not contained in the training set are removed because they are not known to the classifier and therefore cannot be used as replacements for test-time document features.

4.1.1 *Labelled classification corpora*

Classification experiments are carried out using three freely available labelled corpora.

MAAS is a collection of 50*K* film reviews collected from IMDB, split equally between the positive and negative classes (Maas et al., 2011).

R2 is a subset of the popular Reuters-21578 (Lewis et al., 2004). The full data set contains 21578 news articles in 92 classes. It is common to discard all but the 8 most frequent classes, which contain a total of 7674 documents. The distribution of documents across these 8 classes is heavily skewed — the most frequent class accounts for 52% of all documents. Dealing with the range of issues created by class imbalance (Chawla et al., 2004) is beyond the scope of this thesis. I therefore convert the 8-class data set to a two-class one by merging the seven less frequent classes. Each document belongs to a single class only. This results in a balanced data set, which is henceforth referred to as R2. Some statistics are shown in Table 4.1.

Class	Documents	Tokens	ANs	NNs	VPs
Earn	3923	202*K*	12*K*	30*K*	6*K*
Not earn	3751	376*K*	23*K*	49*K*	8*K*

Table 4.1: R2 data set statistics.

The AMAZON corpus is a newly constructed subset of the Amazon product review data set (McAuley and Leskovec, 2013). Five product categories (out of 33) with approximately 200*K* reviews each are selected. All reviews referring to the same product code are concatenated into a single pseudo document. Documents containing less than 50 words and the longest approximately 10% of documents per category are removed[1]. Some properties of the resultant data set are summarised in Table 4.2.

1 Documents were removed to simplify the classification task and eliminate some of the noise due to the automated nature of the collection process. Short documents contain very few NPs and VPs and cannot be reliably classified by the system used in this thesis. Some of the long documents were found to contain poorly stripped HTML markup. Others were reviews for popular products, which attract spam and general conversation that does not relate to the product.

Class	Documents	Tokens	ANs	NNs	VPs
Automotive	28*K*	2*M*	154*K*	190*K*	18*K*
Beauty	20*K*	2.5*M*	218*K*	175*K*	21*K*
Patio	13*K*	1.7*M*	144*K*	135*K*	15*K*
Pet supplies	13*K*	2.2*M*	176*K*	149*K*	21*K*
Baby	5.8*K*	2.8*M*	205*K*	178*K*	21*K*

Table 4.2: `Amazon` data set statistics.

These corpora are selected because they pose different challenges to a classifier. The classes in AMAZON and R2 are largely topic-based and test a compositional algorithm's ability to provide topically related neighbours. In contrast, the fine-grained propositional structure of a document is more important than its topical content in MAAS. For instance, in AMAZON it may be appropriate to substitute "good film" for "bad acting" as both phrases refer to the topic of cinematography. This substitution would not affect the decision-making process of a classifier that has seen both of these phrases in training. In contrast, the same substitution is likely to alter the decision of a classifier in a sentiment classification corpus such as MAAS.

All corpora are tokenised, sentence-segmented, lemmatised, part-of-speech (PoS) tagged and dependency-parsed using Stanford CoreNLP (Manning et al., 2014) and converted to lowercase. PoS tags are converted to a coarser set (Table 4.3). English stopwords[2] and words longer than 25 characters are removed. A Multinomial Naïve Bayes (McCallum and Nigam, 1998; Metsis et al., 2006) classifier[3] is trained. Documents are encoded using a bag-of-words representation with either NP, noun and adjective features or VP, noun, adjective and verb features at train time. At test time only NP or VP features are extracted (Chapter 3.4). All features that are not composable by the particular composer used in an experiment are removed. As a result, some test documents may be left with no features. These are removed from the test set.

4.1.2 *Unigram vectors*

Distributional representations for all unigrams contained in the labelled data are extracted from three large corpora. GIGAWORD is the

2 List of stopwords can be found at http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words. Last visited 16 February 2016.

3 As implemented in SCIKIT-LEARN (Pedregosa et al., 2011)

Coarse tag	Fine-grained tags
N (noun)	NN, NNS, NNP, NPS, NP
J (adjective)	JJ, JJN, JJS, JJR
V (verb)	VB, VBD, VBG, VBN, VBP, VBZ

Table 4.3: Fine-to-coarse part-of-speech tag mapping. Based on Petrov et al. (2011).

AFP section of English Gigaword v4. It contains approximately 200 million tokens of news text. WIKIPEDIA is a mid-2011 copy of English Wikipedia, which contains approximately 1.5 billion tokens. Only the body of each article is extracted. CWIKI is a cleaned version of WIKIPEDIA, provided by `lateral.io`[4]. The key intuition behind this corpus is that Wikipedia contains a large number of automatically generated articles, which are usually detrimental to model performance. However, these tend to attract little interest from human readers. CWIKI therefore contains only those articles that were viewed more than 20 times on a day in June 2015. The corpus contains 525 million tokens.

All unlabelled corpora are pre-processed similarly to the labelled ones. Each word is concatenated with its coarse-grained part-of-speech tag. This provides a basic form of word sense disambiguation. For instance, different vector representations will be learnt for "bank" when used as a noun or a verb. A sample sentence from CWIKI reads as follows:

> the/DET novel/N be/V also/RB release/V as/CONJ a/-DET audiobook/N ./PUNCT

For dependency-based counting models (Chapter 2) the following features are extracted:

NOUNS All adjectival modifiers and conjuncts for the target noun, as well as all words for which the target noun serves as a direct object, subject, compound noun modifier or a prepositional object.

ADJECTIVES All nouns the adjective modifies and all of its conjuncts.

VERBS all adverbial modifiers, subjects, objects and all prepositions the verb is an object of.

4 Available at https://blog.lateral.io/2015/06/the-unknown-perils-of-mining-wikipedia/. Last accessed 15 August 2015.

The direction of each dependency relation is also recorded. For instance, the noun feature "amod-DEP:romanian/J" indicates the target noun has been the governor of an amod relation with "romanian" (had an amod dependent). Some of the 6038 unique dependency features of "population/N" are "dobj-HEAD:make/V" (197 times in WIKIPEDIA), "dobj-HEAD:have/V" (95946 times) and "amod-DEP:german/J" (932 times). These unigram vectors are henceforth referred to as DEPENDENCY vectors.

For proximity counting-based models a window size of 7 words on either side of the target unigram is used. Some of the 7460 features on "population/N" in WIKIPEDIA are "town/N" (17324 times in WIKIPEDIA), "germany/N" (466) and "during/CONJ" (3220). The context window of a word is not allowed to cross sentence boundaries. On average across the corpus, the effective size of the window is about 5. These unigram vectors are henceforth referred to as WINDOW vectors.

Only entries and features that appear more than 40 times in the unlabelled data are kept in the model. For each entry, features that co-occur with it less than 15 are removed for that entry, but not for other entries. Vectors are re-weighted with PPMI and then reduced to 100 dimensions with SVD. Initial experiments suggested PPMI improves vector quality considerably, while SVD results in minor reductions. Despite this, SVD is used in all experiments below for practical reasons — it would have been impracticable to run all of the experiments in this chapter given the constraints of available resources and time.

In terms of neural models, I experiment with the algorithm of Mikolov et al. (2013a) as implemented in GENSIM (Řehůřek and Sojka, 2010). Embeddings were learnt using the skip-gram model with hierarchical sampling, 100 dimensions, a window of size 5[5] and a minimum frequency of 50. This model will be referred to as WORD2VEC.

I use Pennington et al.'s C implementation of the GLOVE model[6]. Parameters are VOCAB_MIN_COUNT=50, VECTOR_SIZE=100, MAX_ITER=30 and WINDOW_SIZE=5.

I also use a pre-trained $L2$-normalised 100-dimensional version of the embeddings proposed by Collobert and Weston (2008). These are included with the code of Socher et al. (2011) (Section 4.1.3) and are trained by Turian et al. (2010, Section 6) on 37 million words from

5 This is comparable to the effective window size for my implementation of counting models.

6 Available at http://nlp.stanford.edu/projects/glove/. Last accessed 16 August 2015.

the RCV1 newswire corpus. Throughout this thesis these vectors are referred to as TURIAN. The embeddings are modified as follows.

The entries distributed by Socher et al. (2011) were not preprocessed, so some some types are repeated. For instance, the tokens "average", "Average" and "averages" all have an associated vector. I remove all punctuation tokens, then lowercase and lemmatise each entry. Where multiple entries map to the same canonical form I select the shortest original entry; ties are broken by giving preference to words that are already lowercased. The lemma "average" is therefore represented by the vector associated with the raw entry "average" (as opposed to "Averages"). The resultant canonical entry forms do not have an associated part-of-speech tag, which makes them incompatible with the rest of the entries used in this chapter. I expand each canonical form to three PoS-tagged lemmata by appending an N, J and V tag; all forms share the same vector representation. For example, the canonical form "average" expands to "average/N", "average/J" and "average/V", which all use the vector for the un-canonicalised token "average".

4.1.3 *Composing unigram representations*

Distributional representations for three types of phrases are built:

AN a noun phrase consisting of a noun and a single attributive adjective, where an `amod` dependency relation (De Marneffe and Manning, 2008) holds between the two. Noun phrases consisting of multiple adjectives and a noun are treated separately, e.g. "large black cat" is translated to "large cat" and "black cat". This is in line with most existing work in distributional compositional semantics, where only noun phrases containing a single adjectival modifier are considered for simplicity.

NN a noun phrase formed by two or more nouns, where a `nn` relations holds. These are broken down similarly to ANs.

SVO/VP a transitive sentence consisting of a verb and its noun subject and object. An `nsubj` relation must hold between the verb and the subject, and `dobj` between the verb and the object. In a slight abuse of established terminology, I will interchangeably refer to such sentences as transitive verb phrases (VPs) or subject-verb-objects (SVOs).

I experiment with a number of compositional algorithms, which are reviewed in detail in Section 2.2. The composers henceforth referred to as ADD and MULT implement pointwise addition and multiplication. SOCHER composes using a recursive autoencoder neural network as introduced by Socher et al. (2011). I use Socher et al.'s MATLAB implementation[7], which is hardcoded to use TURIAN vectors.

BARONI utilises the algorithm of Baroni and Zamparelli (2010) to learn a matrix for each modifier of a noun phrase. This approach achieves state-of-the-art performance at NP composition when evaluated intrinsically (Grefenstette et al., 2013). Matrices for both adjective and noun modifiers are learnt as follows:

1. Construct a space containing the corpus-observed count vectors for up to 200K most frequent nouns, 200K most frequent verbs and 100K most frequent adjectives.

2. Reduce space to 100 dimensions using Singular Value Decomposition. The number of dimensions is chosen such that the model is compatible with the pre-trained TURIAN vectors.

3. Construct corpus-observed vectors for all NPs that appear at least 50 times in the unlabelled corpus. Observed window vectors for NPs are built similarly to these for unigrams.

4. Apply the SVD transformation learnt in Step 2 to all NP vectors collected in Step 3.

5. Select all adjective and noun modifiers that appear at least once in the labelled corpus and in at least 50 NPs in the unlabelled corpus.

6. Learn a matrix[8] for each of the selected modifiers using the selected NPs as training data.

GUEVARA (Guevara, 2010) is trained similarly to BARONI, except a single matrix is learnt for all adjectival and noun modifiers instead of one per modifier.

COPYSUBJ, COPYOBJ, FADD and FMULT are the copy-subject, copy-object, Frobenius Addition and Frobenius Multiplication category-theoretical model from Section 2.2.4.

7 Available at http://tinyurl.com/socher2011. Last accessed 16 August 2015.
8 Using https://github.com/composes-toolkit/dissect with default settings.

Additionally, six non-compositional baselines are used. RANDV assigns a random vector to each document feature. RANDN does not assign a vector to phrases, but returns randomly selected train-time features for each test-time feature. LEFT and RIGHT represent a noun phrase with the vector of its leftmost and rightmost constituent respectively (only applicable to NP composition). OBSERVED uses the corpus-derived observed vector for an NP or VP. VERB models a VP using the vectors of its head. BOF is the traditional bag-of-features model, where the features are NPs, VPs and unigrams at training time and only NPs and VPs at test time.

Statistics about the sizes of various thesauri are shown in Table 4.4.

CHOICE OF COMPOSER In this chapter the choice of which composer to apply to which word vectors is driven by both theoretical and practical considerations. The ADD, MULT, LEFT and RIGHT composers are henceforth referred to as "core" algorithms because they can be applied to any grammatical structure and any unigram vectors, regardless of how they were constructed. In contrast, COPYOBJ, FADD and FMULT are only applicable to verb phrases and BARONI and GUEVARA are only applicable to noun phrases. BARONI, GUEVARA and OBSERVED require corpus-attested (observed) vectors for phrases to be extracted, which is easily done for count window vectors and can in principle be done for WORD2VEC and GLOVE. SOCHER can only be applied to TURIAN embeddings as the code needed to re-train SOCHER on a different set of word embeddings is not publicly available.

	UNI	NPs	VPs
Add	0.22	5.13	0.92
Left	0.22	6.75	1.11
Gue	0.22	5.13	-
Bar	0.22	1.12	-
Obs	0.22	0.45	0.01
Verb	0.22	-	1.37

(a) GIGAWORD windows

	UNI	NPs	VPs
Add	0.69	7.85	1.28
Left	0.69	8.47	1.34
Gue	0.69	7.85	-
Bar	0.69	3.99	-
Obs	0.69	1.56	0.01
Verb	0.69	-	1.41
CopyObj	0.69	-	0.98

(b) WIKIPEDIA windows

	UNI	NPs	VPs
Add	0.28	5.77	1.01
Left	0.28	7.18	-
Right	0.28	7.65	-
Verb	0.28	-	1.38
CopyObj	0.28	-	0.60

(c) GIGAWORD WORD2VEC

	UNI	NPs	VPs
Add	0.82	8.13	1.30
Left	0.82	8.64	-
Right	0.82	8.95	-
Verb	0.82	-	1.39
CopyObj	0.82	-	1.00

(d) WIKIPEDIA WORD2VEC

	UNI	NPs	VPs
Add	0.82	8.08	1.30
Left	0.82	8.62	-
Right	0.82	8.92	-
Verb	0.82	-	1.40

(e) WIKIPEDIA GLOVE

	UNI	NPs	VPs
Add	0.32	5.80	0.99
Left	0.32	7.24	-
Right	0.32	7.64	-
Verb	0.32	-	1.36
Socher	0.32	4.92	0.82

(f) TURIAN

Table 4.4: Number of document features that appear in any of the labelled corpora and are also present in a particular DM. All figures are divided by 10^5 for brevity. UNI, NPs and VPs stand for unigrams (adjective, nouns and verbs), noun phrases (ANs and NNs) and verb phrases (SVOs) respectively. The maximum value in the UNI, NPs and VPs column is 2.5, 9.53 and 1.43, for a total of 13.46 or $1.346M$ document features (adjectives, nouns, verbs, NPs and VPs) across all labelled corpora. These can be used to calculate percentage coverage. For example, TURIAN vectors with SOCHER composition cover $4.92/9.53 = 51.6\%$ of all NPs in the labelled corpora. MULT is omitted as it has the same coverage as ADD. COPYSUBJ, FADD and FMULT have the same coverage as COPYOBJ.

4.1.4 *Measuring classifier performance*

The performance of document classifiers is measured here in terms of accuracy. Confidence intervals are estimated via bootstrapping (Efron and Tibshirani, 1994, Chapter 13) as follows.

The labelled document collection is split randomly into a training (80%) and evaluation (20%) section. The choice of a training and test set is random, but it is kept the same throughout all experiments. A classifier is trained (on the former) and evaluated (on the latter) once. Its predictions for each document in the test set are recorded along with gold standard labels. The predictions are resampled 500 times with replacement, each time calculating an accuracy score. I report the mean and 68% interval centered on the mean of the 500 accuracies.

An example of bootstrap estimation of confidence interval is shown in Table 4.5. A model is trained and subsequently evaluated once on a collection of 10 data points. The gold standard for the test set is 1111100000 and the predictions of the model are 0111100001, for an accuracy of 0.8. Each time, a new sample of 10 items is selected with replacement from the original test set (second column) and an accuracy score is computed for the sample (last column). The accuracies are averaged over all samples. The distribution of bootstrap scores is shown in Figure 4.1.

#	Selected indices	Accuracy
1	4 6 3 2 9 2 9 5 7 8	0.7
2	3 6 4 9 1 8 9 4 9 5	0.5
3	4 0 1 2 5 4 3 4 5 9	0.6
4	0 1 1 5 2 8 5 7 3 8	1.0
5	3 4 0 6 4 5 9 2 9 1	0.6
⋮	⋮	⋮
10000	6 7 5 9 9 1 5 5 7 9	0.7
	Mean over 1000 samples	0.795

Table 4.5: Bootstrap estimation of model accuracy.

