Diverse Context for Learning Word Representations

Manaal Faruqui

CMU-LTI-16-008

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:

Chris Dyer (chair), Carnegie Mellon University
Noah A. Smith, Carnegie Mellon University
Eduard Hovy, Carnegie Mellon University
Marco Baroni, University of Trento

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies

To my parents, who showed me what education can do.

Abstract

Word representations are mathematical objects that capture a word's meaning and its grammatical properties in a way that can be read and understood by computers. Word representations map words into equivalence classes such that words that share similar properties to each other are part of the same equivalence class. Word representations are either constructed manually by humans (in the form of word lexicons, dictionaries etc.) or obtained automatically using unsupervised learning algorithms. Since, manual construction of word representations is unscalable, and expensive, obtaining them automatically is desirable.

Traditionally, automatic learning of word representations has relied on the *distributional hypothesis*, which states that the meaning of a word is evidenced by the words that occur in its context (Harris, 1954). Thus, existing word representation learning algorithms like latent semantic analysis (Deerwester *et al.*, 1990; Landauer and Dumais, 1997), derive word meaning in terms of aggregated co-occurrence counts of words extracted from unlabeled monolingual corpora.

In this thesis, we diversify the notion of context to include information beyond the monolingual distributional context. We show that information about word meaning is present in other contexts like neighboring words in a semantic lexicon, context of the word across different languages, and the morphological structure of the word. We show that in addition to monolingual distributional context these sources provide complementary information about word meaning, which can substantially improve the quality of word representations. We present methods to augment existing models of word representations to incorporate these knowledge sources.

Acknowledgement

The last four years at Carnegie Mellon have been the best years of my life till now. I thoroughly enjoyed these years because I was working on something that fascinates me while being in a company of an incredibly smart and amazing set of people.

I have been extremely lucky to have a great advisor, Chris Dyer, whose depth of knowledge often makes me exclaim "*Chris, you are awesome, how do you know so much?!*", and whose level of enthusiasm for research in natural language processing, I will never be able to match. Chris is an excellent thesis advisor. I believe I could not have been happier and more satisfied with the work that I have done in my PhD working with someone else other than Chris. Chris provided me with deep technical advice as well as made me look at the big picture. Chris also made sure that I am never stressed out by letting me take as many holidays as I wanted. Chris is an amazing person, and I would love to work with him again in the future.

A lot of my research work has taken shape because of the mentorship that I received from Noah Smith. Whenever I had questions about whether I am doing something which is "really" useful or not, or what is the right way of doing good research, I consulted Noah. The way Noah amazingly manages his work and time is inspiring, and I wish I will become as efficient as him one day. I would like to thank him for investing his time in helping me.

It has been a great pleasure to have shared my grad life with amazing collaborators and friends that I made in my research group. Thanks Yulia Tsvetkov and Dani Yogatama for being helpful collaborators. I learnt a lot about writing research papers from you. I am thankful to Rajarshi Das, Ankur Gandhe and Keerthiram Murugesan for the many refreshing coffee breaks that we took together. I had an amazing time with c-lab and noahs-ark members that include Waleed Ammar, Wang Ling, Avneesh Saluja, Austin Matthews, Swabha Swayamdipta, Jesse Dodge, Sujay Jauhar, Kartik Goyal, Nathan Schneider, Lingpeng Kong, Guillaume Lample, Kazuya Kawakami, and Adhiguna Kuncoro. Thanks Arnab Bhattacharya, Ankita Mangal, Adhithi Shwetha Adhi for hanging out with me all the time, and being a strong source of support. Thanks Piyush Panigrahi for the enlightening and encouraging discussions about life and research.

I would like to thank Shankar Kumar and Ryan McDonald for hosting me in the summer of 2014, and 2015 as a research internship student in Google New York and Google London

respectively. It was a great experience working with you and I hope to be able to collaborate with you in the near future. While at Google I got to work on real-world NLP problems. A part of the work presented in this thesis was the research project that I carried out at Google London with Ryan.

I am indebted to my parents, Jasmeen and Suhail, for teaching me to believe in myself and giving me the courage to make my own decisions. They taught me the importance of education in life, and always supported me in my endeavours. I am blessed to have a younger sister, Dania, who makes me feel alive, and is a constant source of entertainment. Thanks for taking care of our parents while I was busy doing my PhD. Finally, I would like to thank Nisarga for her patience, love, and encouragement that kept me going. Thanks for keeping me happy, taking me on numerous vacations, and being always there for me. I know you feel happier than me about me graduating with a PhD.

Contents

Chapter 1

Introduction

What is word meaning? Anything that conveys information about how a word can be used in a language, what information the word conveys, or how a given word is related to other words in the language, can be said to represent the meaning of a word (Frege, 1884; De Saussure, 1989). Understanding the meaning of words is essential for effective natural language processing. For example, the word *playing* can mean taking part in a sport, participating in an activity etc. Its part-of-speech is verb with gerund verb form. These are some of the aspects of word meaning which help us identify the word's role in a language. Thus, for developing natural language processing models that can understand languages, its essential to develop a structure that can capture aspects of word meaning in a form that is readable and understandable by computers.

Word meaning representation (or generally called word representation) is a mathematical object associated with each word, often a vector. Each dimension's value corresponds to a feature and might even have a semantic or grammatical interpretation (Turian *et al.*, 2010). Word representations map words into equivalence classes such that words that are similar to each other are part of the same equivalence class. In this thesis we look at this aspect of word meaning, which maps words into equivalence classes. For example, in word clusters, words are grouped into a finite number of classes, and words in the same class are supposed to be similar. In word vectors, words are represented in a continuous vector-space of fixed dimensionality, and words that lie close together in this vector-space are supposed to be similar.

Another example of word representation are word lexicons. Word lexicons can contain information about different linguistic properties of a language and relationship between different words, such as WordNet (Fellbaum, 1998), FrameNet (Baker *et al.*, 1998), or morpho-syntactic lexicons (Hajič and Hladká, 1998a). The morpho-syntactic attributes for the word *playing* are Pos:Verb, VerbForm:Gerund, Tense:Present etc. which can be represented as a vector where every dimension corresponds to a particular morpho-syntactic attribute. Although such lexicons contain rich information about a word's meaning, they are difficult to

obtain as they are often constructed manually by humans. Thus, there is a need of methods that can construct word representations readily without requiring expensive resources across various languages.

The distributional hypothesis of Harris (1954) states that the meaning of words is evidenced by the contexts they occur in. This hypothesis was famously articulated as "*You shall know a word by the company it keeps*" by Firth (1957). For example, since the words *playing*, *jumping*, and *running* might often occur with similar contextual words in a corpus, they should have similar meaning, and thus similar word representations. Information about the nature of occurrence of a word in a language is called word distributional information. Word distributional information can be easily captured from word co-occurrence statistics (how many times a particular word occurs in the context of another word) computed from unlabeled text corpora, which is the primary reason for popularity of distributional word representations as word representations. The distributional hypothesis has motivated several effective techniques for obtaining meaning representations of words using unannotated text corpora. These techniques capture word meaning in the form of word clusters (Brown *et al.*, 1992) or word vectors (Turney and Pantel, 2010). These word representations are shown to reflect intuitions about lexical semantics (Turney, 2001) and are also useful in downstream NLP applications (Turian *et al.*, 2010; Guo *et al.*, 2014).

Although there is much evidence in favor of the distributional hypothesis, in this thesis we show that the quality of word representations (as evaluated on a bunch of intrinsic and extrinsic tasks) obtained using distributional information can be further improved by various other sources that provide us information about the word's meaning. Figure 1.1 shows the different sources of information that we use to improve distributional word representations. An example of an additional source of word meaning is the morphological structure of the word itself (Faruqui *et al.*, 2016). When a person reads the word *played* in English, they can identify that this is probably the past tense of the verb *play*, because of the presence of the suffix "*-ed*". We present methods to include such sources of information about word meaning in word representations. We experiment with a range of word representations that include word clusters, word vectors and word lexicons. We show how semantic lexicons, morphology and translational evidence can be incorporated into word meaning representations that have been trained or constructed using distributional evidence.