Confidence intervals are commonly estimated via bootstrapping when evaluating systems with a fixed test set (Tjong Kim Sang and De Meulder, 2003). This is because 1) it allows different systems to be compared fairly as potential differences due to how the data is partitioned are eliminated, and 2) other techniques such as k-fold cross-validation can be computationally expensive. In my framework,

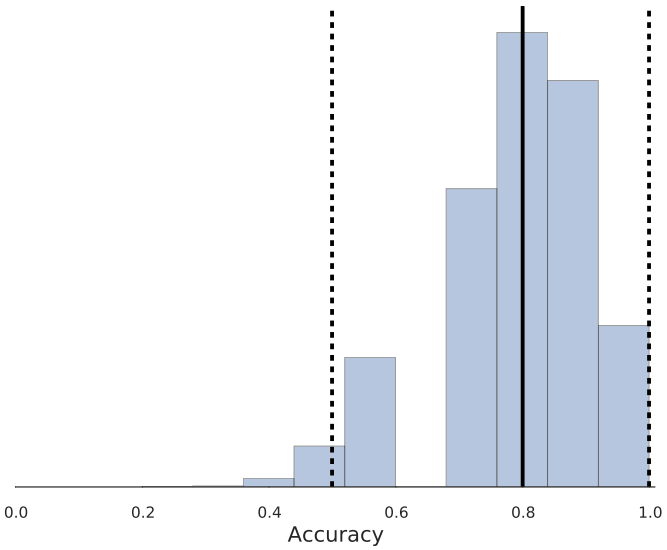


Figure 4.1: Distribution of accuracy scores for a system, estimated via bootstrap resampling. The solid black line denotes mean accuracy over 50000 samples, which is .798. The dashed lines denote the 95% confidence interval, which is calculated by taking the 2.5-th and 97.5-th percentile of the empirical distribution and spans the range 0.5–1. The interval is so broad because of the small size of the data set (only 10 data points).

a nearest-neighbour query is performed for each document feature in the test set. Even with a KD tree, this process can be slow for high-dimensional DMs or ones with many entries[9]. Bootstrapping reduces the number of nearest neighbour queries and significantly improves running time.

4.1.5 *Significance testing*

Significance tests are carried out via a randomised hypothesis testing procedure, also known as a permutation test (Efron and Tibshirani, 1994, Chapter 15). This approach is preferred over other methods, such as t-test or overlap in confidence intervals, because it is non-parametric and has good theoretical guarantees.

Given two models M_A and M_B, their predictions a and b and the corresponding gold standard labels G_A and G_B, the null hypothesis is that M_A is not significantly different to M_B. If $|a|$ denotes the size of a, we require that $|a| = |G_A|$ and $|b| = |G_B|$, but it is possible that $|a| \neq |b|$. Recall that test-time documents with no features in a

9 For an 80–20 split of AMAZON an average of $1.4M$ queries are performed over a pool of $200K$–$500K$ candidate neighbours. A single classification experiment takes between several tens of minutes and a day. MAAS and R2 are significantly faster.

particular distributional model are removed. Therefore, M_A and M_B may be able to classify a different number of documents at test time.

Denote the difference in accuracy between M_B and M_A as Δ. The aim is to test if Δ is statistically significant, i. e. if a value of the test statistic as large as Δ could have been observed by chance. The algorithm is best illustrated with an example. Suppose we are given a test set of 4 documents, divided equally between two classes, and that M_A and M_B can classify 4 and 3 of those documents respectively (Table 4.6). The point estimates for the accuracies of the two systems are $2/4 = 0.5$ and $2/3 \approx 0.66$ respectively, and $\Delta \approx 0.16$. To perform a significance test, both predictions and gold standard labels are concatenated (Table 4.6a). The rows of the resultant matrix are shuffled, but the first column is always reset to its original state (Table 4.6b). After shuffling, the accuracy of M_A and M_B is .75 and 0.33 respectively. The value $\Delta = -0.42$ is recorded and the data are shuffled again (Table 4.6c). Accuracies are now 0.5 and 0.66 and $\Delta = 0.16$. When the shuffling is repeated many times, typically in the order of hundreds or thousands, the proportion of time where Δ decreased (compared to the original value of 0.16) after shuffling is the p-value of the test. When calculating the ratio, a smoothing factor of 1 is traditionally added to the enumerator and denominator. In our example, Δ decreased once and did not decrease once, so $p = (1+1)/(1+2) \approx 0.66$. There is therefore insufficient evidence to reject the null hypothesis that M_B is as good as M_A.

M	G	P
M_A	0	1
M_A	0	0
M_A	1	0
M_A	1	1
M_B	0	0
M_B	0	1
M_B	1	1

(a) System output

M	G	P
M_A	0	0
M_A	1	0
M_A	0	0
M_A	1	1
M_B	0	1
M_B	0	1
M_B	1	1

(b) Permutation 1

M	G	P
M_A	0	1
M_A	0	0
M_A	0	0
M_A	0	1
M_B	1	1
M_B	1	1
M_B	1	0

(c) Permutation 2

Table 4.6: An illustration of randomised significance testing. M stands for Method name and is used to identify the two methods being compared. G is the Gold standard label for each of the documents, i. e. the concatenation of G_A and G_B. P is the Prediction of each system, i. e. the concatenation of a and b.

The intuition behind randomised significance testing is simple. Supposing M_B is genuinely better than M_A, when a and b are shuffled M_B would exchange some of its correct predictions for M_A's incorrect predictions. The accuracy of M_B would decrease and the accuracy of M_A would increase, resulting in a reduction in Δ. The larger the original difference, the higher the probability that the difference would be reduced. Conversely, if M_A and M_B are not significantly different, the shuffling would exchange M_A's predictions for approximately equally good predictions. The accuracies of M_A and M_B would not change considerably, and Δ would increase or decrease with an equal probability.

A NOTE ON TERMINOLOGY The term "significant" is used frequently in the remainder of this chapter in the sense of "statistical significance". It refers to a difference between two models that is too large to have been observed by chance. For instance, a model with a mean accuracy of 0.72 ± 0.0001 is significantly better than a model that scores 0.71 ± 0.0001. However, while the difference may not be due to chance, it is not large enough to be *scientifically significant*. Although it may be statistically significant, an improvement of 0.01 is too small to be of practical importance. In addition, I shall use the term "considerable" to refer to cases where the difference subjectively appears to be large, but I do not wish to make a statement about statistical significance.

4.2 NON-DISTRIBUTIONAL CLASSIFICATION RESULTS

Table 4.7 shows the performance of several classifiers that do not make use of distributional information, namely the BOF, RANDN and RANDV models described in Section 4.1.3. These should be viewed as a strong baseline against which classifiers backed by a distributional model can be compared.

Both random baselines achieve similar scores (Table 4.7). These are indicative of the class balance in the data sets — R2 and MAAS are balanced, whereas AMAZON is skewed in favour of Automotive and Beauty product reviews. Comparing the three AMAZON experiments, a reasonably high accuracy can be obtained with only NPs as features. Adding unigrams at test time results in a moderate improvement. However, using VPs instead of NPs results in a considerable reduction in accuracy. This is likely due to the sparsity of verb phrases

Corpus	Test features	BOF	RANDV	RANDN
R2	NP	0.928	0.502	0.502
Maas	NP	0.72	0.49	0.48
Amazon	NP	0.845	0.218	0.218
Amazon	J+N+NP	0.897	0.218	0.218
Amazon	J+N+V+VP	0.732	0.218	0.218

Table 4.7: Accuracy of non-distributional classifiers. Bootstrap confidence interval is less than < 0.02 around the mean, typically in the order of 0.01.

(recall they are trigrams). Note that the performance of these models is lower than published results (Maas et al., 2011) due to my use of a fairly simple classification pipeline with no feature selection, feature reweighting or dimensionality reduction and only a small subset of all unigrams and bigrams as features at test time.

4.3 TASK VERIFICATION

One of the core requirements towards an evaluation framework is that random vectors should result in random performance. Section 3.1.5 demonstrated only two of the four word similarity data sets satisfy this requirement. This section answers the following research question:

> Is the extrinsic framework presented in Chapter 3 sensitive to the quality of word vectors?

The evaluation is done as follows. Similarly to Section 3.1.5, uniform random noise $\mathcal{U}(-n, n)$ is added to all elements of all NP vectors, where n is a free parameter. For the purposes of this example, a WORD2VEC model with ADD composition trained on GIGAWORD is used. The model is evaluated on the AMAZON corpus with noun, adjective and NP features at train time and NP features only at test time. Results are shown in Figure 4.2. The x axis corresponds to the value of the parameter n. The black horizontal line is the RANDV baseline. Performance degrades considerably as more noise is added. There is a significant ($p < 0.01$) difference between each pair of adjacent data points in the graph with the exception of the very first pair (noise levels 0 and 0.2). These results show the classification framework meets the requirement above and can reliably detect degradation in vector quality.

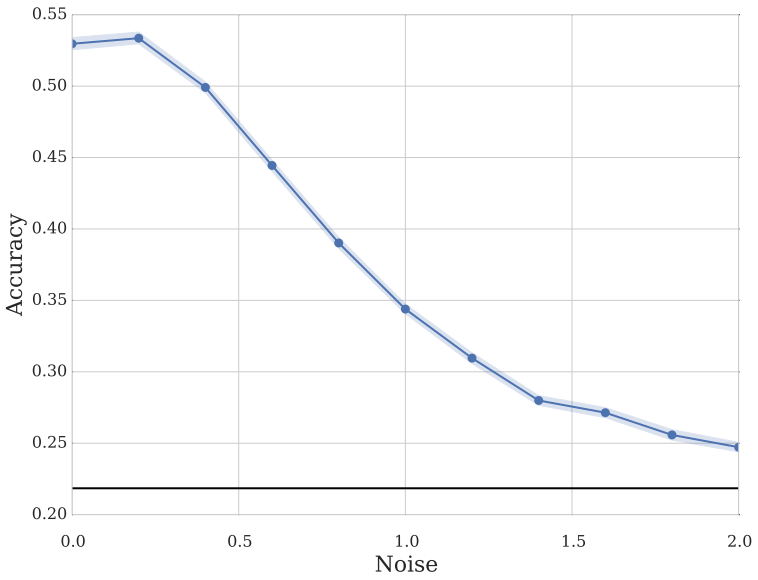


Figure 4.2: Effect of adding noise to vectors. Shaded area shows the 68% confidence interval.

4.4 NOUN PHRASE EXPERIMENTS

This section compares different word embeddings and compositional algorithms. I investigate how models of composition should be instantiated in the context of document classification. Here the focus is exclusively on noun phrases; verb phrases are considered in Section 4.5. All reported results are on the AMAZON labelled corpus (Section 4.1.1) unless stated otherwise. The reasons for this will become clear in Section 4.4.2.

Note that even though this thesis focuses on models of composition, a lot of the discussion in this section is about word-level models. This is because word embeddings are an essential component of any compositional system. Also, the evaluation framework presented in Chapter 3 can be use to evaluate both word-level and phrase-level distributional models.

4.4.1 *Embedding and composition algorithm*

This section attempts to answer the following question:

> Which combination of word embeddings and NP composition algorithm performs best for the AMAZON and R2 topical corpora?

Figure 4.3 shows the performance of the four core compositional algorithms when applied to WORD2VEC, GLOVE and counting unigram embeddings trained on WIKIPEDIA. Counting models are not significantly different to GLOVE in terms of classifier accuracy when LEFT or RIGHT composition is used. However, window models perform significantly better than GLOVE with MULT composition. The opposite is true for ADD composition. This suggests the quality of the unigram vectors produced by both models is comparable, but GLOVE is more amenable to additive than to multiplicative composition.

Window vectors are marginally but significantly better than dependency ones, except in the case of MULT composition. I hypothesise this is because dependency embeddings are sparser in general, but also due to my use of only a subset of all dependency relations of an entry.

WORD2VEC consistently and significantly outperforms GLOVE. This contradicts the results of Baroni et al. (2014), who found the two models to be comparable in a range of word similarity tasks. I am unable to offer an explanation for this result.

ADD composition outperforms MULT, and both are considerably better than LEFT and RIGHT regardless of what unigram embeddings are used. This is a general trend in all experiments in this chapter. One possible explanation is that LEFT and RIGHT are inferior as they discard half of the available distributional information. There is a possibility that MULT reverses the sign of elements of word embeddings, which sends composed vectors far away in semantic space from their constituents. For instance, if $\overrightarrow{black} = [-1, -2]$, $\overrightarrow{cat} = [-.1, -1]$ and $\overrightarrow{democracy} = [1, 0.5]$, then $\overrightarrow{black\_cat} = [-1, -2] \odot [-.1, -1] = [.1, 2]$, which is closer to $\overrightarrow{democracy}$ than to either $\overrightarrow{black}$ or $\overrightarrow{cat}$. The question of why GLOVE embeddings are less suitable for multiplicative composition than WORD2VEC requires further investigation.

Table 4.8 shows the performance of all compositional algorithms applicable to window word vectors. Again, ADD and MULT are the top performers, closely followed by GUEVARA. BARONI performs poorly and is not significantly different from a random baseline. I hypothesise this is due to the interaction between the task setup and BARONI's poor coverage. Recall when replacing document features with their distributional neighbours at test time, only those features for which a vector can be derived by a particular model of composition are considered. Some documents may be left with very few features, or only with features which are uninformative to a classifier.

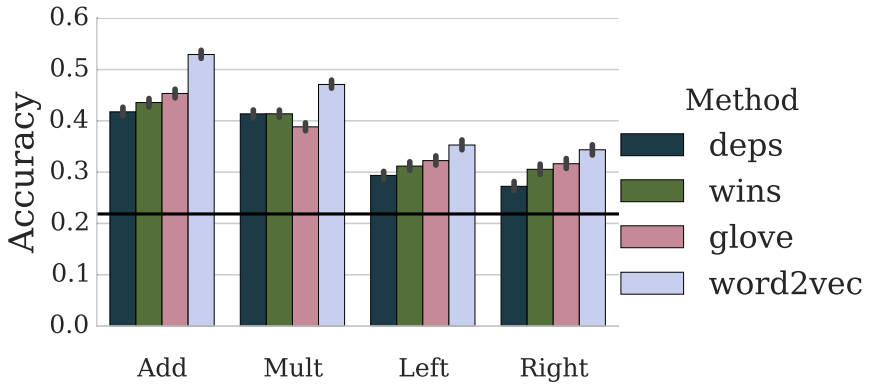


Figure 4.3: WORD2VEC vs GLOVE vs counting models trained on WIKIPEDIA and evaluated at AMAZON. Black horizontal line is the RANDN baseline. Error bars span the 68% confidence interval.

This in turn results in low classification accuracy, even if the replacements subjectively appear appropriate. The task therefore penalises composers with poor coverage. BARONI can provide vectors for a small subset of all NPs contained in the labelled data because of its reliance on observed phrase vectors for training. With a threshold of 50 observed NP vectors per modifier only 2788 modifier matrices can be learnt from WIKIPEDIA, which means BARONI can compose less than 29% of all document features contained in the labelled corpora. For GIGAWORD these figures are 456 and 9% respectively. The coverage of other composers such as ADD is as much as 6 times higher (Table 4.4).

Composer	Accuracy
Add	0.44±0.01
Mult	0.41±0.01
Guevara	0.39±0.01
Left	0.31±0.01

Composer	Accuracy
Right	0.31±0.01
Observed	0.28±0.01
Baroni	0.23±0.01
RandN	0.22±0.01

Table 4.8: Window vectors with various composers. Each composer is significantly better than the one following it in the table, with the exception of 1) LEFT and RIGHT, and 2) BARONI and the random baseline. Throughout the rest of this chapter, ± denotes the range of the (bootstrapped) empirical distribution of the mean. For example, 0.31 ± 0.01 means accuracy ranges from 0.3 to 0.32 with an average of 0.31.

Does model coverage relate to accuracy?

To empirically test the hypothesis that low coverage negatively correlates with classifier accuracy, a series of experiments are conducted where the coverage of a model X is "reduced to" that of another model Y. A vector for a phrase is successfully returned by X only if one would have been returned by Y[10]. This is implemented by training X as usual and removing all vectors that do not occur in Y. Coverage reduction eliminates the advantage of a DM, which may be due to better coverage, and isolates the effect of vector quality. Figure 4.4a shows the accuracy of the four core composers when their coverage is reduced to that of BARONI (trained on the same corpus). Accuracy decreases considerably for three out of the four composers, but not as far as BARONI's original accuracy of 0.23. This suggests the poor accuracy of BARONI can only partially be attributed to coverage issues.

Conversely, the differences between WORD2VEC and count window vectors in Figure 4.3 cannot be explained by coverage as the two thesauri contain a comparable number of entries (Table 4.4). The results of reducing WORD2VEC's coverage to that of count windows is shown in Figure 4.4b. As expected, none of the differences are significant.

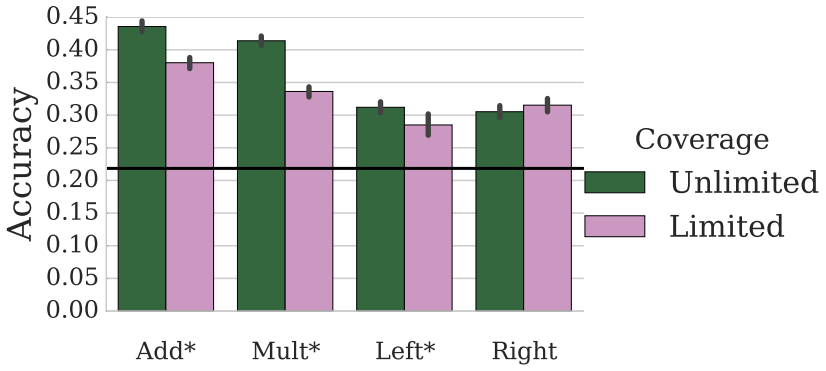
4.4.2 *Domain and size of unlabelled corpus*

This section investigates the following question:

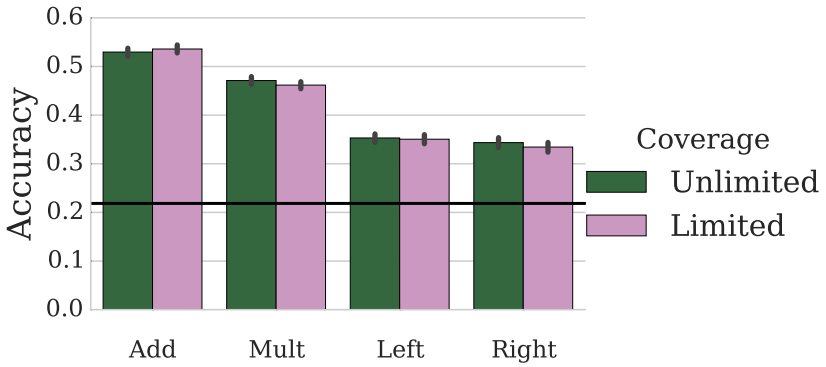
> How is classification accuracy affected by qualitative and quantitative differences in the unlabelled data used to construct word embeddings?

I compare three WORD2VEC models with identical parameter settings, trained on 1) the entire GIGAWORD corpus, 2) 15% of WIKIPEDIA, which contains approximately the same number of tokens as GIGAWORD, and 3) the entire WIKIPEDIA corpus. WIKIPEDIA-based DMs perform considerably better than GIGAWORD-based DMs across the board (Figure 4.5). Using a different corpus of equivalent size can have a comparable or larger effect to increasing the amount of unlabelled data sixfold (from 15% to 100% of WIKIPEDIA). Following

10 Note that reducing coverage in this way has no effect if the vocabulary of Y is a superset of the vocabulary of X.



(a) Core composers reduced to coverage of BARONI. Count windows vectors trained on WIKIPEDIA.



(b) Core composers using WORD2VEC embeddings reduced to coverage of core composers using count windows.

Figure 4.4: Reduced coverage results. In both sub-figures, the left and right bar correspond to accuracy before and after coverage is reduced respectively. Significant differences ($p < 0.01$) are marked with an asterisk. The black horizontal line is the RANDV baseline.

Pennington et al. (2014), who observe similar differences in performance between WIKIPEDIA and GIGAWORD, a plausible explanation is that WIKIPEDIA captures a broader range of topics than GIGAWORD, thus providing better coverage of the diverse range of topics in the AMAZON data.

However, GIGAWORD consistently outperforms WIKIPEDIA across all word embedding algorithms at the R2 corpus (Table 4.9). I hypothesise this is because both GIGAWORD and R2 are newswire corpora so the learnt word representations result in more appropriate replacements for this particular classification task. GLOVE and dependency window vectors are comparable, and both are worse than WORD2VEC and count window vectors.

However, the confidence intervals at R2 are considerably larger than they are at AMAZON. This is because the data set is smaller,

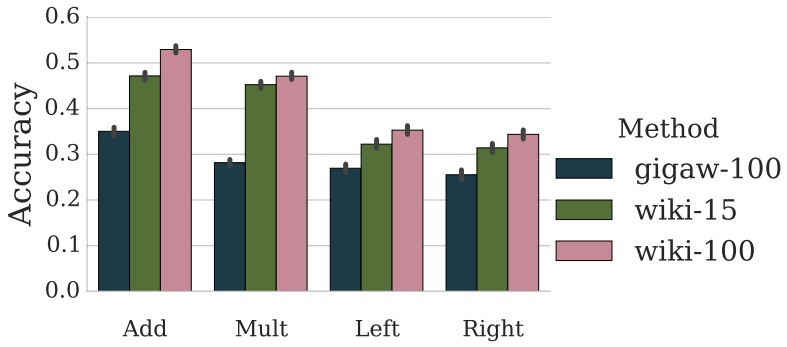


Figure 4.5: Gigaword vs Wikipedia as unlabelled corpus

and so is the train-time vocabulary. This has the potential to reduce accuracy. At test time, a distributional model may only have a limited set of replacements to choose from. The list of candidate replacements for a test-time feature can be empty, or it can contain only poor neighbours (in the context of that particular classification task). The smaller the set of candidates is, the higher the odds it contains no good neighbours and the more sensitive accuracy is to the training-time vocabulary. This causes accuracy to vary considerably when the labelled data set is small, so it is harder to find significant differences between DMs. For this reason most of the results reported in this chapter are on the AMAZON corpus, which is large enough to ensure accuracy does not vary much.

Embeddings	Unlabelled	Accuracy
WORD2VEC	GIGAWORD	0.741±0.029
WORD2VEC	WIKIPEDIA	0.680±0.032
WORD2VEC	CWIKI	0.648±0.034
Windows	GIGAWORD	0.719±0.032
Windows	WIKIPEDIA	0.669±0.037
Dependencies	GIGAWORD	0.641±0.035
Dependencies	WIKIPEDIA	0.614±0.036
GLOVE	WIKIPEDIA	0.618±0.027

Table 4.9: Performance at R2. Selected combinations of word embedding algorithm and unlabelled corpus; ADD composer.

Learning curves

This section seeks to better understand the behaviour of DMs when adding more unlabelled data.

I train WORD2VEC on increasingly larger subsets of WIKIPEDIA. Results are shown in Figure 4.6. The performance of all compositional algorithms improves as extra unlabelled data is added, though gains diminish quickly after around 600M tokens (about 40% of Wikipedia). This is a modest amount of data compared to the tens of billions of tokens sometimes used to train DMs of this kind (Pennington et al., 2014). This result appears to contrast with the general finding that more data consistently improves embedding quality. Also, vectors trained on CWIKI perform considerably better than ones trained on plain WIKIPEDIA, highlighting the importance of good-quality unlabelled data.

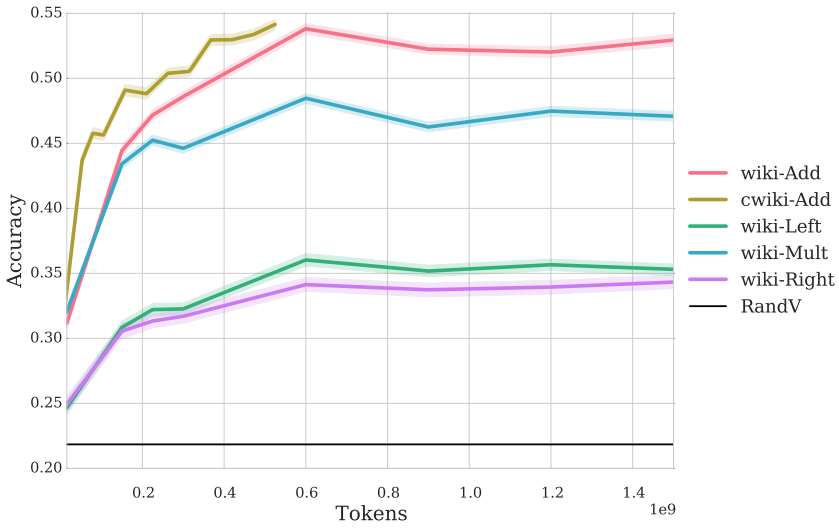


Figure 4.6: WORD2VEC learning curve when trained on WIKIPEDIA and CWIKI. The minimum value on the x axis is 1.5M tokens. The shaded region represents the 68% confidence interval around the mean. The black horizontal line is the RANDV baseline.

The general shape of the learning curves can be explained by considering two distinct events that occur simultaneously as more unlabelled data is added to a partly trained model:

VECTORS FOR NEW TYPES Type coverage improves as new word types are encountered and embeddings are learnt for them.

BETTER VECTORS FOR KNOWN TYPES More occurrences of types that were in the model initially are encountered.

I hypothesise the initial performance gains in Figure 4.6 are due to the DM acquiring coverage of the target domain, and subsequent improvements are due to improved embedding quality arising from the availability of more training data.

Relative contribution of quantity and domain of unlabelled data

This section investigates the relative contribution of quantity and domain of unlabelled data. This can be measured empirically by isolating the influence of new types entering the system. The experiment is designed as follows. Starting with a DM trained on the first N% of WIKIPEDIA, remove from the model all vectors for words that do not occur in the first $M \ll N$% of WIKIPEDIA (reduce the coverage of the former to the coverage of the latter, as in Section 4.4). The vocabulary of the former is strictly a superset of the vocabulary of the latter.

Results are shown in Figure 4.7. The x axis corresponds to the value of N, or the percentage of WIKIPEDIA used to train WORD2VEC. Each coloured curve corresponds to a different value of N. The top curve, labelled $M = 100\%$ in the figure, corresponds to not placing any constraints on the vocabulary of the DM (the same curve is labelled `wiki-Add` in Figure 4.6).

Only a modest reduction in accuracy is observed between $M = 100\%$ and $M = 10\%$. However, the difference between $M = 100\%$ and $M = 1\%$ is considerably larger. This confirms the hypothesis that the large initial gains in accuracy are due to improvements in the model as vectors for new types are learnt, and subsequent modest improvements are due to better signal for existing types. In other words, a certain critical amount of unlabelled data is needed in order to reach reasonable performance, and after that point returns diminish quickly. I will refer to that amount as "critical threshold".

Understanding the critical threshold

The critical threshold is a property of the unlabelled corpus used to train distributional models, but also of the labelled corpus the model is applied to. It is not a hard-and-fast boundary, but rather a measure of how much of the unlabelled corpus needs to be ingested before classification accuracy stops improving. For the WIKIPEDIA–AMAZON corpus combination, the critical threshold is at about $200M$ tokens (Figure 4.6). As we saw earlier, this value is closely related to type coverage. The relationship between tokens ingested and type

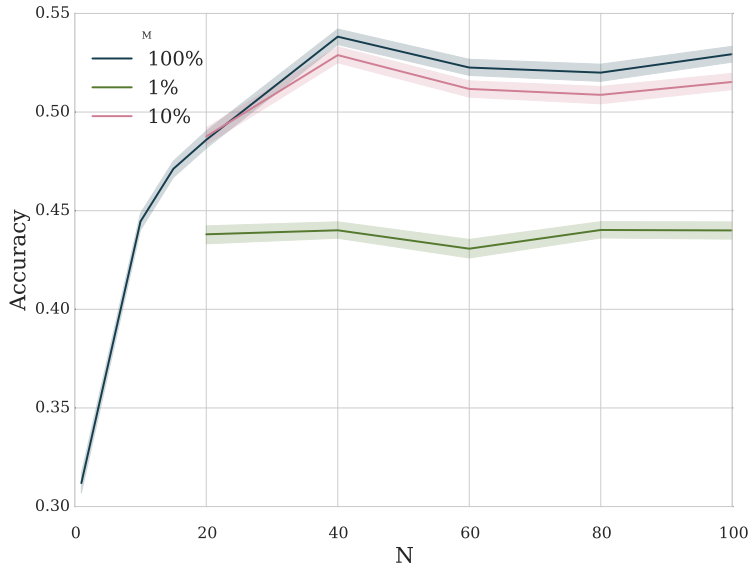


Figure 4.7: Learning curve with reduced coverage (WORD2VEC, ADD composition).

coverage is shown in Figure 4.8. The x axis corresponds to the number of WIKIPEDIA tokens used for training, and the y axis shows the number of adjective, noun, verb and NP types which are covered by the DM. The number of AN and NN types only includes those NPs which are contained in AMAZON. The number of noun types tends to grow linearly with the number of tokens, while the number of new adjective and verb types grows sub-linearly. The number of NPs covered by the DM grows rapidly until $x = 200M$, after which point it levels off.

Qualitatively, the new types entering the system before the threshold is exceeded are a mixture of different parts of speech and are what one might consider the core vocabulary of English. The majority of new entries after the critical threshold are proper nouns, which is understandable given the encyclopaedic nature of WIKIPEDIA. However, these do not necessarily occur in the labelled corpus. The number of composable NPs therefore grows slowly, which causes the small improvements in classification accuracy seen in Figure 4.6.

4.4.3 *Turian vectors*

This section answers the following research question:

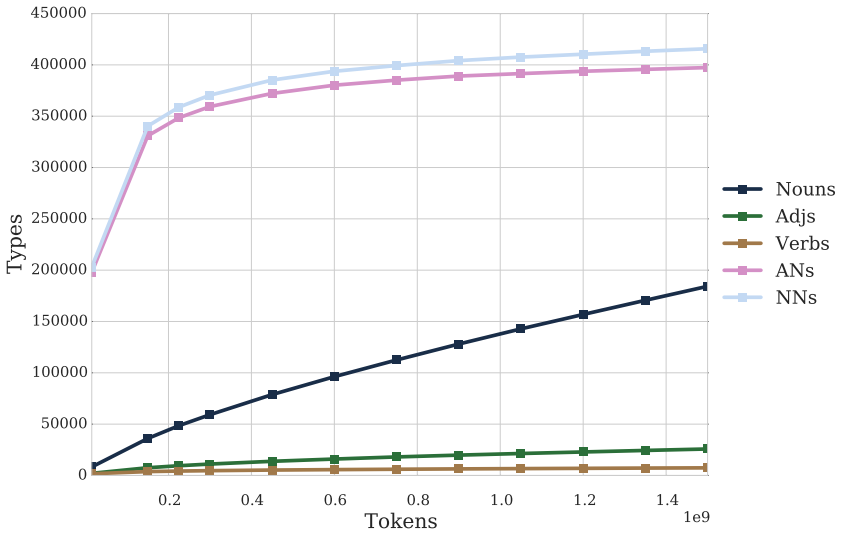


Figure 4.8: Type coverage of WORD2VEC as a function of number of tokens (from WIKIPEDIA) used for training.

Which NP composition algorithm performs best for TURIAN word embeddings?

TURIAN vectors are considered separately from count, GLOVE and WORD2VEC because they were built using a different labelled corpus, and should therefore not be compared directly to other models investigated in this chapter.

Empirical results are shown in Table 4.10. All compositional algorithms perform only marginally better than a random baseline. LEFT and RIGHT are not significantly better than random ($p = 0.04$ and $p = 0.5$ respectively).

Composer	Accuracy
Add*	0.24±0.01
Mult*	0.24±0.01
Socher*	0.24±0.01

Composer	Accuracy
Left	0.23±0.01
Right	0.22±0.01
RandN	0.22±0.01

Table 4.10: Accuracy of TURIAN embeddings. An asterisk indicates a model is significantly better than the random baseline.

Given that TURIAN word vectors with SOCHER composition achieved a state-of-the-art result at the Microsoft Research Paraphrase task (Socher et al., 2011), the question arises as to why they perform poorly in this evaluation. There are several possible explanations for this. First, the type coverage of TURIAN vectors is relatively lower compared to WORD2VEC or GLOVE. Because of the computational cost

associated with training the embeddings[11] there are only 50*K* word embeddings available. After lowercasing and lemmatisation only 33*K* word embeddings are left, some of which are for non-content words such as "which" and "to". The total number of nouns, verbs and adjectives is approximately 23*K*, 6*K* and 2*K* respectively. Overall, SOCHER is able to compose 801*K* of all document features contained in the labelled corpora, compared to over 1.3*M* for ADD or MULT composition over count vectors (Table 4.4).

Second, TURIAN vectors were trained on an unlabelled corpus which is an order of magnitude smaller than GIGAWORD, WIKIPEDIA and CWIKI. WORD2VEC models trained on tens of millions of tokens of unlabelled text (as opposed to hundreds of millions) also perform poorly (Figure 4.6). Assuming the performance of TURIAN improves similarly to that of WORD2VEC, one would expect the model to become significantly better with more data. However, training TURIAN embeddings on hundreds of millions of tokens would be computationally prohibitive.

Third, TURIAN's unigram embeddings subjectively appear to be of lower quality compared to WORD2VEC. Tables 4.11 and 4.12 show the nearest neighbours of 12 randomly selected unigram entries produced by TURIAN and a WORD2VEC model trained on 15% of WIKIPEDIA. While it seems to mix neighbours of different parts of speech (c.f. "torrential"), it returns more appropriate neighbours than TURIAN (c.f. "seize", "claim", "preventative"). The neighbours of "andrade" (a common Portuguese surname) returned by TURIAN are other surnames from different regions, while WORD2VEC has "souza", "vives" and "barros", which are all common in Portugal and Galicia.

Figure 4.9 shows a 2D t-SNE projection (Van Der Maaten and Hinton, 2008) of the vectors for several countries and their capitals, following Mikolov et al. (2013c, Figure 2). Mikolov et al. suggest that the semantic relation "is capital of" corresponds to a vector addition operation; the relative position of a country and its capital in semantic space is consistent across countries. Mikolov et al. (2013c, p 5) show that WORD2VEC is able to "learn implicitly the relationships between [concepts]". The same applies to GLOVE vectors (Figure 4.9). However, the relation learnt by WORD2VEC trained on GIGAWORD is considerably less consistent compared to WIKIPEDIA. TURIAN vectors perform even worse.

11 Turian et al. (2010, Section 6) report it took several weeks to train the embeddings over only 37*M* words. In contrast, WORD2VEC can be trained over 1.5*B* words of text in several hours on commodity hardware.