1.1 Thesis Outline

This thesis consists of the following chapters:

- In chapter 3, we discuss the problem of evaluation of word vector representations. We first discuss the conventional approaches to word vector evaluation and their shortcom-

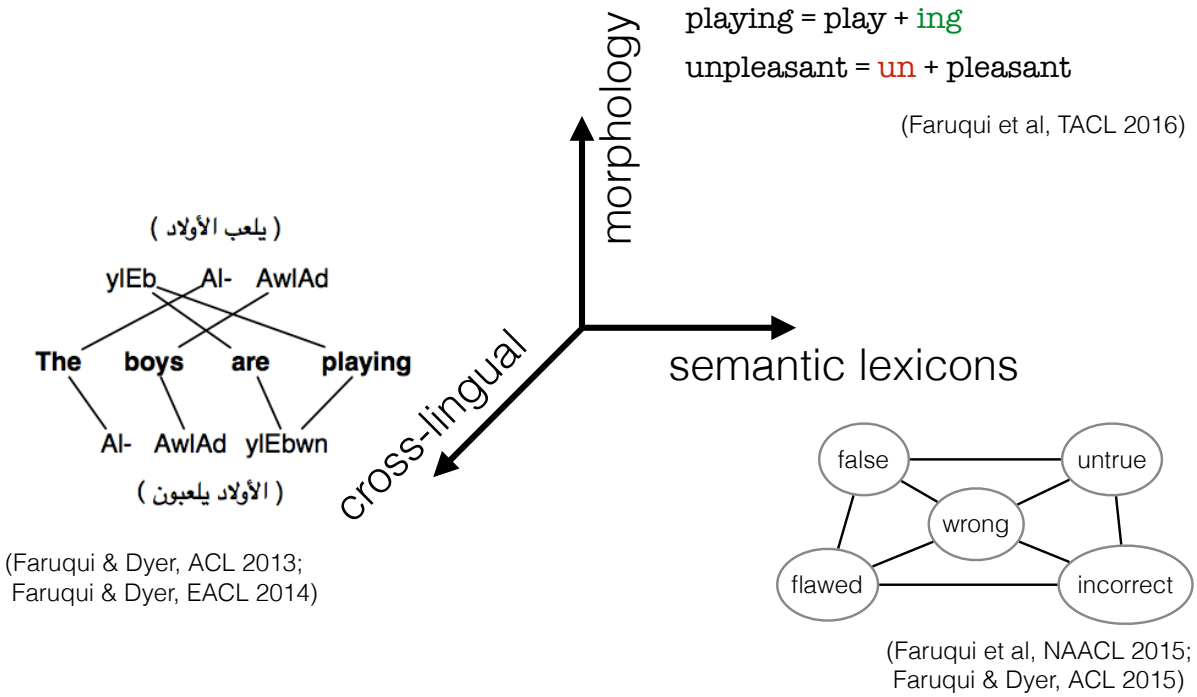

Figure 1.1: The three sources of information in addition to distributional context that we integrate in word representation models.

ings. We then propose an intrinsic evaluation metric for word vector evaluation which computes how well the dimensions in word vectors can quantify linguistic properties.

- In chapter 4, we first show how highly sparse, and binary word vectors can be constructed using information present solely in word lexicons, which we call *non-distributional* word vectors. We then show how information present in manually or automatically constructed lexicons can be used to improve and construct word vector representations. We present a method called *retrofitting*, to incorporate this information in word vectors.

- In chapter 5, we work with morpho-syntactic lexicons. A morpho-syntactic lexicon enumerates all possible morphological and syntactic roles that a word can display in a language. Such lexicons are manually created and have low coverage. In many languages, the morphological structure of a word can be used to identify some of its morphological and syntactic roles. We present a graph-based method that expands a given seed morpho-syntactic lexicon to the size of the vocabulary using morphological relations between words in addition to distributional information.

- In chapter 6, we argue for incorporating translational context in addition to distributional context in word representations. We show that the translation of a word across

languages contains important cues about the meaning of a word, which are not captured in monolingual distributional context. First, we present a bilingual clustering model that produces word clusters which are informed using both monolingual and translational context. Next, we present a canonical correlation-based method that incorporates bilingual context in word vector models.

- In chapter 7, we summarize the contributions of the thesis and provide directions for future work.

1.2 Thesis Statement

In this thesis I advocate for models of word representation that are constructed not only from the monolingual distributional information of the word but also from other sources such as word lexicons, morphology, and translation of a word across languages. I take existing models of word representations and provide methods to augment each one of them to utilize aforementioned sources of word meaning information. I argue that developing models that capture different views of word meaning and their interaction yields better informed word representations. I show that word representations obtained this way will show an improvement in performance on downstream NLP tasks when used as features.

Chapter 2

Word Representations

The performance of machine learning methods relies heavily on the choice of data representation. Thus, much of the effort in deploying machine learning algorithms goes into the feature engineering to best represent data in terms of features that can support effective machine learning. Feature engineering is the process of using human knowledge and prior knowledge about a task to come up with a set of representative features for the data. Even though good feature engineering often translates to great results in machine learning tasks, its reliance on manual labor makes it quite expensive.

In order to expand the ease of applicability of machine learning, it is desirable to make learning algorithm less dependent on feature engineering. *Representation learning* is the learning of representations of the data that make it easier to extract useful information when building classifiers or other predictors (Bengio *et al.*, 2013). Representation learning, when applied to words used in natural language, generates word representations. Word representations are mathematical objects that contain informative features about a word's meaning. Learning word representations is of crucial importance in NLP, as naive mappings of raw text (for example, bag of words) lose meaning of the word, and lead to high-dimensional, sparse representations with heavy-tailed distributions. Word representations have been shown to be useful in downstream NLP tasks like named entity recognition, POS-tagging, dependency parsing (Turian *et al.*, 2010; Guo *et al.*, 2014; Bansal *et al.*, 2014).

Learning word representations in NLP has traditionally relied on the distributional hypothesis. Distributional information of words can be extracted from large unlabeled text corpora which are available for many of the world's languages. Thus, unsupervised learning of word representations using the distributional hypothesis has become widely popular. There are two major types of word representations that can be learned from corpora in an unsupervised manner: word clusters and word vectors. The core of the word representation learning algorithms is to find distributionally similar words and assign them word representations that can map them into the same equivalence classes. For word cluster models, this means that

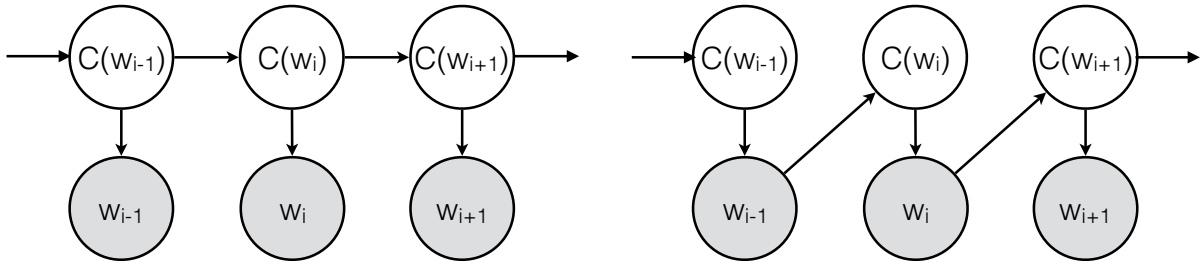

Figure 2.1: Word clustering models of Brown *et al.* (1992) on the left, and Saul and Pereira (1997) on the right. Word clusters in both models are induced through language modeling using cluster-based models. Words are observed (grey nodes), and clusters are unobserved variables (white nodes) which are induced by the word clustering algorithm.

distributionally similar words should be assigned to the same word cluster. For word vectors, distributionally similar words should lie close together in vector space. We will now discuss each of these in detail.

2.1 Word Clusters

A word cluster is a group of words which ideally captures syntactic, semantic, and distributional regularities among the words belonging to the group. Representing a word by its cluster id helps map many words to the same point, and hence leads to reduction in the number of parameters. For example, if the vocabulary of a language is of $100,000$ words, a naive bag-of-words model would require $100,000$ parameters to represent all the words. However, if we use a clustering algorithm to cluster the vocabulary in 100 groups, then we can represent the whole vocabulary by just 100 parameters. The clustering scheme in which every word gets assigned to one cluster is called *hard clustering* (Brown *et al.*, 1992). The clustering scheme in which a word can have multiple memberships in different clusters with different probabilities is called *soft clustering* (Pereira *et al.*, 1993).

In soft clustering, a word w can be presented as the collection of conditional probabilities of different word clusters given the word: $p(c_i|w)$, where c_i is a possible word cluster id and $\sum_i p(c_i|w) = 1$. In hard clustering the word is assigned strictly to only one cluster, and thus $p(c_i|w) = 1$ for only a unique cluster c_i, and thus $\forall_{j \neq i} p(c_i|w) = 0$. In these terms, soft clustering is strictly more expressive than hard clustering. However, the ease of use of hard word clusters as features in downstream tasks have made them pervasive in NLP (Bekkerman *et al.*, 2003; Turian *et al.*, 2010). Hard clusters also provide a simple way of understanding and interpreting word meaning: by looking at the words in a word cluster, once can inspect if there are any commonalities that they share. For example, some words clusters might contain semantically similar words (*run*, *play*, *playground* etc.) whereas some other cluster

might contain syntactically similar words (*writing*, *walking*, *running* etc.).

Word clusters can be learned from a corpus in an unsupervised fashion. Unlabeled corpora of natural language text provides information about the distributional nature of words. Word clustering models that cluster words according to their distributional information are called distributional word clustering models. Other ways of clustering words are also possible, for example, based on spelling features like capitalization or suffixes. Distributional word clusters are shown to contain words that have similar semantic or syntactic meaning, and thus provide useful feature for tasks which rely on the meaning of words like document classification (Baker and McCallum, 1998; Dhillon *et al.*, 2002). An example of a distributional word cluster from Brown *et al.* (1992) is *write*, *writing*, *pen*, *book* etc. Clearly these words are semantically related and representing them by the same cluster id can help reduce sparsity in the data.

Word clusters were originally obtained as a side-product in models that were designed to perform language modeling. In language modeling, the task is to predict the next word in a given sequence of previously observed words. If the previously observed word sequence at time t is $w_1, w_2, \cdots w_{t-1}$, then the task is to predict $p(w_t|w_1, w_2, \cdots w_{t-1})$. In an n-gram language model, where the history is constrained to only the previous $n-1$ words, this probability can be written as $p(w_t|w_{t-n+1}, w_{t-n+2}, \cdots w_{t-1})$. In an n-gram language model, we treat two histories as equivalent if they end in the same $n-1$ words. For a vocabulary of size V, an n-gram language model thus has $O(V^n)$ independent parameters.

Such large number of parameters in an n-gram language model makes it difficult to achieve reliable estimates. To avoid this problem, Brown *et al.* (1992) propose to perform language modeling by partitioning the vocabulary of the language in a finite number of clusters, and then modeling the sequence of words as emissions produced by a hidden sequence of clusters. Concretely, every word is emitted by a hidden cluster that contains it, and the next cluster in the sentence depends on the cluster of the previous word. Figure 2.1 shows the model used by Brown *et al.* (1992) for language modeling. The number parameters of this model is $C^2 + VC$, where C is the number of clusters, and thus it's linear in the size of the vocabulary.

The model of Brown *et al.* (1992) is a hidden-Markov model, with words as observed sequence and clusters as the unobserved sequence. The parameters of the model are estimated using the Baum-Welch algorithm, by performing a forward and reverse traversal on the word sequence. Training this HMM over a large corpus effectively clusters the vocabulary in word clusters. Saul and Pereira (1997) present an alternative model to perform language modeling using a class-based model. The model proposed by them is shown to the right in figure 2.1. The parameters in their model are the transition probabilities between words and word classes. The advantage of their model over Brown *et al.* (1992) is that the parameters can be easily estimated using EM algorithm without requiring multiple passes over the word sequence. Word clusters can also be obtained using a matrix decomposition method called *spectral clustering* (Ng *et al.*, 2002), where the matrix can be constructed from word co-

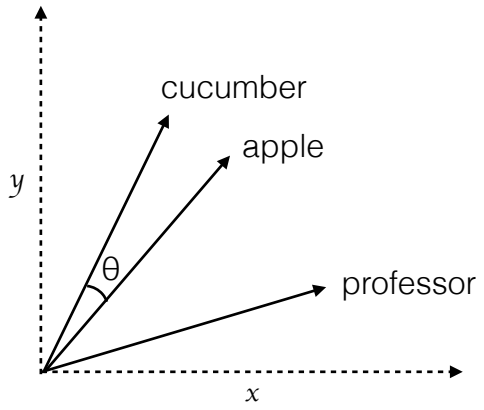

Figure 2.2: Word similarity between a word pair is computed as cosine of the angle (θ) between their word vectors.

occurrence counts computed over an unlabeled corpus (Goyal and Hovy, 2014). Overall, these models show how word clusters can be obtained using distributional information extracted from natural language text.

2.2 Word Vectors

Till now we have discussed how words can be represented using word clusters. One limitation of word clusters is that they only provide similarity or difference between two words along only one "dimension". That is, either a pair of given words belong in the same word cluster or not. Hinton (1984) argued that representations of entities should be represented by a pattern of activity distributed over many concepts, and every concept can be involved in representing many different entities. For example, *princess* is, as kind of a person, similar to *man*, and *woman*, and as royally, similar to *castle*, and *crown*. In other words, there exists a many to many mapping between concepts and entities, and such representations are called *distributed representations*. We now discuss the distributed representation of words, commonly called *word vectors* or *word embeddings* in detail.

A word vector is a finite dimensional vector of real numbers which represents a word in the vector space. The dimensions stand for context items (for example, co-occurring words), and the coordinates depend on the co-occurrence counts. The similarity between two words can then be measured in terms of proximity in the vector space. One simple way of computing word similarity between two words is the cosine similarity (cosine of the angle between the two word vectors) between the two word vectors. Figure 2.2 shows three words in a two-dimensional vector space. The similarity between the words cucumber and apple can be computed as $cosine(\theta)$.

Word vectors can be learned from a corpus in an unsupervised fashion. We will now

describe the approaches that use distributional information extracted from large unlabeled corpora to construct word vectors. On a coarse grained scale, these methods can be divided into two major sections: matrix factorization based spectral methods, and neural network based embeddings methods. We now describe each one of these in enough detail to establish the necessary foundation for the rest of the thesis.

2.2.1 Matrix Factorization Methods

Constructing a vector space of word meaning starts from extracting distributional information from a text corpus. According to the distributional hypothesis, word-word co-occurrence is indicative of a word's meaning. Such word-word co-occurrence can be easily extracted from text corpora. Consider the small sample corpus containing six sentences below:

The boy is playing in the field.
The boy is running in the field.
The boy is playing in the playground.
The girl is playing in the playground.

If we extract co-occurrence counts at the sentence level i.e., every word is said to be in the context of another word in the same sentence, then we can represent the counts in the form of the following matrix:

$$\mathbf{M} = \begin{array}{c} \\ \textit{playing} \\ \textit{running} \\ \textit{field} \\ \textit{playground} \end{array} \begin{array}{c} \begin{array}{cccccc} \textit{playing} & \textit{running} & \textit{field} & \textit{playground} & \textit{boy} & \textit{girl} \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 2 & 2 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 2 & 0 \\ 2 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

M is the matrix of co-occurrence counts of words in the corpus. The rows of **M** are representative of the words that we want to obtain the representations for (in this case, words related to sports), and the columns of the matrix are the features that are important for determining the word's representations (in this case, all words except the stop words). Every row in the matrix now is the vector representation of the word corresponding to the row. Thus, the similarity between words *field*, and *playground* can be computed as $cosine(\vec{field}, \vec{playground}) = 0.67$.

Counts Weighting. The co-occurrence counts in the matrix can be weighted to give more weight to more surprising elements and less weight to expected events. For example, in measuring similarity between the words *mouse* and *rat*, the context *dissect* is more discriminative than the contexts *have* or *like*. The counts in **M** can be replaced by the pointwise mutual information (PMI) (Church and Hanks, 1990), which is shown to work well for obtaining word

vectors (Turney, 2002). Other variations of this score include the positive-PMI, which is more interpretable (Bullinaria and Levy, 2007).

Context Features. Simple word-word co-occurrence has been shown to be a good source of features for obtaining word vector representations that are semantic in nature, are useful in semantic tasks like word similarity, synonym selection, text classification etc. Turney (2001). However, the columns in the matrix $\mathbf{M}$ can also be designed to capture syntactic context of the words, for example, by considering the context of a word to be words that are related to it by a syntactic dependency relation in the sentence (Lin, 1998; Padó and Lapata, 2007; Baroni and Lenci, 2011). Such contextual features are shown to yield word vectors that follow *functional* similarity instead of the usual *topical* similarity (Levy and Goldberg, 2014a).

Another form of context that is often used for obtaining semantic word vectors in cross-lingual context. The hypothesis behind using cross-lingual context is that if two words in one language translate to the similar words in another language, then they might have similar meaning (Bannard and Callison-Burch, 2005). Thus, if information about the translation of a word is available through word alignments of word lexicons, then these translated words can also be used as features in the columns of the word co-occurrence matrix (Utt and Padó, 2014; Gardner *et al.*, 2016). We will return to cross-lingual evidence in chapter 6.

Dimensionality Reduction. The rows of $\mathbf{M}$ are word vector representations according to the distributional hypothesis, but the high dimensionality of the vectors (the number of columns in the matrix is the magnitude of the order of the vocabulary size) can make it impractical to use them. There are $O(V^2)$ order entries in the matrix which are estimated from a corpus, and thus they are poorly estimated. Also, not all the dimensions in the matrix are useful, as they might be pulling irrelevant noise, or information that is uninformative in discriminating between word types. Further, lots of information in the matrix can be redundant as similar words (when acting as column features) will provide similar evidence. Thus, its important to represent the information present in the matrix in a lower dimensional space, where all the dimensions provide informative features.