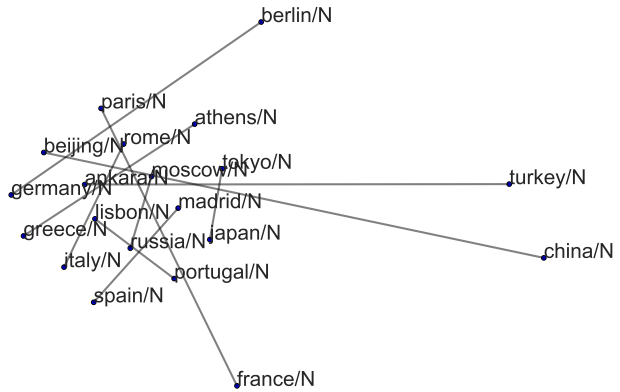
I hypothesise the good performance of SOCHER over TURIAN vectors at paraphrase identification is at least partially due to the nature of the problem. It is possible to achieve near state-of-the-art performance at that task by only considering lexical overlap features. The SMWBCUNF baseline of Rus et al. (2014) achieves a F1 score of .819, whereas Socher et al. (2011) score 0.836. It is therefore possible that having vectors for content words and a sensible means of combining those is more important than the quality of the vectors. Another possible explanation is that Socher et al. leverage information about the content of entire sentences, as opposed to focusing on noun phrases only.

Entry	Neighbours
andrade	aslam, rahman, aguirre, weinstein, chandra
giant	monopoly, retailer, group, utility, unit
seize	dissolve, renew, suppress, restore, donate
fundamental	inflationary, external, structural, speculative
affidavit	organization, affair, squad, essay, audience
claim	request, link, plan, challenge, drive
sikh	gambian, tutsi, bugging, zairean, rioting
rest	size, duration, bottom, end, depth
israel	beijing, washington, bonn, pakistan, russia
arrow	atlantic, mississippi, mall, iberian, caribbean
preventative	periodic, disgrace, compensatory, conclusive
torrential	intermittent, severe, persistent, gusty, planting

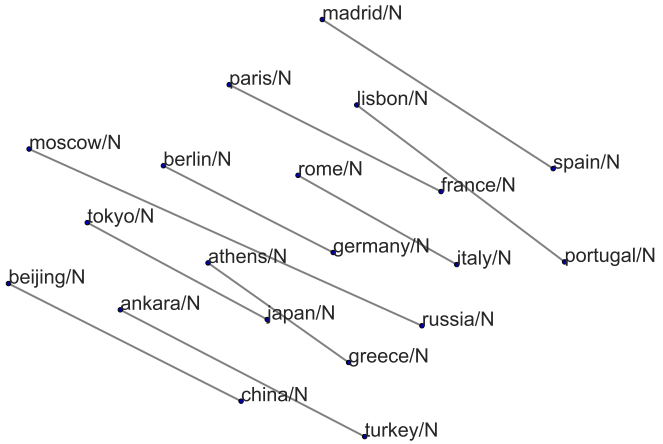
Table 4.11: TURIAN neighbours of randomly selected words

Entry	Neighbours
andrade	souza, camilo, vives, leopoldo, barros
giant	gigantic, legless, man-eating, serpentine, horned
seize	capture, re-take, attack, demoralise, crush
fundamental	principle, epistemic, objective, indeterminism
affidavit	testimony, al-arian, subpoena, demjanjuk, minkow
claim	allege, assert, believe, suggest, acknowledge
sikh	sikhs, hindus, shi'ite, hindu, shiite
rest	dais, keep, gorge, gibbet, parihaka
israel	iran, palestine, al-sadr, gush, salafi
arrow	bullet, quiver, gauntlet, upturned, sword
preventative	preventive, prophylactic, life-saving, high-risk
torrential	downpour, monsoonal, rainstorm, freezing, sleet

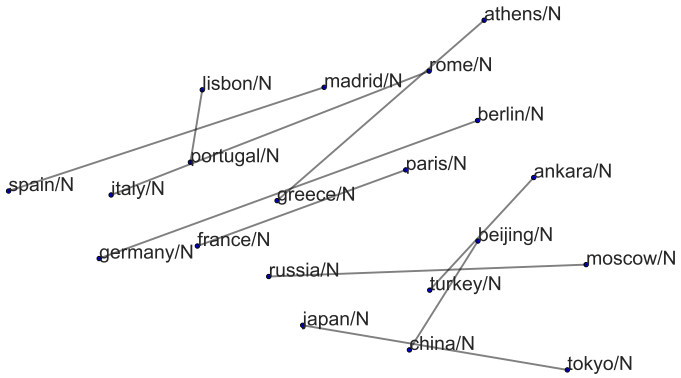
Table 4.12: WORD2VEC neighbours of randomly selected words



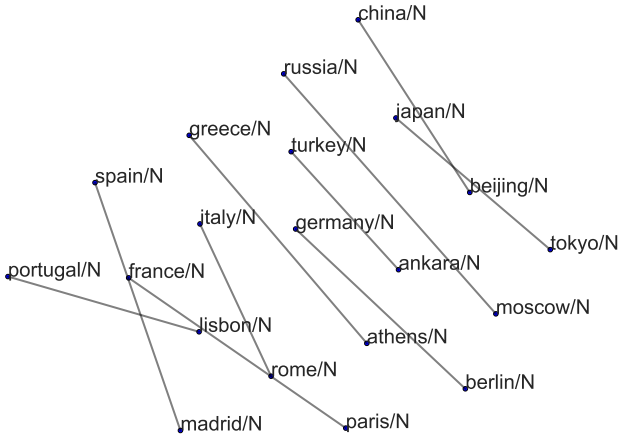
(a) TURIAN vectors



(b) GLOVE vectors on WIKIPEDIA



(c) WORD2VEC on GIGAWORD



(d) WORD2VEC on WIKIPEDIA

Figure 4.9: t-SNE embeddings of several counties and their capitals.

Summary of noun phrase experiments

The results above suggest that the determining factor in building word representations is data quality rather than quantity; in some cases only a small amount of unlabelled data is required to reach peak performance. This highlights the importance of training distributional models on clean unlabelled data that matches the domain where the model is applied. Another important factor is whether a compositional model can provide a vector for a document feature. Models with low coverage such as BARONI perform poorly in practice.

Neural algorithms for building single-word representations such as WORD2VEC and GLOVE perform better than counting-based ones regardless of what composition is used. Pointwise addition can match or exceed the utility of more sophisticated compositional proposals.

4.5 VERB PHRASE EXPERIMENTS

Section 4.4 focused on noun phrase composition. However, significant progress has also been made in the area of transitive sentence composition. This section evaluates several algorithms for composing transitive sentences such as "person buy car" or "child likes fruit", which were introduced in Sections 2.2.1 and 2.2.4. The question addressed by this section is:

> Which combination of word embeddings and VP composition algorithm performs best at topical document classification?

Experiment details

A total of six composition algorithms are considered. As before, ADD and MULT compose by addition and multiplication respectively. VERB is a baseline where each VP is represented by the vector of its head (the verb). COPYOBJ, FADD and FMULT are the category-theoretical models of Grefenstette and Sadrzadeh (2011)[12].

Category-theoretical composers are trained as described by Milajevs et al. (2014). A verb must be used as the head of at least 3 distinct

12 I also experimented with the multi-step regression model of Grefenstette et al. (2013) but found its coverage to be extremely poor. Results for that model are not shown because they are not significantly different from random.

VPs in an unlabelled corpus in order to be represented as a matrix. With that threshold matrices for 400–550 verbs can be learnt (Table 4.13). WIKIPEDIA has considerably better coverage than GIGAWORD (27*K* VPs versus 10*K*). The distribution of VPs used to learn a verb matrix $\overline{Verb}$ is highly skewed, with a mean of 50, a median of 10, a mode of 3 and a maximum of 2.8*K* for WIKIPEDIA. Some of the most frequent VP heads are "receive/V" (head of 859 distinct SVOs in WIKIPEDIA), "contain/V" (791), "win/V" (1371) and "include/V" (2898).

Unlabelled	Embeddings	SVOs	Mean	Verbs
WIKIPEDIA	GLOVE	27*K*	50	547
WIKIPEDIA	WORD2VEC	27*K*	50	542
GIGAWORD	WORD2VEC	10*K*	25	408

Table 4.13: Statistics for category-theoretical models. "SVOs" stands for total number of verb phrases which occur in both AMAZON and the respective unlabelled corpus (WIKIPEDIA or GIGAWORD). "Mean" stands for the mean number of SVOs per verb. "Verbs" is the total number of verb matrices learnt.

Similarly to noun phrase experiments in Section 4.4, lexical overlap between a VP and its replacements was not allowed. The document features extracted at train time are adjectives, nouns, verbs and VPs. At test time, only VP features are extracted. Adjectives and nouns are added at train time because verbs and verb phrases are relatively sparse (compared to noun phrases), so classification performance suffers. When accuracy is close to the random baseline, it is difficult to compare compositional algorithms. Please refer back to the discussion in Section 4.4.2 for an explanation.

Results

Only two of the NP experiments are replicated with VPs. Figure 4.10 is analogous to Figure 4.3 and considers the effect of domain and quantity of labelled data used to build word embeddings. Figure 4.11 is analogous to Table 4.8; it investigates the effect of the word embedding algorithm.

The overall pattern of results is similar across NPs and VPs — WIKIPEDIA is better than GIGAWORD, WORD2VEC is marginally better than GLOVE and count window vectors, adding more unlabelled data is beneficial and ADD is consistently the best composer. However,

these trends are less pronounced due to lower overall accuracy, which is in turn at least partially due to the low number of verb phrases in the AMAZON corpus. It is therefore difficult to draw definitive conclusions about the quality of composition as some combinations of word vectors and compositional algorithms perform close to the random baseline.

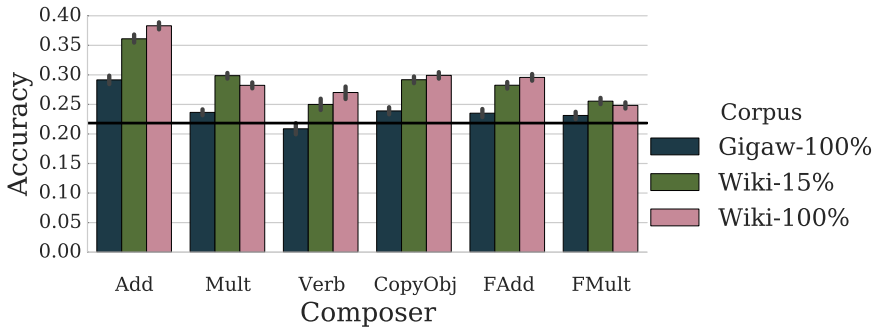


Figure 4.10: Verb phrases: effect of size and domain of unlabelled data (WORD2VEC embeddings).

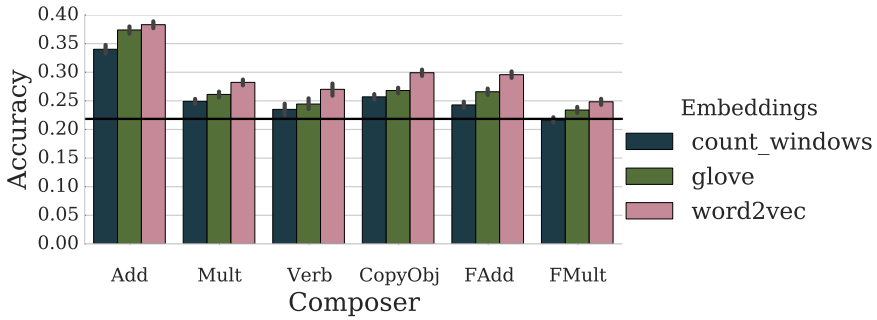


Figure 4.11: Verb phrases: effect of composition algorithm. Word embeddings trained on WIKIPEDIA.

Qualitative analysis and discussion

Table 4.14 shows the nearest neighbours of 14 randomly selected VPs with and without the lexical overlap constraint. Two trends are apparent. First, the neighbours are almost exclusively other VPs. Similarly, the neighbours of verbs (not shown) tend to be other verbs. A similar pattern is observed for ADD composition. Second, the lexical overlap constraint greatly affects neighbours. If overlap is allowed, the top neighbours exhibit little lexical diversity. It is very tempting to annotate phrases as similar because of they have words in common. However, without lexical overlap it is hard to intuitively reason if

the neighbours are appropriate. Our causal definitions of similarity (Section 3.1.2) do not generalise beyond unigrams. One cannot contextualise or visualise a sentence the way one would visualise a noun. This brings us back to the discussion of what space sentences live in and whether different grammatical types are comparable (Section 2.4.2).

Entry	Neighbours
seller sell bottle	government face obstacle , money buy brand
kit contain length	model feature strip , version use concentrate
japan face deterioration	taiwan introduce cut , korea maintain surplus
age create sense	child share information , time leave reply , parent save money
scene indicate coincidence	effect live footage , nature match character , presence spark laugh
lucy love backpack	No non-overlapping neighbours in top 75
seller sell bottle	seller sell item , seller sell bag , seller sell perfume
kit contain length	kit contain combination , kit contain duplicate ,
japan face deterioration	japan buy chip , japan accept application , japan face penalty
age create sense	age buy item , age acquire wisdom , age enjoy message
scene indicate coincidence	scene lack charm , scene lack feeling , scene demonstrate mentality
lucy love backpack	lizzie love flavor , kate love series , katie love food

Table 4.14: Nearest neighbours of randomly selected VPs with (top) and without (bottom) lexical overlap constraint. WORD2VEC embeddings trained on WIKIPEDIA and composed with COPYOBJ.

4.6 EFFECT OF TASK SETUP

This section investigates the effects of various parameters of the evaluation task introduced in Chapter 3 on the conclusions in this chapter. The following aspects are investigated:

EXTREME FEATURE EXPANSION In the framework above all document features, regardless of whether they have been seen in the training set, are replaced with nearest neighbours. This provides a more comprehensive evaluation as a wider selection of the distributional model is considered. Section 4.6.1 considers traditional feature expansion, where only features that do not occur in the training data are replaced with their nearest neighbours.

TEST-TIME FEATURES I choose to consider only NP or VP features at test time in order to focus on a model's ability to find good neighbours for phrases. Section 4.6.2 investigates the case where both NP and unigram features are included at test time.

LEXICAL OVERLAP A phrase and its replacements are not allowed to have a word in common, which prevents simple non-compositional algorithms from doing well in my framework, but also potentially reduces the accuracy of other models by placing further constraints on what can be a neighbour of a document feature. Section 4.6.3 considers the effect of allowing lexical overlap.

NUMBER OF REPLACEMENTS In all experiments above, a test-time document feature was replaced by its $k = 3$ nearest thesaurus neighbours. Section 4.6.4 studies the effect of varying the k parameter.

Results in this section are consistent with those presented in Section 4.4.

4.6.1 *Extreme feature expansion*

With the exception of the work of Lebret et al. (2013), distributional models tend to be used as an additional source of information in a machine learning pipeline rather than as the only source. This is typically done via feature expansion, either by replacing features with nearest neighbours (Andreas and Klein, 2014) or by appending distributed features to existing representations (Turian et al., 2010).

One non-standard feature of my framework is that all features at test time are replaced. In contrast, traditional feature expansion models leverage distributional information at test time to expand (replace) features that haven't been seen during training. Table 4.15 shows the performance of a document classifier with standard feature expansion.

Composer	Accuracy
Bag-of-NPs	0.85±0.01
Add	0.81±0.01
Right	0.81±0.01
Left	0.80±0.01
Mult	0.77±0.01

Table 4.15: Standard feature expansion results on Amazon dataset (WORD2VEC trained on WIKIPEDIA.)

First, the differences between different compositional algorithms are smaller than in Section 4.4. This is because in a large corpus such as AMAZON many features are encountered at both train and test time so test-time features are expanded infrequently. Accuracy is therefore largely determined by surface forms, while differences between composers are not emphasized.

Second, all composers in this section perform significantly worse than a non-distributional classifier. Preliminary experiments suggested distributional models are beneficial only when very little training data (less than 50 documents) is available, which agrees with the results of Huang and Yates (2009) and Andreas and Klein (2014). Another way of improving performance is to specialise embeddings to the task (Socher et al., 2013b). When training data is abundant performance gains due to distributional information are typically very modest or statistically insignificant (Turian et al., 2010; Andreas and Klein, 2014).

4.6.2 *Choice of test-time features*

A second important aspect of my framework is that both unigram and noun phrase document features are used for training, but only noun phrase features are extracted at test time. This means a NP feature can be replaced by another NP or by an unigram, but a unigram feature cannot be replaced (because it is not extracted). This puts emphasis

on the properties of composition algorithms rather than word embeddings. However, results would have been similar if unigram features were included at test time. Figure 4.12 compares the accuracy of a classifier with and without unigrams at test time. While overall accuracy figures are higher with extra features, the general pattern is identical to that of Figure 4.5. Vectors built on WIKIPEDIA are again of better quality than ones built on GIGAWORD, and adding more unlabelled data is also marginally beneficial.

It is worth noting that ADD and MULT benefit more from the inclusion of unigram features at test time. To understand this behaviour, consider the vectors involved in classifying the simple document "black cat". The three features that could be extracted are "black_cat", "black" and "cat". When only NP features are extracted, both ADD and RIGHT have a single document feature to work with. However, the two composers behave differently when unigram features are also added. With ADD composition, the system gains two extra vectors, $\overrightarrow{cat}$ and $\overrightarrow{black}$, which can be used to insert features into the document. With RIGHT, one of the unigram vectors is not really new, because $\overrightarrow{black\_cat} = \overrightarrow{cat}$ (by the definition of RIGHT). The classifier is therefore gaining less information from the addition of unigram features.