There are several ways of projecting a matrix into a lower dimensional space. The most widely used such method is singular value decomposition (SVD). SVD decomposes $\mathbf{M}$ into the product of three matrices $\mathbf{U}\Sigma\mathbf{V}^\intercal$, where $\mathbf{U}$ and $\mathbf{V}$ are in column orthonormal form (i.e., the columns are orthogonal and have unit length, $\mathbf{U}^\intercal\mathbf{U} = \mathbf{V}^\intercal\mathbf{V} = \mathbf{I}$) and Σ is a diagonal matrix of singular values. If $\mathbf{M}$ is of rank r, then Σ is also of rank r. Let Σ_k, where $k < r$, be the diagonal matrix formed from the top k singular values, and let $\mathbf{U}_k$ and $\mathbf{V}_k$ be the matrices produced by selecting the corresponding columns from $\mathbf{U}$ and $\mathbf{U}$. The matrix $\hat{\mathbf{M}} = \mathbf{U}_k\Sigma_k\mathbf{V}_k^\intercal$ is the matrix of rank k that best approximates the original matrix $\mathbf{M}$, in the sense that it minimizes the approximation errors. That is, $\hat{\mathbf{M}} = \mathbf{U}_k\Sigma_k\mathbf{V}_k^\intercal$ minimizes $\|\hat{\mathbf{M}} - \mathbf{M}\|_F$ over all matrices $\hat{\mathbf{M}}$ of rank

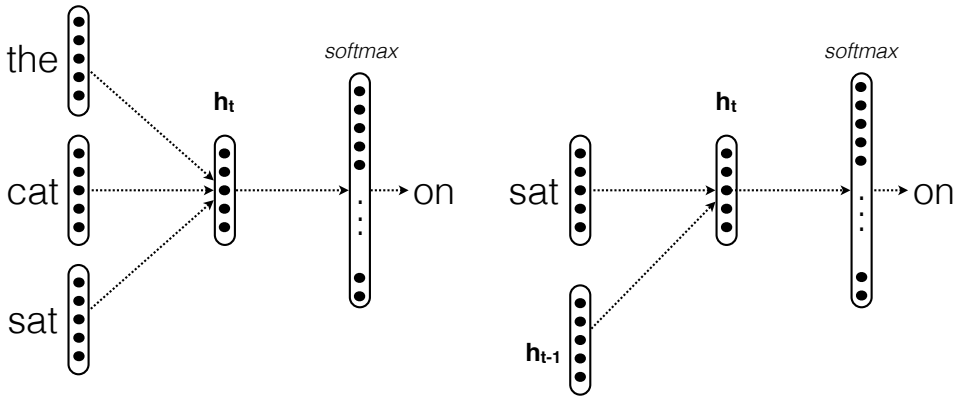

Figure 2.3: Feed-forward neural language model with window size 2 (left) and recurrent neural language model (right). The models are predicting the next word "on" given the previous words "the cat sat".

k, where $\|\cdots\|_F$ denotes the Frobenius norm (Golub and Van Loan, 1996).

The lower dimensional projection using SVD has been qualitatively analysed to display the properties of identifying the latent meaning from the rows and columns of the original matrix (Deerwester *et al.*, 1990; Landauer and Dumais, 1997), reducing the noise in the input matrix **M** (Rapp, 2003), and discovering higher order co-occurrence among the rows (word vectors) of the matrix (Lemaire and Denhiere, 2008). Other ways of obtaining lower dimensional representations from a matrix include principal component analysis (PCA), independent component analysis (ICA), non-negative matrix factorization (NMF), canonical correlation analysis (CCA), CUR decomposition. Each of these methods have different pros and cons when compared to SVD. The explanation of such methods is beyond the scope of this thesis, and we refer the readers to available comprehensive survery in this field (Sayed and Kailath, 2001; Koren *et al.*, 2009).

2.2.2 Neural Word Embeddings

Word vector representations when trained as a part of a neural network aimed at solving a particular task are called word embeddings. Such tasks can be divided into two major categories: language modeling and co-occurrence prediction. Similar to the previous section (§2.2.1) words will again be represented in terms of their context in a corpus, the difference lies in the problem formulation. We now describe how word embeddings can be obtained using such models.

Language Modeling

Bengio *et al.* (2003) introduced a neural language model where the task is to predict the next word given the previous words in a sentence. In their framework every word is represented as a finite-dimensional vector of real numbers. The joint probability function of word sequences is computed in terms of word vectors using a feed-forward neural network. Then the parameters of the neural network and the word embeddings can be trained by maximizing the log likelihood of the model on the data using, for example, backpropagation and gradient ascent.

Language modeling using simple feed-forward networks can only take into account a fixed number of words in the already observed sequence for making the next prediction. However, recurrent neural network models that condition on the entire history of previously observed words have been shown to significantly improve language modeling (Mikolov *et al.*, 2010). Even in the recurrent neural network models, words are represented as word vectors which are optimized as part of the language modeling objective. Figure 2.3 shows the basic structure of these models. In both models, the input words are used to compute a hidden layer which is then used to predict the next word in the sequence using a softmax layer. The softmax function computes the probability of observing every single word in the model's vocabulary, concretely:

$$p(w_i|\mathbf{h}) = \frac{\exp(\mathbf{W}\mathbf{h} + \mathbf{b})_i}{\sum_j \exp(\mathbf{W}\mathbf{h} + \mathbf{b})_j} \tag{2.1}$$

where, $\mathbf{W}, \mathbf{b}$ are model parameters, and $\mathbf{h}$ is the hidden layer vector computed for the prediction. The computation in eq. 2.1 is performed for every word w_i in the vocabulary.

Co-occurrence Prediction

Recent advances in training word embeddings have mostly relied on models that try to predict a word in its context irrespective of the order of the words in the context (Mikolov *et al.*, 2013a; Mnih and Kavukcuoglu, 2013; Pennington *et al.*, 2014). The fact that words occuring both to the left and right of a given word are used to predict the word make such a setup different from language modeling. In language modeling, the task is to predict the next word in the sequence of previously observed words. Word embeddings trained using such models have been shown to better represent the semantics of the word, as opposed to language modeling-based approaches (Zhila *et al.*, 2013; Faruqui and Dyer, 2014b).

In traditional neural models softmax is a costly computation as it computes the probability of observing every single word in the model's vocabulary (as the vocabulary keeps increasing with corpus size according to Heaps' law (Heaps, 1978)), which quickly makes it intractable to be used with large amount of training data. These methods replace the costly softmax function used in traditional neural models by different approximations, such as noise-constrastive estimation (Gutmann and Hyvärinen, 2010), negative sampling (Mikolov *et al.*, 2013a), or hi-

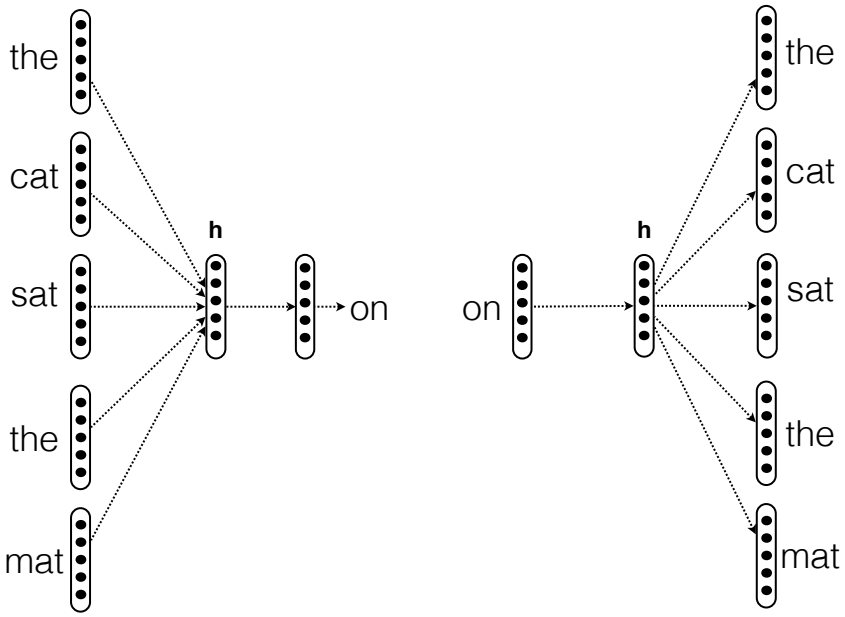

Figure 2.4: The CBOW or LBL (left) and Skip-Gram or iVLBL (right) models. In the sentence *the cat sat on the mat*, the models are predicting the word *on* given the context (left) or the context given the word *on* (right).

erarchical softmax (Goodman, 2001; Morin and Bengio, 2005). With these approximations the computation at the last layer of the network becomes independent of the size of the vocabulary making training possible. For details on these objectives we refer the reader to the original papers and the summary by Dyer (2014).

Figure 2.4 shows the two main models that are used to predict co-occurring words. The figure on the left predicts a word in the center given the contextual words on the left and right side of the word in a sentence. The model predicts *on*, given words in its left and right context in the sentence *the cat sat on the mat*. These models are called the continuous bag of words (CBOW) and the log bilinear models (vLBL). The figure on the right predicts the words on the left and right context of a given word in the sentence. Thus, the model learns to predicts the contextual words given the word *on*. These models are called the skip-gram and the inverse log bilinear model (ivLBL) (Mikolov *et al.*, 2013a; Mnih and Kavukcuoglu, 2013).

2.2.3 Bridging Matrix Factorization & Neural Methods

Matrix factorization methods are attractive because they often obtain a global optimum using a closed form expression (for example, SVD, PCA, CCA etc.), whereas neural networks converge to a local optimum. Matrix factorization methods have a closed form solution provides for stability in the output representations. Also, matrix factorization methods only rely on aggregated co-occurrence counts of pairs of words, whereas neural methods require each

observation of a word in context to be presented separately. Neural methods usually use stochastic gradient ascent to solve a non-convex objective function, which is hard to optimize and thus rely heavily on initialization of parameters and the optimization algorithm. On the other hand, a globally optimal solution is not always required, and the solution returned by training the neural network models is shown to work well in practice (Goodfellow *et al.*, 2015).

One limiting disadvantage of the matrix factorization methods is the absence of an associated probability density or an explanatory generative model. Having an associated probability density would facilitate comparison of matrix factorization models to other probabilistic models. Such an interpretation would also make it easier for matrix factorization models to be directly used as components in larger probabilistic models. Motivated by such objectives, previous research has shown that various matrix factorization algorithms can be seen as performing optimization of a probabilistic objective (Salakhutdinov and Mnih, 2008). For example, PCA has been shown to be solving a maximum likelihood objective (Tipping and Bishop, 1999), which was then extended and generalized to CCA (Bach and Jordan, 2005).

Levy and Goldberg (2014b) show that the single layer neural network of skip-gram model when trained using negative sampling, is equivalent to factorizing a matrix of word co-occurrence counts weighted by shifted positive PMI (positive PMI values changed by a constant amount). They showed that matrix decomposition is better at optimizing the objective of the skip-gram (Mikolov *et al.*, 2013a) and the log bilinear model (Mnih and Kavukcuoglu, 2013) both in terms of intrinsic error, and performance on downstream tasks. This result establishes that there is a trade-off between speed and accuracy while choosing the neural or the matrix factorization models for learning word vectors. Matrix factorization methods are more accurate in optimizing the intrinsic error, but constructing a large co-occurrence matrix and performing matrix factorization on it requires expensive computational resources. Neural models trained using gradient descent can support easier distributed computation, and are faster to train (Dean *et al.*, 2012).

Chapter 3

Evaluation

Word representations are supposed to capture a word's meaning in a language. However, the concept of "meaning" is usually abstract and difficult to interpret. Different forms of word representations capture different aspects of meaning. For example, distributional word representations capture the co-occurrence statistics of a word with other words or phrases extracted from large unlabeled corpora, which provides evidence of its meaning. As discussed in chapter 2, there are several different models which can be used to obtain word vector representations. We want to evaluate the quality of these word vector representations irrespective of the models that produced them. The models used to obtain these representations optimize different loss functions, which makes it hard to compare the intrinsic quality of different forms of word representations. Thus, there are two main issues with intrinsic evaluation of the quality of word meaning representations:

- Abstract definition of word meaning
- Incompatible intrinsic evaluation scores across models

To do away with the problem of intrinsic quality evaluation, many existing tasks measure how well the notion of word similarity according to humans is represented by word vectors (§3.1). The most widely used such tasks are the word similarity evaluation tasks (Finkelstein *et al.*, 2002; Agirre *et al.*, 2009), followed by the word analogy tasks (Mikolov *et al.*, 2013b). Word vectors are also evaluated on the basis of their utility as features in downstream tasks like sentiment analysis, dependency parsing (§3.2). However, such evaluation tasks are mere proxies, and there is no theoretical proof of how extrinsic evaluation is representative of the quality of word vector representations.

In this chapter, we first describe the various tasks which are currently used as for evaluation of word vectors. These tasks are used across the thesis for evaluation of word vectors. We then present an evaluation metric, called QVEC, which is able to measure the intrinsic quality

of word vectors across different models of word vector representations (§3.3).[1] QVEC measures how well the word vectors are able to quantify linguistic content of words. We show that this intrinsic evaluation metric has strong correlation with the performance on word vectors on downstream evaluation tasks.

3.1 Similarity Correlation Measures

We now describe the tasks that are used to evaluate how well word vectors are able to capture the notion of word similarity and analogy.

3.1.1 Word Similarity

The task of word similarity evaluation measures how well does word similarity in the vector space correspond to the notion of word similarity according to humans. The datasets are constructed using human crowdsourcing. A word similarity dataset consists of pairs of words which are presented to humans for annotation, where every annotation measures how similar the two words are as perceived by a human on a scale of 1-10 (or any other scale provided individually for every dataset). These annotations are then aggregated across all subjects to obtain an average measure of similarity between the two words. For example, *professor/cucumber* receives 0.31 and *dollar/buck* receives 9.22 average rating. To measure how well word vector representations capture word similarity in the vector space, we calculate similarity between a given pair of words by the cosine similarity between their corresponding vector representation, as shown in Figure 2.2. We then report Spearman's rank correlation coefficient (Myers and Well, 1995) between the rankings produced by the word vector model against the human rankings.

There have been a number of word similarity datasets produced by the research community over time. There are currently more than 12 datasets that are exclusively used for measuring the effectiveness of word vectors to capture word similarity, as we list on `www.wordvectors.org` (Faruqui and Dyer, 2014a). The size of these datasets varies widely from the smallest dataset containing as little as 30 word pairs (Miller and Charles, 1991) and the largest containing 3000 word pairs (Bruni *et al.*, 2012). We use a subset of these word similarity datasets that are most widely used and are sufficiently large in size to control for instability in results.

The first word similarity dataset is the WS-353 dataset (Finkelstein *et al.*, 2001) containing 353 pairs of English words that have been assigned similarity ratings by humans. This was the first word similarity dataset containing more than 100 word pairs and is still one of the most widely used datasets. The second benchmark is the MEN dataset (Bruni *et al.*, 2012).

[1] Previously published in Tsvetkov *et al.* (2015).

Dataset	Word pairs
WS-353	353
MEN	3000
SimLex	999

Table 3.1: Size of word similarity datasets used for evaluating word vector representations.

MEN contains 3000 pairs of randomly selected words to ensure a balanced range of similarity levels in the selected word pairs. Each word pair is scored on a [0, 1]-normalized semantic relatedness scale via ratings obtained by crowdsourcing on Amazon Mechanical Turk.

It has been claimed that the word pairs in WS-353 and MEN capture both similarity and relatedness instead of exclusively capturing word similarity (Agirre *et al.*, 2009; Hill *et al.*, 2014). For example, *coffee/cup* are rated more similar than *train/car*, even though *train/car* share many common properties (function, material, dynamic behavior, wheels, windows etc.). Although clearly different, *coffee* and *cup* are very much related. Thus, relatedness and similarity are different concepts (Budanitsky and Hirst, 2006; Agirre *et al.*, 2009). A new word similarity dataset SimLex-999 (Hill *et al.*, 2014) was created to model similarity in particular rather than relatedness or association. It contains a balanced set of noun, verb and adjective pairs. This is the third dataset that we use for measuring word similarity. Table 3.1 shows the sizes of different word similarity datasets.

3.1.2 Word Analogy

Mikolov *et al.* (2013b) present two new semantic and syntactic relation dataset composed of analogous word pairs. It contains pairs of tuples of word relations that follow a common relation. For example, in *England : London :: France : Paris*, the two given pairs of words follow the semantic country-capital relation. The semantic dataset (SEM-REL) contains three other such kinds of relations: country-currency, man-woman, city-in-state and overall 8869 such pairs of word tuples. The syntactic dataset (SYN-REL) contains nine such different kinds of relations: adjective-adverb, opposites, comparative, superlative, present-participle, nation-nationality, past tense, plural nouns and plural verbs. An example of the syntactic word analogy is *walk : walked :: talk : talked*, where, in each pair, the second word is the past tense of the first word. Overall there are 10675 such syntactic pairs of word tuples.

The task here is to find a word d that best fits the following relationship: $a : b :: c : d$ given a, b and c. Mikolov *et al.* (2013b) show that analogous word pairs exhibit linguistic regularity in the vector space, for example:

$$\mathbf{x}_{king} - \mathbf{x}_{man} \approx \mathbf{x}_{queen} - \mathbf{x}_{woman} \tag{3.1}$$

where $\mathbf{x}_w$ represents the vector of word w. Since *king* is the royal form of *man*, the vector

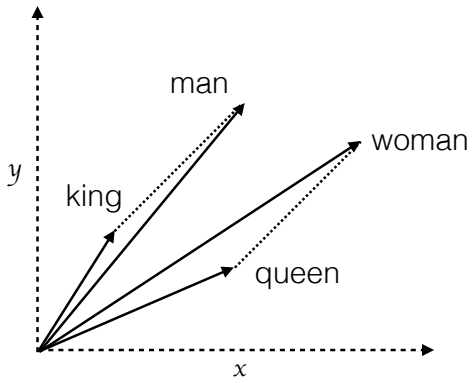

Figure 3.1: Linguistic regularity shown between analogous word pairs. The dotted line connecting the words is the relation vector.

representing the relation between these two words should be the same as that between *queen* and *woman*. Figure 3.1 shows the relationship between these word pairs in the vector space.

We use the vector offset method (Mikolov *et al.*, 2013b) that computes the vector:

$$\mathbf{y} = \mathbf{x}_a - \mathbf{x}_b + \mathbf{x}_c \tag{3.2}$$

where, $\mathbf{x}_a, \mathbf{x}_b$ and $\mathbf{x}_c$ are vectors of words a, b and c respectively and returns the vector $\mathbf{x}_w$ from the whole vocabulary which has the highest cosine similarity to $\mathbf{y}$:

$$\mathbf{x}_w = \underset{w \in V - \{a,b,c\}}{\arg\max} \frac{\mathbf{x}_w \cdot \mathbf{y}}{|\mathbf{x}_w| \cdot |\mathbf{y}|} \tag{3.3}$$

Here, w contains all words in the vocabulary except for the words in the query, i.e, $w \neq \{a, b, c\}$. It is worth noting that this is a non-trivial $|V|$-way classification task where V is the vocabulary of the language.