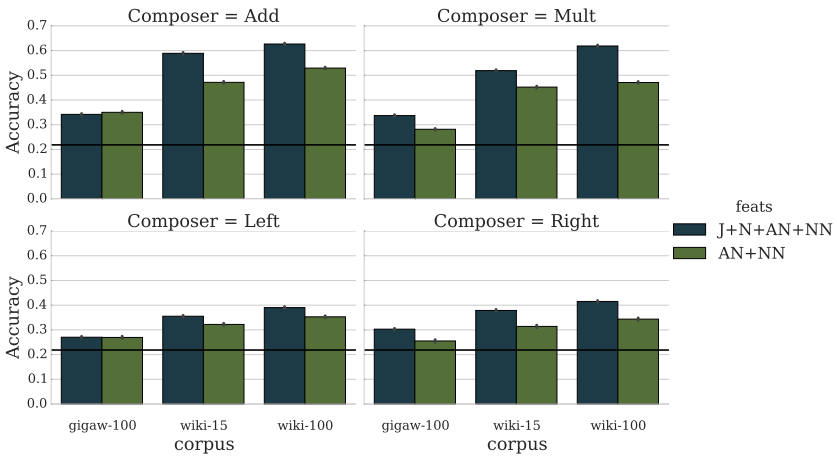


Figure 4.12: Features used at test time. Legend: "J"=adjective, "N"=nouns, "AN"=adjective-noun compound, "NN"= noun-noun compound. In all cases J+N+AN+NN features are used at train time.

4.6.3 *Lexical overlap*

Lexical overlap refers to the constraint that an entry may not have any words in common with its neighbours (Chapter 3). For instance,

no noun phrase that contains the words "black" or "cat" is permitted as a neighbour of "black cat". This ensures that it is not possible for a distributional model to succeed merely by operating on surface forms. Consider a pseudo compositional algorithm which for each NP returns a neighbour randomly selected out of the set of other NPs that share the same head. The algorithm need not consider the meaning of the NP, only the surface form of the words it contains. Such an algorithm would likely do well in a topic-based classification task such as AMAZON, because the head of a noun phrase characterises its topic well. For example, any kind of windscreen would be a good replacement for "cracked windscreen" in the context of differentiating documents related to cars from ones related to beauty products.

Figure 4.13 compares the performance of the classifier with and without the lexical overlap constraint. As expected, accuracy improves considerably when overlap is allowed, with ADD and MULT only 1.5 and 4.5 percentage points below the non-distributional classifier (significantly different, $p < 0.01$). However, the overall pattern remains the same — ADD outperforms MULT, and both are significantly better that LEFT and RIGHT.

The high performance of ADD when overlap is allowed can be explained by qualitative analysis. In many cases, the vector for a NP composed by addition remains very close to the vector of the head or modifier. For instance, the nearest neighbour of "black cat" in many models is "cat". A possible explanation is that the vector of one of the constituents of a NP is much longer than the vector of the other constituent and it dominates the resultant NP vector. This is especially common in count vectors where vector length is not regularised and can be addressed by explicit normalisation (Polajnar and Clark, 2014). As a result, a large number of NPs are replaced at test time by NPs that share one of their constituents. Without an explicit constraint to prevent this, the distributional model is effectively bypassed and what we arrive at is a bag-of-words classifier with unigram features, which is not surprisingly very close in accuracy to a bag-of-NPs one. This issue is discussed again in Section 4.8.

4.6.4 *Number of replacements*

The final system parameter investigated is k, the number of nearest neighbours inserted instead of a document feature at test time. In a typical feature expansion scenario, e. g. Andreas and Klein (2014), a

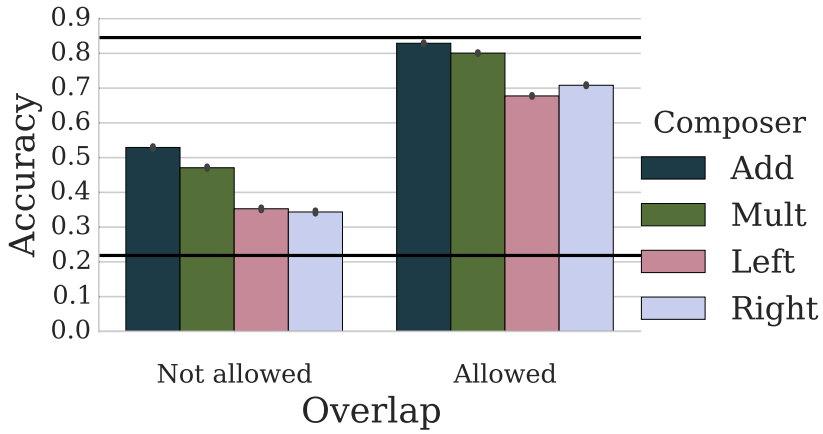


Figure 4.13: Effect of lexical overlap constraint (WORD2VEC trained on WIKIPEDIA). The upper and lower horizontal lines denote a bag-of-NPs and a random baseline respectively.

value of $k = 1$ is used. The motivation for using a different value is that sometimes the best neighbour[13] may not be ranked highest. Also, thesaurus entries often have multiple plausible neighbours. Using more than just the top neighbour provides a more comprehensive test environment, where entire neighbourhoods in semantic space are considered instead of points. In a distributional model of reasonable quality one would expect performance to increase initially with k as appropriate neighbours are added to the document vector, thus reducing sparsity and adding in good document features. As k is increased further, one would expect poor neighbours to start entering the system and accuracy to decrease as a result. In the extreme case where k is equal to the number of document features the accuracy of the classifier should match that of the random baseline. However, the running time of the evaluation procedure increases quickly with k, so it is impractical to verify this experimentally.

The empirical relation between k and accuracy for four compositional algorithms is show in Figure 4.14. Accuracy increases considerably until $k = 50$, after which it levels off. In the case of MULT accuracy begins to decrease rapidly as expected. However, the findings of Section 4.4 hold for a wide range of k values.

One perhaps surprising aspect of that result is that accuracy does not generally degrade considerably between $k = 50$ and $k = 1000$. This suggests neighbours number 50 to 1000 are equally appropriate. This can be explained qualitatively. Noun phrases, especially ones composed with LEFT or RIGHT, and to a lesser degree with ADD,

13 In the context of a task

tend to have a very large number of acceptable neighbours. I refer to this property as "composer productivity". For instance, suppose the nearest neighbours of the unigram "vw" are "mercedes", "audi" and "bmw". Any adjective followed by "mercedes" is a good neighbour of "red vw" when using RIGHT composition. This makes is easy to generate a large number of good neighbours for a NP by taking the Cartesian product of the set of all adjectives in the labelled data with the set of all unigram neighbours of the head. In practice, not all combinations would occur in the labelled data, but Figure 4.14 suggests there can be as many as 1000 appropriate neighbours of a NP.

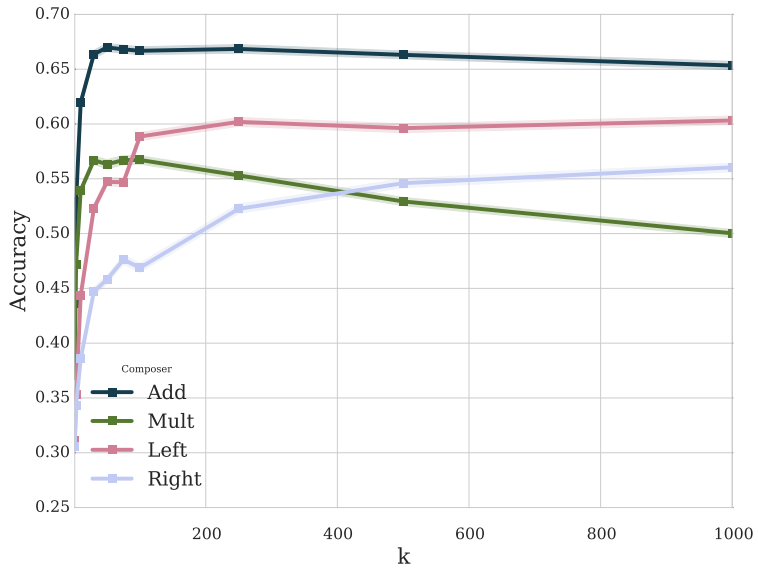


Figure 4.14: Effect of k parameter (WORD2VEC trained on WIKIPEDIA)

The effect of composer productivity can be negated by not considering NP features at test time, but instead using only adjectives and nouns. Results are shown in Figure 4.15. Accuracy increases initially, but begins to decrease rapidly after $k = 10$ as expected. This suggests each unigram has 10 appropriate neighbours on average for the AMAZON data set.

The k parameter also controls the stability of the evaluation task on small labelled data sets. As mentioned in Section 4.4.2, small labelled data sets contain a smaller number of train-time document features, which cause the evaluation task to be less reliable than it is for larger data sets. Figure 4.16 shows a noise validation like those shown in Figure 3.2 and 4.2 at the small R2 corpus. Note when $k = 3$ accuracy is not a monotonically decreasing function of noise, which is undesir-

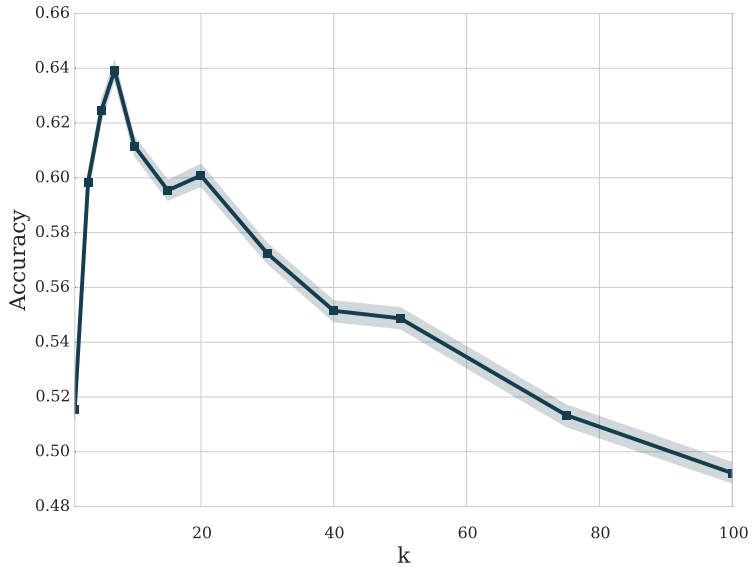
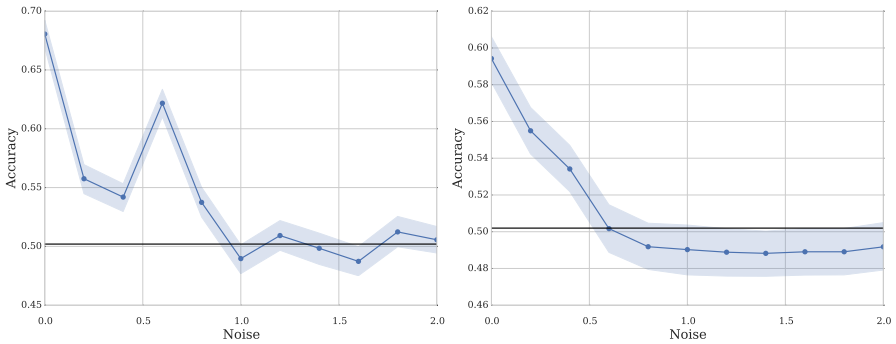


Figure 4.15: Effect of k parameter with unigram features only (WORD2VEC trained on WIKIPEDIA)

able. However, when the value of $k = 30$ is increased, accuracy begins to degrade as expected as noise is added. The variance of the measurement is still considerable, which suggests smaller corpora may be inappropriate for the framework (see Section 5.2). The differences between $k = 3$ and $k = 30$ suggest including multiple replacements per document feature helps overcome the sparsity induced by using a small labelled corpus, which results in a more reliable degradation of accuracy as noise is added.



(a) $k = 3$ (b) $k = 30$

Figure 4.16: Noise validation on R2 corpus

Summary of Task setup section

All results of Section 4.4 are consistent with the figures reported in this section. Embeddings trained on WIKIPEDIA outperform those trained on GIGAWORD, and ADD composition is significantly better than other competitors. This holds for a wide range of k values regardless of whether or not unigram features are included at test time.

4.7 EXPLOITING MULTIPLE MODELS

The previous sections considered different ways of learning distributional representations, either from the same data set or from a different one. The primary goal of those experiments was to identify the single best distributional model. While some models perform better than others, I hypothesise that the semantic representations learnt by each DM are in fact complementary. Each DM can be thought of as a distinct view into the same domain. The knowledge of an ensemble of models may therefore exceed that of any single model in the ensemble. This is related to boosting (Freund and Schapire, 1997) in the machine learning literature. Another related field is multi-view learning, where qualitatively different feature sets (variable groups) are combined to improve performance (Xu et al., 2013). For example, a document can be represented by 1) information describing its content such as bag-of-words or an LDA vector, 2) meta-information such as author name, affiliation and human-provided keywords, and 3) a citation network that represents related work. While views are incompatible[14], each of them may offer useful insight into the problem space.

This section investigates an instantiation of the multi-view idea in the context of word embeddings. The following questions are considered:

1. Is the utility of views trained on a different equally sized sample of the same unlabelled corpus comparable (Section 4.7.1)?

2. Can the information obtained from multiple views be used to improve classification accuracy (Section 4.7.2)?

Results suggest that thesauri built from different samples of the same corpus achieve approximately equal classification accuracy when

14 In the sense that embeddings from one view cannot be compared to embeddings from another view.

Entry	ID	Neighbours
stalin/N	1	lenin/N, trotsky/N, bukharin/N, brezhnev/N
	2	lenin/N, hitler/N, goebbels/N, kerensky/N
	3	lenin/N, goebbels/N, neurath/N, kerensky/N
microsoft/N	1	ibm/N, ms-do/N, linux/N, netware/N
	2	ibm/N, symbian/N, sdk/N, unix/N
	3	linux/N, ms-do/N, ibm/N, unix/N
fugitive/N	1	fugitive/J, ex-slave/N, manumission/N
	2	escapee/N, fugitive/J, convict/N, ill-treatment/N
	3	convict/N, non-combatant/N, infiltrator/N
car/N	1	truck/N, motorbike/N, automobile/N
	2	truck/N, motorbike/N, rear-engined/J
	3	truck/N, motorbike/N, bike/N, lorry/N
paris/N	1	marseille/N, brussels/N, saint-cloud/N
	2	brussels/N, aix-en-provence/N, angers/N
	3	brussels/N, strasbourg/N, dijon/N, nice/N
smith/N	1	mckay/N, robinson/N, fuller/N, moore/N
	2	taylor/N, jones/N, moore/N, bennett/N
	3	taylor/N, jones/N, foster/N, miller/N

Table 4.16: Neighbours of several randomly selected unigrams in three different WORD2VEC models. Each row shows the neighbours of one run with identical parameter settings.

considered in isolation. However, a combination of multiple views can outperform each individual view. I introduce a novel way of re-ordering the nearest neighbours of an entry to rank more appropriate neighbours higher.

4.7.1 *Comparing views*

Each unlabelled corpus has unique characteristics that affect the quality of the distributional models trained on it (Section 4.4.2). This is also true for multiple distinct views of the same corpus. Consider a small sample of three WORD2VEC models, trained with identical parameter settings on different 15% samples of WIKIPEDIA (Table 4.16). While many (but not all) of the nearest neighbours of an entry are the same, their order is often permuted. A natural question to

ask, which is seldom considered in the literature, is whether these differences affect the utility of a distributional model.

The degree to which neighbour sets differ can be evaluated numerically using the Dice coefficient, which is defined for two sets $\mathcal{A}$ and $\mathcal{B}$ as[15]:

$$d(\mathcal{A}, \mathcal{B}) = \frac{2\,|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A}| + |\mathcal{B}|}$$

For each pair of the three WORD2VEC models trained on 15% of WIKIPEDIA, I plot the distribution of Dice coefficients between the top 100 neighbours of 5000 randomly selected unigrams (Figure 4.17. The sample is selected such that all three views have a vector for each entry in the sample. Overlap is low, with a mean of less than 0.3. This suggests each of the three models is producing a different neighbour set for each entry in the sample.

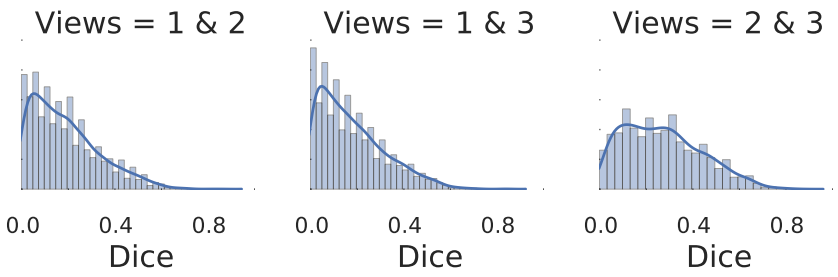


Figure 4.17: Distribution of Dice coefficient of top 100 neighbours of 5000 randomly sampled unigram entries across views.

The result raises the question if one of the three models is better than the others. Qualitatively, the answer is no (Table 4.16). While the embeddings (and therefore neighbours) learnt by WORD2VEC do vary between views, visual inspection of nearest neighbours suggests all three views learn equally appropriate word representations. The quality of each view can be measured using the framework described in Chapter 3. The embeddings learnt by the three models are composed with the four core composers and evaluated at the AMAZON corpus. There are only small differences between the utility of the vectors of each repetition (Figure 4.18). The first two views contain unigrams of comparable quality (no significant difference), whereas the third view in Figure 4.17 is significantly ($p < 0.01$) worse by about

15 Note this formulation ignores order information and only considers if neighbour sets contain the same entries. An alternative measure that does not suffer from that drawback is Spearman's rank correlation coefficient. However, it is only applicable in cases where all the neighbours of an entry are the same but in a different order, which is not the case here.

2 percentage points. These results suggest the utility of the top neighbours for the AMAZON task is comparable, which renders their ordering unimportant. This result is in line with the finding in Section 4.6.4, which demonstrated that each NP has a large number of equally appropriate neighbours (in the context of a particular classification task).

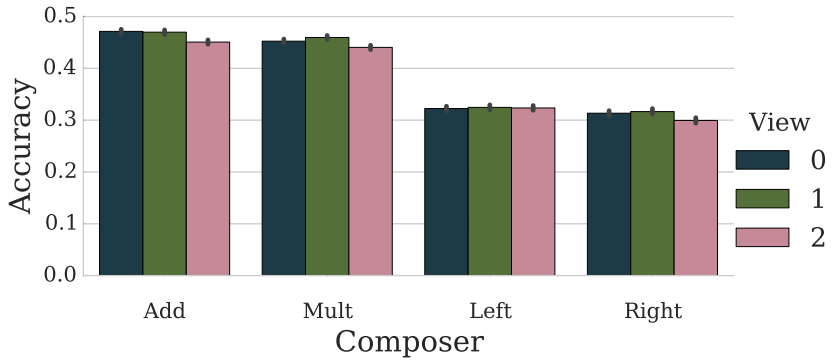


Figure 4.18: Performance of three views into the same domain (WORD2VEC, 15% WIKIPEDIA)

4.7.2 *Combining views*

Building on the idea that different views of the data contain complementary information, this sections presents a comparison of two methods for improving neighbour quality by using information contained in multiple views. The first is addition of entry vectors across views. There is empirical evidence in the literature that summing multiple vector representations of the same word is beneficial (Pennington et al., 2014, Section 4.2). The theoretical reasons why this might be valuable have not been studied extensively. It has been suggested that summation may reduce the variance of classifiers based on neural distributed representations (Ciresan et al., 2012, Section 2.5). Here I use averaging instead of summation, which effectively scales all word vectors by a constant factor and does not affect the ordering of nearest neighbours.

The second method of obtaining better neighbours is novel and is based on the observation that high-quality neighbours tend to be consistent across repeated runs of the same model. Table 4.17 shows several entries that illustrate that pattern. Consistency is again measured using the Dice coefficient of the first 100 neighbours of each entry, averaged over all pairs of views. For brevity only the first up

to four neighbours from three independent views are shown. The subjective quality of neighbours appears to correlate with the Dice coefficient. I hypothesise the reason for this is that if the set of contexts an entry appears in is consistent, then WORD2VEC converges on similar word clusters regardless of how the embeddings are initialised or what data is used for training. Conversely, if the "signal" is not strong enough, the learnt word clusters are different each time. The number of independently trained distributional models is a free parameter — I experiment with values from 2 to 5.

Multiple views can also be understood in the context of the results in Section 4.4.2. Recall a small batch of unlabelled data containing only about 30% of WIKIPEDIA is sufficient to obtain good accuracy in the classification task. The remaining 70% can be used as an extra two batches of 35%, each sufficient to obtain good vectors, instead of as one batch of 70%, which improves little over the first small batch.

Entry	Dice	Neighbours
balkans/N	0.12	transcaucasia/N, transoxiana/N, chalukyas/N south-eastern/N, kastoria/N, interbellum/N transcaucasia/N, apennines/UNK, peninsular/J
inborn/J	0.13	empathic/J, self-efficacy/N, depersonalization/N empathic/J, self-concept/N, uninhibited/J nonverbal/J, salience/N, nde/N, self-harm/N
lesbian/J	0.49	gay/J, lesbian/N, transgender/J, bisexual/J gay/J, transgender/J, transgendered/J gay/J, transgendered/J, bisexual/N, transgender/N
essay/N	0.65	pamphlet/N, monograph/N, poem/N, book/N book/N, pamphlet/N, poem/N, monograph/N pamphlet/N, book/N, poem/N, monograph/N

Table 4.17: Neighbours of several unigrams in three identical WORD2VEC models and mean Dice coefficient of the neighbour sets.

The intuition above can be used for feature expansion by combining and reordering the neighbours of an entry provided by each view. Neighbour lists are merged and sorted by 1) decreasing order of how many of the lists a neighbour occurs in, 2) increasing sum of ranks in the original lists, and 3) increasing sum of Euclidean distances to the entry in all lists. Neighbours are returned in this sorted order.

The neighbour reordering algorithm above is best illustrated with an example. Three neighbour lists for a hypothetical entry are shown in Table 4.18. The lists are merged into the set $\{a, b, c, d, e, f\}$, which is sorted according the criteria above. This yields the neighbour list in the first columns of Table 4.19. The best neighbour is a and the worst one is e.

Table 4.20 shows the nearest neighbours of "mercedes/N" returned by three WORD2VEC models (views). The original neighbours are a mixture of other car manufacturers ("citroen/N", "lancia/N"), car models ("gallardo/N", "tipo/N"), racing teams ("lola/N", "panoz/N") and race drivers ("alesi/N", "nuvolari/N"). The reordered list of neighbours (bottom row) is more coherent and contains only names of other car manufacturers or models.

View	Neigh 1	Neigh 2	Neigh 3	Neigh 4
1	a (.1)	b (0.3)	c (0.3)	d (0.4)
2	f (.1)	a (0.2)	e (0.4)	c (0.5)
3	d (.1)	b (0.4)	c (0.5)	a (0.6)

Table 4.18: Three hypothetical neighbour lists for an entry. The numbers in brackets indicate the Euclidean distance between each neighbour and the hypothetical entry.

Neighbour	Contained in	Σ rank	Σ euclidean
a	3	7	.9
c	3	10	1.3
d	2	5	.5
b	2	5	.7
f	1	1	.1
e	1	3	.4

Table 4.19: Reordered list of neighbours for the entry in Table 4.18.

View	Neigh 1	Neigh 2	Neigh 3	Neigh 4	Neigh 5
1	ascari/N	citroen/N	sauber/N	gallardo/N	lola/N
2	ferrari/N	lola/N	ducati/N	tipo/N	panoz/N
3	lancia/N	carrera/N	ferrari/N	alesi/N	nuvolari/N
-	ferrari/N	lancia/N	citroen/N	carrera/N	maserati/N

Table 4.20: Original and reordered lists of neighbours for "mercedes/N".

The effect of the two methods for improving embedding quality on the accuracy of a classifier is shown in Figure 4.19. Vector averaging results in a small decrease in accuracy. Neighbour reordering consistently and significantly improves accuracy by as much as 4 percentage points when k (number of replacements for each test-time feature, Section 4.6.4) is low. However, reordering does not significantly improve accuracy when k is high.

A possible explanation for this is that when $k = 3$, a document feature is less likely to be returned as a neighbour by multiple views by chance. Those few features that do occur in multiple views are likely to be of high quality; the reordering method pushes these to the top of the list, resulting in improved performance. When $k = 30$ there is considerably more overlap between the neighbour lists returned by each view, so the first sorting heuristic becomes less useful.

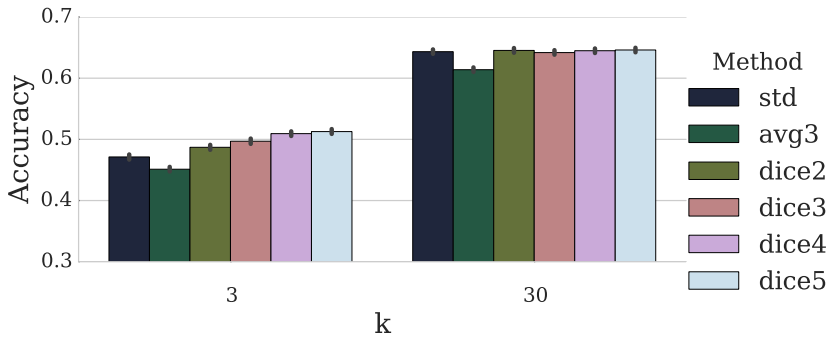


Figure 4.19: Effect of rebuilding embeddings (WORD2VEC trained on 15% of WIKIPEDIA). "std" stands for the approach from Chapter 3, where only a single view is used to obtain nearest neighbours. "avg3" stands for averaging vectors for each entry across three repetitions. "dice$N$" stands for consistency-based reordering using N WORD2VEC models trained on different samples of WIKIPEDIA.

Section summary and further work

In this section, distributional models trained on different samples of the same data were presented as distinct views into the same domain. Although views are qualitatively different, their individual utility is comparable. However, a combination of multiple views based on neighbour reordering can achieve higher classification accuracy than each individual view alone. In contrast, averaging of embeddings does not improve performance.

The definition of what constitutes a view considered here is somewhat narrow. It would be interesting to experiment with combining views built with different distributional or non-distributional (Section 2.4.1) models, or on different data sets. However, this work is beyond the scope of the present thesis.

4.8 ALTERNATIVE DOCUMENT REPRESENTATION

This section explores a different way of using the information contained in distributional word representations for document classification. Up to now, documents were encoded as a bag of words or phrases and distribution information entered the system in the form of discrete nearest neighbours of each word or phrase. In this section, a document is represented as a distribution over phrase clusters, which are built using a distributional model (Lebret and Collobert, 2014).

Description of method

The basis of the proposal is Vector Quantisation (VQ), which was originally developed for lossy compression in the signal processing literature (Gray, 1984). The key intuition is that a continuous vector space can be divided into a fixed number of "prototypes". As a result, entire areas of the space, which contain an infinite number of points, can be represented by the nearest prototype. The set of prototypes is commonly referred to as the "code book". The size of the code book controls the balance between compression ratio and information loss. Without any prior knowledge of the data being encoded, the prototype vectors can be chosen uniformly from the entire space. If knowledge of the domain is available, prototypes can be chosen in areas where the data is dense so as to maximise the number of points

that map to the same prototype, which improves compression and minimises data loss. This is typically achieved through clustering.

In natural language processing, VQ has been used to reduce the size of term-document matrices in feature classification (Baker and McCallum, 1998; Lebret and Collobert, 2014). A necessary first step is to represent the discrete units used in document classification, such as words and phrases, as vectors. Baker and McCallum (1998) achieve this by viewing a word as a probability distribution (vector) over class labels. The set of words in a labelled corpus can be clustered; the learnt cluster centroids form the code book. A document is represented as a distribution over centroids instead of as a distribution over words. That representation is significantly more compact and only results in a small loss of accuracy. A disadvantage of this approach is that a labelled corpus is required to build word vectors. Also, because labelled corpora are typically small in size, most phrases occur very infrequently and may have skewed or uninformative vectors. The work of Baker and McCallum is therefore practically limited to unigram features only.

Lebret and Collobert (2014) extend the work of Baker and McCallum by constructing word representations in an unsupervised way using a distributional model. This enables the use of phrase representations via a composition operation. Lebret and Collobert use averaging, which has been shown to perform well in practice (Mikolov et al., 2013a). N-gram document features capture more information about the content of a document, resulting in increased performance.

This section builds on the work of Lebret and Collobert by exploring a wider range of composition algorithms. The focus is on noun phrases only in order for results to be comparable to those in the rest of this chapter. The distributed representations for all adjectives, nouns and NPs contained in AMAZON are clustered using the mini-batch k-means algorithm with $k \in [100, 200, \ldots, 2000]$. Documents are converted into vectors of dimensionality k as follows. Each cluster is assigned a unique integer identifier from 0 to k. For each document feature f and its corresponding cluster identifier i, the i-th position of the k-dimensional document vector is incremented.

Empirical evaluation

The VQ document representation is also subjected to a noise validation by adding uniform random values $\mathcal{U}(-n, n)$ to each distribu-

tional vector prior to clustering. Results are shown in Figure 4.20. As with the extreme feature expansion document representation, VQ exhibits the desirable property that accuracy tends toward that of a random baseline as more noise is added.

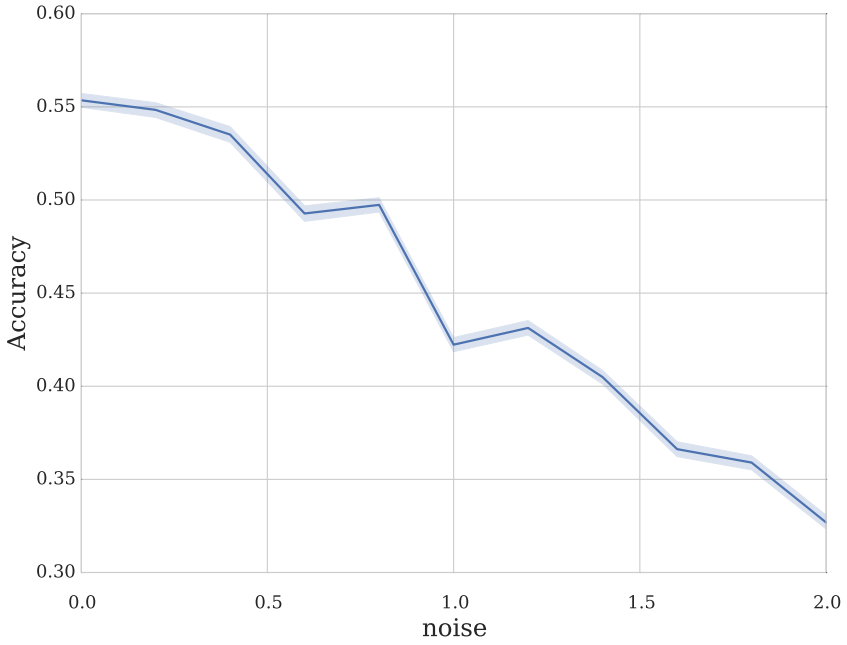


Figure 4.20: Noise validation of VQ task. WORD2VEC trained on GIGAWORD.

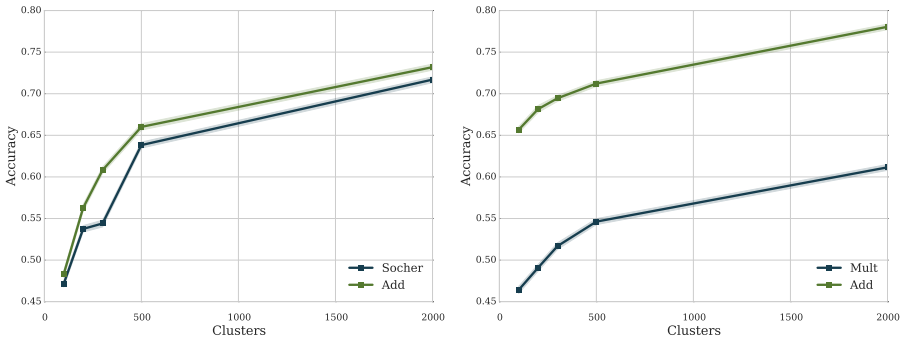
Table 4.21 shows the performance of several combinations of word embedding algorithms and unlabelled corpora, when composed with ADD and clustered with $k = 100$ clusters. As before, TURIAN vectors perform worst, but the gap between them and other methods is considerably smaller. CWIKI vectors are marginally better than WIKIPEDIA ones, which in turn are considerably better than GIGAWORD vectors. WORD2VEC is outperformed by GLOVE by around 4 percentage points ($p < 0.01$).

Overall, accuracy is significantly higher when using VQ document representation compared to extreme feature expansion. The best models in Table 4.21 approach an accuracy of 0.7, whereas the same vectors with feature expansion only reach 0.55 without lexical overlap (Figure 4.5) and 0.8 with lexical overlap (Figure 4.13). However, the VQ models shown here should only be directly compared to models where lexical overlap is permitted. This is because the lexical overlap constraint is difficult to encode into the objective function of k-means, so lexically overlapping entries can be placed in the same cluster.

Embeddings	Corpus	Accuracy
TURIAN	-	0.483±0.005
WORD2VEC	GIGAWORD	0.553±0.005
WORD2VEC	WIKIPEDIA	0.656±0.005
WORD2VEC	CWIKI	0.674±0.005
GLOVE	WIKIPEDIA	0.695±0.005

Table 4.21: Different corpora and embedding algorithms with ADD composition at AMAZON corpus, 100 clusters.

Figure 4.21 shows a comparison of ADD and MULT composition as well as the effect of the number of clusters k. As before, ADD significantly outperforms MULT. However TURIAN vectors at $k = 2000$ are almost on par with WORD2VEC (0.73 vs 0.77). In all cases accuracy increases considerably with k, which suggests being able to make more fine-grained distinctions is beneficial for the AMAZON task. Unfortunately, the expected running time of k-means scales linearly with k, so it is impractical to increase k further. It should be noted the limiting case where $k = \|V\| \approx 1.6 \times 10^6$, i. e. each feature is placed in its own cluster, is equivalent to the non-distributional bag-of-words approach, which achieves an accuracy of 0.84. From a practical perspective the best models shown here (WORD2VEC and GLOVE with ADD composition) achieve a good trade-off between running time and accuracy. These results are in line with those of Baker et al. and Lebret and Collobert in that the VQ model achieves slightly lower accuracy than a full-blown bag-of-words model using a more compact document representation.