3.2 Extrinsic Evaluation

We now describe the extrinsic tasks that have been used in this thesis for evaluation of word vectors. Extrinsic evaluation is important because we want to know how useful word vector representations are in downstream tasks independent of their semantic analysis. The tasks described here are chosen to evaluate different aspects of word meaning, such as semantics and syntax.

3.2.1 Sentiment Analysis

Sentiment analysis is the problem of identifying subjective information present in natural language texts and classifying into different levels of polarity. Usually, texts that are indicative of anger, sadness, frustration, unhappiness etc. are labeled as *negative* instances, whereas texts indicative of happiness, joy, satisfaction etc. are labeled as *positive* instances of sentiment. For example, the sentence *"The movie had a great script"* is labeled *positive*, and *"The movie failed to deliver anything exciting"* is labeled *negative*. The task in sentiment analysis is to predict the sentiment polarity label of a given text. We evaluate how informative are the features present in word vector representations in the downstream task of sentiment analysis.

Pang and Lee (2005) created a dataset containing binary sentiment labels on sentences from movie review excerpts. The coarse-grained dataset of positive and negative classes has been split into training, development, and test datasets containing 6,920, 872, and 1,821 sentences, respectively. This dataset was further annotated at the phrase level by Socher *et al.* (2013), but we only use sentence-level annotations. Since our aim is to evaluate the utility of word vectors as features in predicting sentiment, we train an ℓ_2-regularized logistic regression classifier on the average of the word vectors of a given sentence to predict the coarse-grained sentiment tag at the sentence level, and report the test-set accuracy of the classifier. If a sentence is represented as a list of words $\vec{s} = \langle w_1, w_2, \ldots, w_n \rangle$, then the sentence vector can be obtained by averaging the word vectors in the sentence, sometimes called a "bag of means" representation:

$$\mathbf{s} = \frac{\sum_{i=1}^{n} \mathbf{w}_i}{n}. \tag{3.4}$$

3.2.2 Noun-Phrase Bracketing

We now evaluate if word vectors can be useful in predicting the correct parse structure of a noun phrase. *Blood pressure medicine*: A machine to measure the blood pressure or a pressure machine with blood? While our intuition based on the meaning of this phrase prefers the former interpretations, the Stanford parser which lacks semantic features, incorrectly predicts the latter as the correct parse.[2] The correct syntactic parsing of sentences is clearly steered by semantic information (Fillmore *et al.*, 1968), and consequently the semantic plausibility of other possible parses can provide crucial evidence about their validity. We want to see how much of semantic information can be provided by word vector representations to resolve such ambiguity in parsing of noun phrases.

Lazaridou *et al.* (2013) constructed a dataset from the Penn TreeBank (Marcus *et al.*, 1993) of noun phrases (NP) of length three words, where the first can be an adjective or a noun and the other two are nouns. The task is to predict the correct bracketing in the parse tree for a given noun phrase. For example, *local (phone company)* and *(blood pressure) medicine* exhibit *right* and

[2] `http://nlp.stanford.edu:8080/parser/index.jsp`

left bracketing respectively. Since our aim is to evaluate the utility of word vectors as features in predicting the bracketing, we only the word vectors of the three words as features in the model. We append the word vectors of the three words in the noun phrase in order and use them as features in an ℓ_2-regularized logistic regression classifier. The dataset contains 2227 noun phrases split into 10 folds. The classifier is tuned on the first fold and cross-validation accuracy is reported on the remaining nine-folds.

3.3 Intrinsic Evaluation: Qvec

Till now we have discussed various tasks for extrinsic application of word vectors. Despite their ubiquity, there is no standard scheme for intrinsically evaluating the quality of word vectors: vector quality is traditionally judged by its utility in downstream NLP tasks. This lack of standardized evaluation is due, in part, to word vectors' major criticism: word vectors are linguistically opaque in a sense that it is still not clear how to interpret individual vector dimensions, and, consequently, it is not clear how to score a non-interpretable representation. Nevertheless, to facilitate development of better word vector models and for better error analysis of word vectors, it is desirable,

1. To compare word vector models easily, without recourse to multiple extrinsic applications whose implementation and runtime can be costly
2. To understand how features in word vectors contribute to downstream tasks
3. To have a fast-to-compute evaluation metric

We propose a simple intrinsic evaluation measure for word vectors. Our measure is based on component-wise correlations with manually constructed "linguistic" word vectors whose components have well-defined linguistic properties. Since vectors are typically used to provide features to downstream learning problems, our measure favors *recall* (rather than precision), which captures our intuition that meaningless dimensions in induced vector representations are less harmful than important dimensions that are missing. We thus align dimensions in a distributional word vector model with the linguistic dimension vectors to maximize the cumulative correlation of the aligned dimensions. The resulting sum of correlations of the aligned dimensions is our evaluation score. Since the dimensions in the linguistic vectors are linguistically-informed, the alignment provides an "annotation" of components of the word vector space being evaluated. To show that our proposed score is meaningful, we compare our intrinsic evaluation model to the standard (semantic) extrinsic evaluation benchmarks. For nine off-the-shelf word vector representation models, QVEC obtains good correlation ($0.34 \leq r \leq 0.89$) with the extrinsic tasks.

word	nn.animal	nn.food	...	vb.motion
fish	0.68	0.16	...	0.00
duck	0.31	0.00	...	0.69
chicken	0.33	0.67	...	0.00

Table 3.2: Oracle linguistic word vectors, constructed from a linguistic resource containing semantic annotations.

3.3.1 Linguistic Dimension Word Vectors

The crux of our evaluation method lies in quantifying the similarity between a distributional word vector model and a (gold-standard) linguistic resource capturing human knowledge. To evaluate the semantic content of word vectors, we use an existing semantic resource—SemCor (Miller *et al.*, 1993). From the SemCor annotations we construct a set of linguistic word vectors, as explained below. Table 3.2 shows an example of the vectors.

WordNet (Fellbaum, 1998, WN) partitions nouns and verbs into coarse semantic categories known as supersenses (Ciaramita and Altun, 2006; Nastase, 2008).[3] There are 41 supersense types: 26 for nouns and 15 for verbs, for example, NOUN.BODY, NOUN.ANIMAL, VERB.CONSUMPTION, or VERB.MOTION. SemCor is a WordNet-annotated corpus that captures, among others, supersense annotations of WordNet's 13,174 noun lemmas and 5,686 verb lemmas at least once. We construct term frequency vectors normalized to probabilities for all nouns and verbs that occur in SemCor at least 5 times. The resulting set of 4,199 linguistic word vectors has 41 interpretable columns.

While we experiment with linguistic vectors capturing semantic concepts, our methodology is generally applicable to other linguistic resources (Faruqui and Dyer, 2015). For example, part-of-speech annotations extracted from a treebank would yield linguistic vectors capturing syntactic content of vectors. As will be see in the next section that the runtime of QVEC is linear in the number of linguistic dimensions, we start off with just SemCor and defer using other resources to future work.

3.3.2 Qvec

We align dimensions of distributional word vectors to dimensions (linguistic properties) in the linguistic vectors described in §3.3.1 to maximize the cumulative correlation of the aligned dimensions. By projecting linguistic annotations via the alignments, we also obtain plausible annotations of dimensions in the distributional word vectors. In this section, we formally describe the model, which we call QVEC: quantitative and qualitative evaluation of word vectors.

Let the number of common words in the vocabulary of the distributional and linguistic

[3]Supersenses are known as "lexicographer classes" in WordNet documentation, `http://wordnet.princeton.edu/man/lexnames.5WN.html`

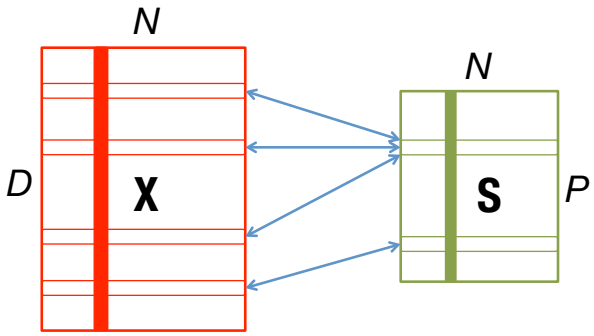

Figure 3.2: The filled vertical vectors represent the word vector in the word vector matrix **X** and the linguistic property matrix **S**. The horizontal hollow vectors represent the "distributional dimension vector" in **X** and "linguistic dimension vector" in **S**. The arrows show mapping between distributional and linguistic vector dimensions.

word vectors be N. We define, the distributional vector matrix $\mathbf{X} \in \mathbb{R}^{D \times N}$ with every row as a dimension vector $\mathbf{x} \in \mathbb{R}^{1 \times N}$. D denotes word vector dimensionality. Similarly, $\mathbf{S} \in \mathbb{R}^{P \times N}$ is the linguistic property matrix with every row as a linguistic property vector $\mathbf{s} \in \mathbb{R}^{1 \times N}$. P is the size of the set of linguistic properties obtained from a manually-annotated linguistic resource.