(a) TURIAN word vectors (b) WORD2VEC, WIKIPEDIA

Figure 4.21: Effect of number of clusters on performance of VQ classifier.

Figure 4.22 shows the learning curve of WORD2VEC using VQ document representation. Similarly to the feature expansion experiments described above, peak accuracy is reached with vectors built on only a small proportion of all available unlabelled data.

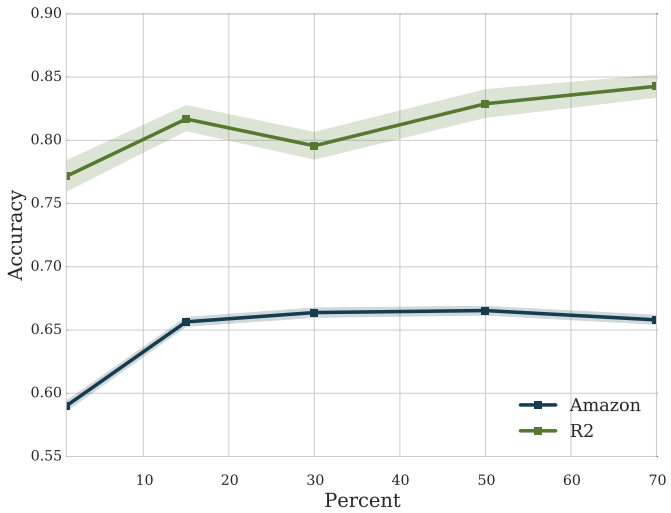


Figure 4.22: WORD2VEC learning curve for VQ document representation (ADD composition, 100 clusters, AMAZON/R2 labelled corpus)

Qualitative analysis

The high accuracy of VQ classifiers can be explained by qualitative analysis. I visually inspected the clusters that form when a WORD2VEC model with ADD composition trained on WIKIPEDIA is clustered into $k = 2000$ clusters. First, the clusters are large, with a mean size of 718. They tend to contain a large proportion of noun phrases that share a single unigram, which I will refer to as the "dominant" word of the cluster. For example, the cluster that contains the NP "bad guy" has a total of 934 entries, 726 out of which contain the modifier "bad". The entire cluster is dominated by "bad". However, in the same cluster there are 84 NPs containing the word "good". In another cluster 1008 out of 1621 contain the word "good" (but also 105 NPs contain "decent", 149 contain "excellent" and 42 contain "perfect". In a third cluster, 830 out of 1328 entries contain the word "movie". Interestingly, the largest cluster (5752 entries) appears to contain only proper nouns. Similar patterns are observed for other combinations of word embeddings and composition algorithm, including TURIAN vectors with SOCHER composition.

It appears that while some clusters are sensible, others contain antonyms ("amazing support", "awful standard" and "boring majority" are in the same cluster). However, the dominant factor that determines the content of a cluster is the fact that lexical overlap between entries in the same cluster is permitted. This allows large clusters dominated by a single word to form. We have already seen this property of lexical overlap in Section 4.6.4, where an entry was shown to have up to a thousand "sensible" neighbours because of lexical overlap.

Consider the effect of clusters dominated by a single word on classifier accuracy. Recall that each dimension in the VQ document representation corresponds to a cluster. In turn, each cluster effectively (but not perfectly) corresponds to a single dominant word. The document representation is therefore an approximate version of a bag-of-words model with as many features as there are clusters. In light of this, it is understandable why the performance of VQ is only slightly lower than that of the bag-of-words models in Section 4.2. The two models are comparable, but VQ introduces some error due to having to compress a document to a comparatively low-dimensional vector. This also accounts for the observed improvement in accuracy as the number of clusters is increased.

Discussion: Lexical overlap and Lexical Diversity

Lexical diversity is an issue with phrase-level similarity. It is very tempting to annotate phrases as similar because they have words in common (Table 4.14). The compositional models investigated here are not good at generating lexically diverse paraphrases — nearest neighbours of a phrasal entry often overlap lexically with it. In this thesis, I introduced an explicit constraint to circumvent that property. It could be argued this is unfair as composers are forced to discard as many as several hundred of their top neighbours. On the other hand, if one thinks of neighbours as possible in-context substitutions or paraphrases, then it is desirable that neighbours are lexically diverse. Replacing one of the words in a (compositional) phrase does not produce a paraphrase for the entire phrase, but just for the word that was replaced. In light of that, it might be beneficial to require neighbours to be lexically diverse.

The results above also provide evidence to support my decision to disallow lexical overlap. If overlap is allowed, one is not fully evaluat-

ing the quality of a distributional model. The role of the DM is largely reduced to something akin to feature selection ("selecting" the dominant word in each cluster); the ability of a DM to place semantically similar phrases in the same cluster is not truly evaluated.

Summary of VQ section

This section explored an alternative document representation scheme based on vector quantisation. Results are generally consistent with those in previous sections — ADD composition outperforms MULT, and embeddings trained on CWIKI are best for the AMAZON corpus. Only a small amount of unlabelled data is needed to reach peak performance. Upon closer inspection, the VQ model was found to be remarkably similar to a bag-of-words model because the lexical overlap constraint is hard to enforce in the clustering step of VQ.

4.9 SENTIMENT ANALYSIS

This section attempts to answer the following question:

> Which combination of word embeddings and composition algorithm performs best for the MAAS and MR sentiment corpora?

Table 4.22 shows the accuracy of the best models at AMAZON when evaluated at sentiment analysis. Two labelled corpora and two ways of building document vectors are considered. Due to the small size of MR, variance is considerably large than it is at MAAS or AMAZON. Extreme feature expansion performs worse than VQ because VQ effectively permits lexical overlap. As before, increasing the number of clusters for VQ improves accuracy. Overall, accuracy is considerably lower than reported in the literature for these data sets, which is 0.87–0.88 for MAAS and 0.87–0.9 for MR (Maas et al., 2011; Wang and Manning, 2012). However, those figures are for models which are not constrained to noun phrases only. A non-distributional model with NP features at test time scores 0.72 at MAAS (Table 4.7), which is close to the best distributional model. Using NP features for sentiment analysis is suboptimal as the classifier will not have a notion of negation, regardless of how good its NP representations are.

Labelled	Method	Embeddings	Unlabelled	Accuracy
MR	EFE	WORD2VEC	CWIKI	0.611±0.058
MAAS	EFE	WORD2VEC	CWIKI	0.566±0.005
MAAS	VQ-100	GLOVE	WIKIPEDIA	0.628±0.01
MAAS	VQ-100	WORD2VEC	WIKIPEDIA	0.622±0.01
MAAS	VQ-100	WORD2VEC	CWIKI	0.621±0.01
MAAS	VQ-500	WORD2VEC	WIKIPEDIA	0.662±0.01
MAAS	VQ-2000	WORD2VEC	WIKIPEDIA	0.688±0.01

Table 4.22: Sentiment results. VQ-N stands for vector quantisation with N clusters (Section 4.8), and EFE stands for extreme feature expansion (Section 3.3).

Discussion

Sentiment analysis is a qualitatively different problem to topic-based document classification. Unlike AMAZON, good performance at MAAS and MR requires a model of the fine-grained propositional structure of a document, including phenomena such as negation ("good" vs "not good") and gradation ("good" vs "very good"). It is well known that general-purpose distributional models such as WORD2VEC do not excel at sentiment analysis because of their tendency to learn similar vectors for antonyms. Results above confirm this experimentally. A growing body of literature is being developed that specialises embeddings for sentiment (Socher et al., 2013b; Kim, 2014). Sentiment analysis is tackled here in my framework to demonstrate that the framework allows one to easily test a distributional model against a different notion of similarity. I do not claim non-specialised distributional models are appropriate in the context of sentiment analysis.

5

CONCLUSION

This chapter summarises the main findings of the thesis, outlines the limitations of the work described in previous chapters and outlines directions for future work.

5.1 SUMMARY OF RESULTS

As distributional semantic models have gained popularity in recent years, the importance of good evaluation methods has increased. However, the vast majority of current work uses intrinsic word similarity evaluation. This thesis demonstrated empirically that three out of the four word similarity data sets considered are unable to reliably detect the presence of random noise in word embeddings. Further, I argued that in addition to the practical problems above, the word similarity task is fundamentally inappropriate because it assumes the existence of a single notion of similarity, which is independent of a particular application.

Motivated by these observations, I proposed a novel extrinsic framework based on distributional feature expansion applied to document classification (DC). In that framework, a document classifier is trained with a bag-of-phrases representation. At test time, all document features are replaced with their nearest neighbour according to a distributional model. A classifier is not allowed to access the actual contents of a test document, but can only view it through the prism of a DM. The accuracy of the classifier when a particular DM is used provides a direct measure of the DM's quality.

I presented an evaluation of counting word embeddings algorithms and the neural models of Collobert and Weston (2008), Mikolov et al. (2013a) and Pennington et al. (2014). These were combined with pointwise composition models as well as with the models of Grefenstette and Sadrzadeh (2011) and Socher et al. (2011). Results suggest the determining factor in building word representations is data quality rather than quantity. In some cases only a small amount of unlabelled data is required to reach peak performance. Neural word representation algorithms perform better than counting-based ones regardless

of what composition is used, but simple pointwise addition consistently outperforms more sophisticated competitors.

5.2 LIMITATIONS OF FRAMEWORK

The proposed evaluation framework suffers from the inherent sparsity problem of natural language research. This is because the set of candidate neighbours for a target document feature is heavily constrained:

- All candidates must be contained in the training portion of the labelled set, which can be very small[1]. This is a known issue of models making use of feature expansion.
- Candidates must not share any unigrams with their target document feature.
- Syntactic typing (postulating that, for example, verb phrases and nouns are not comparable) imposes further constraints on the neighbours of a document feature.

As a result, the list of candidate neighbours for a test-time document feature may be empty, or it may only contain low-quality neighbours, which would not have been selected in an unconstrained scenario. The problem can be particularly pronounced with small labelled data sets. In these, classification accuracy can vary by as much as 15 percentage points over cross-validation, which makes it hard to draw definitive conclusions about the quality of distributional models.

The framework therefore works best with large labelled corpora. This issue is what prompted the use of larger corpora such as AMAZON and MAAS in Chapter 4. Because these are an order of magnitude larger than the next largest corpus I experimented with, results are more stable.

Another criticism of my evaluation framework concerns its focus on noun and verb phrases. State-of-the-art representation learning algorithms explicitly or implicitly exploit latent lexical and syntactic information to abstract, relate and ultimately classify documents. Such systems perform well because there is a lot of information contained in a document that can be leveraged. By removing all but NPs or VPs, my framework sacrifices a lot of that information. The question

1 Contrast that to the work of Dinu and Baroni (2014), whose set of candidate neighbours for a given adjective-noun compound is the Cartesian product of all words in the vocabulary.

therefore remains if high-quality NP and VP composition will in fact generalise and translate to an accurate full-blown classifier.

A third assumption of my framework is that embeddings, composition and classification are separate. Recent work tends to learn these operations jointly and to specialise for a particular task, which results in increased performance. However, such models are treated as a black box and therefore hard to study. Keeping composition separate lets one import linguistic intuition into the process more easily.

5.3 FUTURE WORK

A software implementation of the extrinsic evaluation framework described in Chapter 3 is provided under an open-source license. I believe the community would benefit from a unified software package that enables researchers to easily evaluate their embeddings or compositional algorithms. I therefore plan to integrate and distribute a collection of diverse freely available corpora into the software. My hope is that the framework would become a part of researchers' toolboxes and would be used to assess and improve distributional algorithms as the field moves forward. Even if the software fails to get traction, I hope the empirical evidence against the word similarity task (Section 3.1) would push the community away from that task and towards extrinsic evaluations.

A second direction for future work is to evaluate a wider range of compositional algorithms that apply to different grammatical structures such as relative clauses, noun phrases with multiple modifiers, intransitive verb phrases and eventually complete sentences.

Another strand of work involves using the qualitative insight gained in Chapter 4 to improve performance at real-world classification problems. Chapter 4 suggests accuracy can be improved by:

- Using neural word embeddings trained on a clean unlabelled corpus that matches the domain of the labelled data. Ideally, a DM should be trained on the labelled data, or (in a semi-supervised scenario) on data sampled from the same distribution as the labelled set.

- Using a large number of replacements (30–50) for each test-time document feature.

- Sub-sampling frequent words (Mikolov et al., 2013c).

I hypothesise accuracy would benefit from better heuristics to decide which test-time features to replace. One possibility is the standard feature expansion rule, where only those test-time features that do not occur in the training data are replaced with nearest neighbours. However, this is inherently limited by the contents of the training set. Another possible extension is to perform feature expansion at both training time and test time in order to reduce sparsity. I believe some form of feature selection at train time may also be beneficial. This would remove uninformative features, so that they cannot be used as replacements.

I am particularly interested in two issues that arise in supervised document classification, namely learning with limited labelled data and learning on-line. Both problems occur often in practice and can potentially benefit considerably from distributional models. In this thesis, and often in the literature, the focus in on large labelled corpora. However, a labelled corpus containing hundreds of thousands of documents has probably taken a long time to collect, which means content drift is very likely. The issue I would like to address is that by the time enough data has been labelled for a classifier to reach acceptable performance, the document stream may have changed, rendering that trained classifier irrelevant. Such massive data sets are therefore of purely academic interest. A potential application of distributional models to document classification is therefore to help speed up the process of 1) annotating data, and 2) adapting classifiers to changing content. One possible solution is to incorporate distributional information into the active learning stage of a system like the one of Kober and Weir (2015). The property that makes this problem particularly suitable for DMs is that one can label features as well as documents. Feature labels can then be used to specialise general-purpose word and phrase embeddings for a particular classification task. A DM can also be used to guide users towards more informative documents to annotate.

CLOSING REMARKS

Distributional semantics has made tremendous progress in the past two decades. As the field moves forward, the need for good evaluation methods becomes more pronounced. This thesis argued against the popular family of intrinsic evaluation techniques; presented a novel extrinsic evaluation framework and made first steps towards

understanding distributional models in that framework. This work on extrinsic evaluation will hopefully inform and contribute to research into the important topic of natural language semantics.

BIBLIOGRAPHY

J. Andreas and D. Klein. How much do word embeddings encode about syntax? *Proceedings of ACL*, 2014.

D. Andrzejewski, X. Zhu, and M. Craven. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. *Proceedings of ICML*, pages 25–32, 2009.

C. Baker, C. Fillmore, and J. Lowe. The Berkeley FrameNet project. *Proceedings of ICCL*, pages 86–90, 1998.

L. D. Baker and A. K. McCallum. Distributional clustering of words for text classification. *Proceedings of ACM SIGIR*, pages 96–103, 1998.

T. Baldwin, C. Bannard, T. Tanaka, and D. Widdows. An empirical model of multiword expression decomposability. *Proceedings of the ACL 2003 workshop on Multiword expressions*, pages 89–96, 2003.

M. Baroni and R. Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. *Proceedings of EMNLP*, pages 1183–1193, 2010.

M. Baroni, G. Dinu, and G. Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. *Proceedings of ACL*, 2014.

S. Bartunov, D. Kondrashkin, A. Osokin, and D. Vetrov. Breaking sticks and ambiguities with adaptive skip-gram. *arXiv Preprint arXiv:1502.07257*, 2015.

J. Berant and P. Liang. Semantic parsing via paraphrasing. *Proceedings of ACL*, 2014.

J. Berant, A. Chou, R. Frostig, and P. Liang. Semantic parsing on freebase from question-answer pairs. *Proceedings of EMNLP*, pages 1533–1544, 2013.

C. Biemann and E. Giesbrecht. Distributional semantics and compositionality 2011: Shared task description and results. *Proceedings of the ACL workshop on Distributional Semantics and Compositionality*, pages 21–28, 2011.

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

D. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, 2012.

D. Blei and J. Lafferty. Dynamic topic models. *Proceedings of ICML*, pages 113–120, 2006.

D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. *Proceedings of NIPS*, 16:17, 2004.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Proceedings of JMLR*, 3:993–1022, 2003.

E. Bruni, N.-K. Tran, and M. Baroni. Multimodal distributional semantics. *Proceedings of JAIR*, 49:1–47, 2014.

J. A. Bullinaria and J. P. Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, pages 510–526, 2007.

J. A. Bullinaria and J. P. Levy. Extracting semantic representations from word co-occurrence statistics: stop-lists, stemming, and SVD. *Behavior research methods*, 44(3):890–907, 2012.

J. Carroll, A. Copestake, D. Flickinger, and V. Poznanski. An efficient chart generator for (semi-) lexicalist grammars. In *Proceedings of the 7th European workshop on natural language generation*, pages 86–95, 1999.

N. Chawla, N. Japkowicz, and A. Kotcz. Editorial: Special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1–6, 2004.

D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. *Proceedings of NIPS*, pages 2843–2851, 2012.

S. Clark. Type-driven syntax and semantics for composing meaning vectors. *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*, pages 359–377, 2013.

D. Clarke. A context-theoretic framework for compositionality in distributional semantics. *Computational Linguistics*, 38(1):41–71, 2012.

B. Coecke, M. Sadrzadeh, and S. Clark. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36(1-4):345–384, 2011.

R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of ICML*, pages 160–167, 2008.

A. Copestake. Semantic transfer in verbmobil. *ACQUILEX II working paper 62 and Verbmobil report 93*, 1995.

A. Copestake, D. Flickinger, R. Malouf, S. Riehemann, and I. Sag. Translation using minimal recursion semantics. In *Proceedings of the Sixth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 15–32, 1995.

A. Copestake, D. Flickinger, C. Pollard, and I. Sag. Minimal recursion semantics: An introduction. *Research on Language and Computation*, 3(2-3):281–332, 2005.

M. Dalrymple. *Semantics and syntax in Lexical Functional Grammar: The resource logic approach*. MIT Press, London, 1999.

M. Dalrymple (editor). *Lexical-Functional Grammar*. Wiley Online Library, 2001.

D. Davidov, E. Gabrilovich, and S. Markovitch. Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. *Proceedings of ACM SIGIR*, pages 250–257, 2004.

M.-C. De Marneffe and C. D. Manning. Stanford typed dependencies manual. Technical report, Stanford University, 2008.

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

G. Dinu and M. Baroni. How to make words with vectors: Phrase generation in distributional semantics. *Proceedings of ACL*, 2014.

G. Dinu and M. Lapata. Measuring distributional similarity in context. *Proceedings of the 2010 EMNLP*, pages 1162–1172, 2010.

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. CRC press, 1994.

T. Eiter and H. Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34:103–133, 1997.

K. Erk. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653, 2012.

K. Erk and S. Padó. A structured vector space model for word meaning in context. *Proceedings of EMNLP*, pages 897–906, 2008.

S. Evert. *The Statistics of Word Cooccurrences*. PhD thesis, Stuttgart University, 2005.

L. Fagarasan, E. M. Vecchi, and S. Clark. From distributional semantics to feature norms: grounding semantic models in human perceptual data. *Proceedings of WCS*, pages 52–57, 2015.

M. Faruqui and C. Dyer. Non-distributional word vector representations. *arXiv Preprint arXiv:1506.05230*, 2015.

M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. Hovy, and N. A. Smith. Retrofitting word vectors to semantic lexicons. *arXiv Preprint arXiv:1411.4166*, 2014.

C. Fillmore. Frame semantics. *Linguistics in the morning calm*, pages 111–137, 1982.

L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: The concept revisited. *Proceedings of the 10th international conference on World Wide Web*, pages 406–414, 2001.

J. Firth and F. Palmer. *Selected Papers of JR Firth, 1952-59*. Indiana University Press, 1968.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3(3):209–226, 1977.

A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, and T. Mikolov. DeViSE: A deep visual-semantic embedding model. *Proceedings of NIPS*, pages 2121–2129, 2013.

R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu. Learning semantic hierarchies via word embeddings. *Proceedings of ACL*, 1, 2014.

M. Gamon, A. Aue, S. Corston-Oliver, and E. Ringger. Pulse: Mining customer opinions from free text. In *Advances in Intelligent Data Analysis VI*, pages 121–132. Springer, 2005.

E. Giesbrecht. Towards a matrix-based distributional model of meaning. *Proceedings of the NAACL-HLT*, pages 23–28, 2010.

J.-Y. Girard. Linear logic. *Theoretical computer science*, 50(1):1–101, 1987.

R. González-Ibáñez, S. Muresan, and N. Wacholder. Identifying sarcasm in twitter: A closer look. *Proceedings of ACL-HLT*, pages 581–586, 2011.

R. Gray. Vector quantization. *ASSP Magazine, IEEE*, 1(2):4–29, April 1984.

E. Grefenstette. Towards a formal distributional semantics: Simulating logical calculi with tensors. *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*, 2013.

E. Grefenstette and M. Sadrzadeh. Experimental support for a categorical compositional distributional model of meaning. *Proceedings of the 2011 EMNLP*, 2011.

E. Grefenstette, M. Sadrzadeh, S. Clark, B. Coecke, and S. Pulman. Concrete sentence spaces for compositional distributional models of meaning. *Proceedings of the 9th International Conference on Computational Semantics*, 2011.

E. Grefenstette, G. Dinu, Y.-Z. Zhang, M. Sadrzadeh, and M. Baroni. Multi-step regression learning for compositional distributional semantics. *Proceedings of IWCS*, 2013.

G. Grefenstette. *Explorations in Automatic Thesaurus Discovery*. Springer, 1994.

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

A. Gupta, G. Boleda, M. Baroni, and S. Padó. Distributional vectors encode referential attributes. *Proceedings of EMNLP*, 2015.

Z. Harris. Distributional structure. *Word*, 1954.

K. M. Hermann and P. Blunsom. The role of syntax in vector space models of compositional semantics. *Proceedings of ACL*, pages 894–904, 2013.

W. Hersh, C. Buckley, T. Leone, and D. Hickam. Ohsumed: An interactive retrieval evaluation and new large test collection for research. *SIGIR'94*, pages 192–201, 1994.

F. Hill, R. Reichart, and A. Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *arXiv Preprint arXiv:1408.3456*, 2014.

D. Hope. *Graph-Based Approaches to Word Sense Induction*. PhD thesis, Department of Informatics, University of Sussex, 2015.

E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. Improving word representations via global context and multiple word prototypes. *Proceedings of ACL*, pages 873–882, 2012.

F. Huang and A. Yates. Distributional representations for handling sparsity in supervised sequence-labeling. *Proceedings of ACL-IJCNLP*, pages 495–503, 2009.

T. Janssen. Compositionality: Its historic context. In M. Werning, W. Hinzen, and E. Machery, editors, *The Oxford handbook of compositionality*. Oxford University Press, 2012a.

T. M. V. Janssen. Montague semantics. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Winter 2012 edition, 2012b.

D. a. Jurgens, P. D. Turney, S. M. Mohammad, and K. J. Holyoak. Semeval-2012 task 2: Measuring degrees of relational similarity. *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 356–364, 2012.

M. Kågebäck, O. Mogren, N. Tahmasebi, and D. Dubhashi. Extractive summarization using continuous vector space models. *Proceedings of the CVSMC workshop at EACL*, pages 31–39, 2014.

D. Kartsaklis, M. Sadrzadeh, and S. Pulman. A unified sentence space for categorical distributional-compositional semantics: Theory and experiments. *Proceedings of COLING*, pages 549–558, 2012.

D. Kiela and L. Bottou. Learning image embeddings using convolutional neural networks for improved multi-modal semantics. *Proceedings of EMNLP*, pages 36–45, 2014.

D. Kiela and S. Clark. Multi- and cross-modal semantics beyond vision: Grounding in auditory perception. *Proceedings of EMNLP*, 2015.

D. Kiela, L. Fagarasan, and S. Clark. Grounding semantics in olfactory perception. *Proceedings of ACL*, pages 231–236, 2015.

S.-M. Kim and E. Hovy. Determining the sentiment of opinions. *Proceedings of ACL*, page 1367, 2004.

Y. Kim. Convolutional neural networks for sentence classification. *arXiv Preprint arXiv:1408.5882*, 2014.

R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv Preprint arXiv:1411.2539*, 2014.

R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler. Skip-thought vectors. *arXiv Preprint arXiv:1506.06726*, 2015.

A. Klementiev, I. Titov, and B. Bhattarai. Inducing crosslingual distributed representations of words. *Citeseer*, 2012.

T. Kober and D. Weir. Optimising agile social media analysis. *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 31–40, September 2015.

H. Kriegel and M. Schubert. Classification of websites as sets of feature vectors. *Databases and Applications*, 2004.

M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. From word embeddings to document distances. *Proceedings of ICML*, 2015.

I. Labutov and H. Lipson. Re-embedding words. *Proceedings of ACL*, pages 489–493, 2013.

J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

T. Landauer and S. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211, 1997.

G. Lapesa and S. Evert. A large scale evaluation of distributional semantic models:parameters, interactions and model selection. *Transactions of ACL*, 2:531–545, 2014.

A. Lazaridou, E. Bruni, and M. Baroni. Is this a wampimuk? cross-modal mapping between distributional semantics and the visual world. *Proceedings of ACL 2014*, 2014.

Q. Le and T. Mikolov. Distributed representations of sentences and documents. *arXiv Preprint arXiv:1405.4053*, 2014.

R. Lebret and R. Collobert. N-gram-based low-dimensional representation for document classification. *arXiv Preprint arXiv:1412.6277*, 2014.

R. Lebret, J. Legrand, and R. Collobert. Is deep learning really necessary for word embeddings? *NIPS Workshop on Deep Learning*, 2013.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

O. Levy and Y. Goldberg. Dependency-based word embeddings. *Proceedings of ACL*, 2, 2014a.

O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. *Proceedings of NIPS*, pages 2177–2185, 2014b.

D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397, 2004.

D. Lin. Automatic retrieval and clustering of similar words. *Proceedings of ACL*, pages 768–774, 1998.

W. Ling, C. Dyer, A. Black, and I. Trancoso. Two/too simple adaptations of word2vec for syntax problems. *Proceedings of NAACL*, 2015.

M.-T. Luong, R. Socher, and C. Manning. Better word representations with recursive neural networks for morphology. *Proceedings of CoNLL*, 104, 2013.

M. Lyra, D. Clarke, H. Morgan, J. Reffin, and D. Weir. High value media monitoring with machine learning. *Künstliche Intelligenz*, 27 (3):255–265, 2013.

A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. *Proceedings of the 49th ACL-HLT*, pages 142–150, 2011.

C. Manning. An introduction to formal computational semantics. 2005.

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. *Proceedings of ACL*, pages 55–60, 2014.

M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

J. McAuley and J. Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172, 2013.

J. D. McAuliffe and D. M. Blei. Supervised topic models. *Proceedings of NIPS*, pages 121–128, 2008.

A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. *AAAI-98 workshop on learning for text categorization*, 752:41–48, 1998.

D. McCarthy and R. Navigli. Semeval-2007 task 10: English lexical substitution task. *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, 2007.

D. McCarthy, B. Keller, and J. Carroll. Detecting a continuum of compositionality in phrasal verbs. *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment*, pages 73–80, 2003.

P. J. McCormick, N. Elhadad, and P. D. Stetson. Use of semantic features to classify patient smoking status. *AMIA Annual Symposium Proceedings*, 2008, 2008.

K. McRae, G. S. Cree, M. S. Seidenberg, and C. McNorgan. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37(4):547–559, 2005.

I. Mel'čuk. *Opyt teorii lingvisticheskih modelej Smysl-Tekst: Semantika, sintaksis (Experience of the theory of linguistic Meaning-Text models)*. Jazyki Russkoj Kul'tury, 1999.

I. Mel'čuk and A. Zholkovsky. *Tolkovo-kombinatornyj slovar russkogo jazyka. Opyty semantiko-sintaksicheskogo opisania russkoj leksiki (Explanatory Combinatorial Dictionary of Russian. Towards a Semantic and Syntactic Description of Russian Lexicon)*, volume 14. Gesellschaft zur Förderung slawistischer Studien, 1984.

I. A. Mel'čuk and A. Polguere. A formal lexicon in the meaning-text theory:(or how to do lexica with words). *Computational linguistics*, 13(3-4):261–275, 1987.

V. Metsis, I. Androutsopoulos, and G. Paliouras. Spam filtering with Naive Bayes-which Naive Bayes? *CEAS*, pages 27–28, 2006.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv Preprint arXiv:1301.3781*, 2013a.

T. Mikolov, Q. v. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *arXiv Preprint arXiv:1309.4168*, 2013b.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Proceedings of NIPS*, 2013c.

D. Milajevs, D. Kartsaklis, M. Sadrzadeh, and M. Purver. Evaluating neural word representations in tensor-based compositional settings. *Proceedings of EMNLP*, pages 708–719, 2014.

J. Milićević. A short guide to the meaning-text linguistic theory. *Journal of Koralex*, 8:187–233, 2006.

G. A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

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

J. Mitchell and M. Lapata. Vector-based models of semantic composition. *Proceedings of ACL-08: HLT*, pages 236–244, 2008.

R. Montague. English as a formal language. 1970a.

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

R. Montague. *The proper treatment of quantification in ordinary English*. Springer, 1973.

A. Neelakantan, J. Shankar, A. Passos, and A. McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv Preprint arXiv:1504.06654*, 2015.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: Sentiment classification using machine learning techniques. *Proceedings of EMNLP*, pages 79–86, 2002.

D. Paperno, N. t. Pham, and M. Baroni. A practical and linguistically-motivated approach to compositional distributional semantics. *Proceedings of ACL*, pages 90–99, 2014.

A. Passos, V. Kumar, and A. McCallum. Lexicon infused phrase embeddings for named entity resolution. *arXiv Preprint arXiv:1404.5367*, 2014.

F. Pedregosa, G. Varoquaux, A. Gramfort, v. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, v. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. *Proceedings of EMNLP)*, 12, 2014.

S. Petrov, D. Das, and R. McDonald. A universal part-of-speech tagset. *arXiv Preprint arXiv:1104.2086*, 2011.

T. Polajnar and S. Clark. Improving distributional semantic vectors through context selection and normalisation. *Proceedings of EACL*, pages 230–238, 2014.

T. Polajnar, L. Fagarasan, and S. Clark. Reducing dimensions of tensors in type-driven distributional semantics. *Emnlp*, pages 1036–1046, 2014.

J. Pustejovsky. *The generative lexicon*. MIT Press, London, 1995.

S. Reddy, I. P. Klapaftis, D. McCarthy, and S. Manandhar. Dynamic and static prototype vectors for semantic composition. *IJCNLP*, pages 705–713, 2011.

M. Redington, N. Chater, and S. Finch. Distributional information and the acquisition of linguistic categories: A statistical approach. *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, pages 848–853, 1993.

R. Řehůřek and P. Sojka. Software framework for topic modelling with large corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, May 2010.

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

V. Rus, R. Banjade, and M. Lintean. On paraphrase identification corpora. *Proceeding on LREC)*, 2014.

M. Sadrzadeh, S. Clark, and B. Coecke. The Frobenius anatomy of word meanings I: Subject and object relative pronouns. *Journal of Logic and Computation*, page ext044, 2013.

M. Sadrzadeh, S. Clark, and B. Coecke. The Frobenius anatomy of word meanings II: Possessive relative pronouns. *Journal of Logic and Computation*, 2014.

J. Saeed. *Semantics*. Wiley-Blackwell, London, third edition, 2009.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

K. Schubert. Linguistic and extra-linguistic knowledge. *Machine Translation*, 1(3):125–152, 1986.

H. Schütze. Word space. *Proceedings of NIPS*, 1993.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

R. Socher, E. Huang, J. Pennington, A. Ng, and C. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Proceedings of NIPS*, 24, 2011.

R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic compositionality through recursive matrix-vector spaces. *Proceedings of EMNLP-CoNLL*, pages 1201–1211, 2012.

R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng. Parsing with compositional vector grammars. *Proceedings of ACL*, 2013a.

R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of EMNLP*, pages 1631–1642, 2013b.

K. Sparck-Jones. Experiments in semantic classification. *Mechanical Translation and Computational Linguistics*, 8(3–4), 1965.

K. Sparck-Jones and J. R. Galliers. *Evaluating Natural Language Processing Systems: An Analysis and Review*. Springer Science & Business Media, 1995.

M. Steyvers. Combining feature norms and text data with topic models. *Acta Psychologica*, 133(3):234–243, 2010.

D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin. Learning sentiment-specific word embedding for Twitter sentiment classification. *Proceedings of ACL*, 1:1555–1565, 2014.

S. Thater, H. Fürstenau, and M. Pinkal. Contextualizing semantic representations using syntactically enriched vector models. *Proceedings of ACL*, pages 948–957, 2010.

E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *Proceedings of CoNLL*, pages 142–147, 2003.

J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semi-supervised learning. *Proceedings of ACL*, pages 384–394, 2010.

P. Turney and P. Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1): 141–188, 2010.

P. D. Turney. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, pages 533–585, 2012.

L. Van Der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.

E. Vecchi and M. Baroni. (Linear) maps of the impossible: Capturing semantic anomalies in distributional space. *Of the Workshop on Distributional Semantics and Compositionality*, pages 1–9, 2011.

U. Von Luxburg, R. C. Williamson, and I. Guyon. Clustering: Science or art? *ICML Unsupervised and Transfer Learning*, pages 65–80, 2012.

S. Wang and C. D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. *Proceedings of ACL*, pages 90–94, 2012.

J. Weeds. *Measures and Applications of Lexical Distributional Similarity*. PhD thesis, School of Informatics, University of Sussex, 2003.

J. Weeds, D. Clarke, J. Reffin, D. Weir, and B. Keller. Learning to distinguish hypernyms and co-hyponyms. *Proceedings of COLING*, 2014a.

J. Weeds, D. Weir, and J. Reffin. Distributional composition using higher-order dependency vectors. *Proceedings of the CVSMC Workshop at EACL*, pages 11–20, 2014b.

D. Weir, J. Weeds, and J. Reffin. An alternative conception of compositional distributional semantics. in press.

J. Weston, A. Bordes, S. Chopra, and T. Mikolov. Towards AI-complete question answering: A set of prerequisite toy tasks. *arXiv Preprint arXiv:1502.05698*, 2015.

R. Wicentowski and M. R. Sydes. Emotion detection in suicide notes using maximum entropy classification. *Biomedical Informatics Insights*, 5, 2012.

T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. *Proceedings of HLT-EMNLP*, pages 347–354, 2005.

C. Xu, D. Tao, and C. Xu. A survey on multi-view learning. *arXiv Preprint arXiv:1304.5634*, 2013.

M. Yu and M. Dredze. Improving lexical embeddings with semantic knowledge. *Proceedings of ACL*, 2014.

X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. *arXiv Preprint arXiv:1509.01626*, 2015.