We obtain an alignment between the word vector dimensions and the linguistic dimensions which maximizes the correlation between the aligned dimensions of the two matrices. This is 1:n alignment: one distributional dimension is aligned to at most one linguistic property, whereas one linguistic property can be aligned to n distributional dimensions, see figure 3.2. Let $\mathbf{A} \in \{0,1\}^{D \times P}$ be a matrix of alignments such that $a_{ij} = 1$ iff $\mathbf{x}_i$ is aligned to $\mathbf{s}_j$, otherwise $a_{ij} = 0$. If $r(\mathbf{x}_i, \mathbf{s}_j)$ is the Pearson's correlation between vectors $\mathbf{x}_i$ and $\mathbf{s}_j$, then our objective is defined as:

$$\text{QVEC} = \max_{\mathbf{A} | \sum_j a_{ij} \leq 1} \sum_{i=1}^{D} \sum_{j=1}^{P} r(\mathbf{x}_i, \mathbf{s}_j) \times a_{ij} \tag{3.5}$$

The constraint $\sum_j a_{ij} \leq 1$, warrants that one distributional dimension is aligned to at most one linguistic dimension. The total correlation between two matrices i.e., QVEC is our intrinsic evaluation measure of a set of word vectors relative to a set of linguistic properties. Equation 3.5 can be solved easily as the objective is maximized when a linguistic dimension is aligned to all possible distributional dimensions with which it has a positive correlation. Thus, $r(\mathbf{x}_i, \mathbf{s}_j) \Rightarrow a_{ij} = 1$. The complexity of the algorithm is $O(DPN)$.

QVEC's underlying hypothesis is that dimensions in distributional vectors correspond to linguistic properties of words. It is motivated, among others, by the effectiveness of word vectors in linear models implying that linear combinations of features (vector dimensions) produce relevant, salient content. Via the alignments a_{ij} we obtain labels on dimensions in the distributional word vectors. The magnitude of the correlation $r(\mathbf{x}_i, \mathbf{s}_j)$ corresponds to the

Model	Qvec	Senti
CBOW	40.3	90.0
SG	35.9	80.5
CWindow	28.1	76.2
SSG	40.5	81.2
Attention	40.8	80.1
GloVe	34.4	79.4
GloVe+WN	42.1	79.6
GloVe+PPDB	39.2	79.7
LSA	19.7	76.9
LSA+WN	29.4	77.5
LSA+PPDB	28.4	77.3
Correlation (r)	**0.87**	

Table 3.3: Intrinsic (QVEC) and extrinsic scores of the 300-dimensional vectors trained using different word vector models and evaluated on the Senti task. Pearson's correlation between the intrinsic and extrinsic scores is $r = 0.87$.

annotation confidence: the higher the correlation, the more salient the linguistic content of the dimension. Clearly, dimensions in the linguistic matrix S do not capture every possible linguistic property, and low correlations often correspond to the missing information in the linguistic matrix. Thus, QVEC is a recall-oriented measure: highly-correlated alignments provide evaluation and annotation of vector dimensions, and missing information or noisy dimensions do not significantly affect the score since the correlations are low.

3.3.3 Evaluation

To test QVEC, we select a diverse suite of popular/state-of-the-art word vector models. All vectors are trained on 1 billion tokens (213,093 types) of English Wikipedia corpus with vector dimensionality 50, 100, 200, 300, 500, 1000. These word vector models are described in appendix A. We compare the QVEC to six standard semantic tasks for evaluating word vectors. In addition to word similarity tasks: WS-353, MEN, SimLex (§3.1.1), sentiment analysis (§3.2.1), we use two additional text classification and metaphor predictions tasks (appendix B) in the evaluation suite.

To test the efficacy of QVEC in capturing the semantic content of word vectors, we evaluate how well QVEC's scores correspond to the scores of word vector models on semantic benchmarks. We compute the Pearson's correlation coefficient r to quantify the linear relationship between the scorings. We begin with comparison of QVEC with one extrinsic task—sentiment analysis (Senti)—evaluating 300-dimensional vectors.

As we show in table 3.3, the Pearson's correlation between the intrinsic and extrinsic scores is $r = 0.87$. To account for variance in WORD2VEC representations (due to their random initial-

	WS-353	MEN	SimLex	20NG	Metaphor	Senti
r	0.34	0.63	0.68	0.74	0.75	0.88

Table 3.4: Pearson's correlations between QVEC scores of the 300-dimensional vectors trained using different word vector models and the scores of the downstream tasks on the same vectors.

	20NG	Metaphor	Senti
WS-353	0.55	0.25	0.46
MEN	**0.76**	0.49	0.55
SimLex	0.56	0.44	0.51
Qvec	0.74	**0.75**	**0.88**

Table 3.5: Pearson's correlations between word similarity/QVEC scores and the downstream text classification tasks.

ization and negative sampling strategies, the representations are different for each run of the model), and to compare QVEC to a larger set of vectors, we now train three versions of vector sets per model. This results in 21 word vector sets: three vector sets per five WORD2VEC models plus GloVe, LSA, and retrofitting vectors (§4.2) shown in table 3.3. The Pearson's correlation computed on the extended set of comparison points (in the same experimental setup as in table 3.3) is $r = 0.88$. In the rest of this section we report results on the extended suite of word vectors.

We now extend the table 3.3 results, and show correlations between the QVEC and extrinsic scores across all benchmarks for 300-dimensional vectors. Table 3.4 summarizes the results. QVEC obtains high positive correlation with all the semantic tasks. Table 3.5 shows, for the same 300-dimensional vectors, that QVEC's correlation with the downstream text classification tasks is on par with or higher than the correlation between the word similarity and text classification tasks. Higher correlating methods—in our experiments, QVEC and MEN—are better predictors of quality in downstream tasks.

Next, we measure correlations of QVEC with the extrinsic tasks across word vector models with different dimensionality. The results are shown in figure 3.3. It can be seen that QVEC shows strong correlation with extrinsic evaluation tasks and the correlations remain almost constant across different dimensionality of word vectors. However, for word similarity tasks the correlation varies as the dimensionality changes, even more so for WS-353. To summarize, we observe high positive correlation between QVEC and the downstream tasks, consistent across the tasks and across different models with vectors of different dimensionalities.

Since QVEC favors recall over precision (for aligning distributional dimensions to linguistic dimensions), larger numbers of dimensions will naturally result in higher scores. We therefore impose the restriction that QVEC only be used to compare vectors of the same size, but we now show that its correlation with downstream tasks is stable, conditional on the size of the

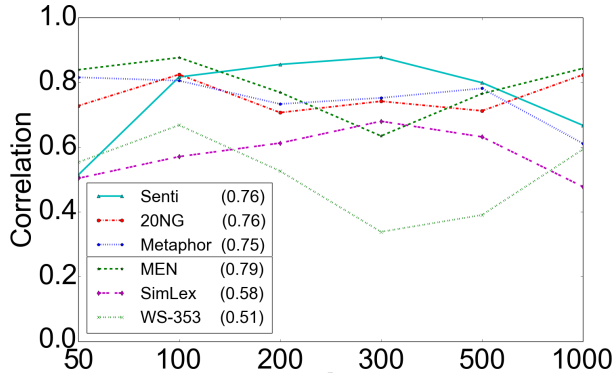

Figure 3.3: Pearson's correlation between QVEC scores and the semantic benchmarks across word vector models on vectors of different dimensionality. The scores at dimension 300 correspond to the results shown in table 3.4. The scores in the legend show average correlation across dimensions.

	50	**100**	**200**	**300**	**500**	**1000**
ρ(QVEC, All)	0.66	0.59	0.63	0.65	0.62	0.59

Table 3.6: Spearman's rank-order correlation between the QVEC ranking of the word vector models and the ranking produced by the aggregated ranking of all tasks across dimensionality size.

vectors being compared. We aggregate rankings by individual downstream tasks into a global ranking using the Kemeny–Young rank aggregation algorithm, for each dimension separately (Kemeny, 1959). The algorithm finds a ranking which minimizes pairwise disagreement of individual rankers. Table 3.6 shows Spearman's rank correlation between the rankings produced by the QVEC and the aggregated ranking across all tasks. The correlation score across different dimensions is fairly stable with a mean of 0.63 and standard deviation of 0.03.

3.3.4 Discussion

Across different experiments we have observed that QVEC is strongly correlated with performance of word vectors on downstream NLP tasks like sentiment analysis, metaphor prediction, and text classification. However, its correlation with traditional word similarity tasks is not consistent across different datasets. Specifically, the lowest correlation scores are obtained for the most widely used task WS-353 (Finkelstein *et al.*, 2002). Recent work on lexical similarity have shown that WS-353 dataset gives high scores to word which are either similar (*car* & *train*) or related (*coffee* & *cup*), and thus it should not be used for measuring word similarity (Agirre *et al.*, 2009; Hill *et al.*, 2014). Such results are confirmed by our experiments which show that QVEC is more strongly correlated with the improved word similarity datasets such as MEN and SimLex which model word similarity exclusively.

QVEC favors recall over precision, thus larger numbers of dimensions will naturally result in higher scores—but not necessarily higher correlations with downstream tasks. Thus, in its current form QVEC can only be used to compare vectors of the same size. QVEC is not scale-invariant, thus an affine transformation of given word vectors will produce a different score. These problems of QVEC have been addressed and resolved by Ammar *et al.* (2016), who use canonical correlation analysis (CCA) to compute the correlation between distributional and linguistic word vector dimensions. The resultant evaluation score is named QVEC+. Since CCA is scale-invariant, and independent of the number of dimensions of the input vectors, the score provided by QVEC+ overcomes the limitations of the QVEC. However, since CCA does not produce explicit alignments between the linguistic and distributional vectors, it's unclear if QVEC+ provides a way of interpretation of distributional dimensions in terms of linguistic properties.

3.4 Conclusion

In this chapter, we first described various evaluation benchmarks that are currently being used for evaluation of word vector representations actively in the research community. We then discussed the limitations associated with evaluation carried out on these benchmarks, which leads us to propose a method for intrinsic evaluation of word vectors. Our method is named QVEC, and it relies on the hypothesis that the uninterpretable dimensions of word vector representations encode certain linguistic proeprties of the words. Thus, QVEC is designed to quantify the linguistic content of word vectors. Our proposed model shows strong relationship—both linear and monotonic—with the scores/rankings produced by the downstream tasks.

Chapter 4

Lexicons

Word lexicons, which provide type-level information about the semantics or syntax of words, should be a valuable resource in constructing word meaning representations. Examples of such word lexicons include WordNet (Miller, 1995), FrameNet (Baker *et al.*, 1998), Paraphrase Database (Ganitkevitch *et al.*, 2013), morpho-syntactic lexicons (Hajič and Hladká, 1998a; Trón *et al.*, 2006), *inter alia*. The information provided by these lexicons might or might not be redundant with distributional evidence – a question that we will explore now. In this chapter, we evaluate the utility of lexicons in constructing and improving word vector representations. We first evaluate how informative are word lexicons in constructing word vector representations in a stand-alone setting i.e., we do not use any sort of distributional evidence in ascertaining the meaning of a word. We show that word vectors constructed using such purely non-distributional information are competitive to methods of obtaining distributional word vectors (§4.1). After establishing the value of lexicons in obtaining word vector representations, we propose a method named "retrofitting" that can use word relations knowledge found in semantic lexicons to improve the quality of distributional word vector representations (§4.2).[1]

4.1 Lexicons as Non-distributional Word Vectors

Our motivation of using linguistic lexicons to construct word vector representations is two-fold. First, we want to evaluate how useful such lexicons are in obtaining the meaning of a word, and how well that competes against various models of distributional word representations. Second, we want to see if we can overcome some of the weaknesses of distributional word vectors models, such as lack of interpretability by constructing vectors from lexicons as discussed next.

Distributional word vectors lie in a continuous space which does not look anything like the representations described in most lexical semantic theories, which focus on identifying

[1]This chapter contains work that was previously published in Faruqui and Dyer (2015) and Faruqui *et al.* (2015).

classes of words (Levin, 1993; Baker *et al.*, 1998; Schuler, 2005; Miller, 1995). Though expensive to construct, conceptualizing word meanings symbolically is important for theoretical understanding and interpretability is desired in computational models. On the surface, discrete theories seem incommensurate with the distributed approach, a problem now receiving much attention in computational linguistics: relating the discrete nature of traditional syntactic and semantic theories with the continuous nature of word representations (Lewis and Steedman, 2013; Kiela and Clark, 2013; Paperno *et al.*, 2014).

Our contribution to this discussion is a new technique that constructs task-independent word vector representations using linguistic knowledge derived from pre-constructed linguistic resources like WordNet (Miller, 1995), FrameNet (Baker *et al.*, 1998), Penn Treebank (Marcus *et al.*, 1993) etc. In such word vectors every dimension is a linguistic feature and 1/0 indicates the presence or absence of that feature in a word, thus the vector representations are binary while being highly sparse ($\approx$ 99.9%). Since these vectors do not encode any word co-occurrence information, they are "non-distributional". An additional benefit of constructing such vectors is that they are fully interpretable i.e, every dimension of these vectors maps to a linguistic feature unlike distributional word vectors where the vector dimensions have no interpretability.

Of course, engineering feature vectors from linguistic resources is established practice in many applications of discriminative learning; e.g., parsing (McDonald and Pereira, 2006; Nivre, 2008) or part of speech tagging (Ratnaparkhi, 1996; Collins, 2002). However, despite a certain common inventories of features that re-appear across many tasks, feature engineering tends to be seen as a task-specific problem, and engineered feature vectors are not typically evaluated independently of the tasks they are designed for. We evaluate the quality of our non-distributional vectors on a number of tasks that have been proposed for evaluating distributional word vectors. We show that non-distributional word vectors are comparable to current state-of-the-art distributional word vectors trained on billions of words.[2]

4.1.1 Feature Extraction

We construct non-distributional word vectors by extracting word level information from linguistic resources. Table 4.1 shows the size of vocabulary and number of features induced from every lexicon. We now describe various linguistic resources that we use for constructing non-distributional word vectors.

WordNet. WordNet (Miller, 1995) is an English lexical database that groups words into sets of synonyms called synsets and records a number of relations among these synsets or their members. For a word we look up its synset for all possible part of speech (POS) tags that it

[2]These vectors can be downloaded at: `https://github.com/mfaruqui/non-distributional`

Lexicon	Vocabulary	Features
WordNet	10,794	92,117
Supersense	71,836	54
FrameNet	9,462	4,221
Emotion	6,468	10
Connotation	76,134	12
Color	14,182	12
Part of Speech	35,606	20
Syn. & Ant.	35,693	75,972
Union	119,257	172,418

Table 4.1: Sizes of vocabulary and features induced from different linguistic resources.

Word	POL.POS	COLOR.PINK	SS.NOUN.FEELING	PTB.VERB	···	ANTO.FAIR
love	1	1	1	1		0
hate	0	0	1	1		0
ugly	0	0	0	0		1
beauty	1	1	0	0		0
refundable	0	0	0	0		0

Table 4.2: Some non-distributional word vectors. 1 indicates presence and 0 indicates absence of a linguistic feature.

can assume. For example, *film* will have SYNSET.FILM.V.01 and SYNSET.FILM.N.01 as features as it can be both a verb and a noun. In addition to synsets, we include the hyponym (for ex. HYPO.COLLAGEFILM.N.01), hypernym (for ex. HYPER:SHEET.N.06) and holonym set of the word as features. We also collect antonyms and pertainyms of all the words in a synset and include those as features in the linguistic vector.

Supersenses. WordNet partitions nouns and verbs into semantic field categories known as supersenses (Ciaramita and Altun, 2006; Nastase, 2008). For example, *lioness* evokes the supersense SS.NOUN.ANIMAL. These supersenses were further extended to adjectives (Tsvetkov *et al.*, 2014a).[3] We use these supsersense tags for nouns, verbs and adjectives as features in the non-distributional word vectors.

FrameNet. FrameNet (Baker *et al.*, 1998; Fillmore *et al.*, 2003) is a rich linguistic resource that contains information about lexical and predicate-argument semantics in English. Frames can be realized on the surface by many different word types, which suggests that the word types evoking the same frame should be semantically related. For every word, we use the frame it evokes along with the roles of the evoked frame as its features. Since, information in FrameNet is part of speech (POS) disambiguated, we couple these feature with the corresponding POS tag of the word. For example, since *appreciate* is a verb, it will have the following features:

[3] `http://www.cs.cmu.edu/~ytsvetko/adj-supersenses.tar.gz`

VERB.FRAME.REGARD, VERB.FRAME.ROLE.EVALUEE etc.

Emotion & Sentiment. Mohammad and Turney (2013) constructed two different lexicons that associate words to sentiment polarity and to emotions respectively using crowdsourcing. The polarity is either positive or negative but there are eight different kinds of emotions like anger, anticipation, joy etc. Every word in the lexicon is associated with these properties. For example, *cannibal* evokes POL.NEG, EMO.DISGUST and EMO.FEAR. We use these properties as features in non-distributional vectors.

Connotation. Feng *et al.* (2013) construct a lexicon that contains information about connotation of words that are seemingly objective but often allude nuanced sentiment. They assign positive, negative and neutral connotations to these words. This lexicon differs from Mohammad and Turney (2013) in that it has a more subtle shade of sentiment and it extends to many more words. For example, *delay* has a negative connotation CON.NOUN.NEG, *floral* has a positive connotation CON.ADJ.POS and *outline* has a neutral connotation CON.VERB.NEUT.

Color. Most languages have expressions involving color, for example *green with envy* and *grey with uncertainly* are phrases used in English. The word-color association lexicon produced by Mohammad (2011) using crowdsourcing lists the colors that a word evokes in English. We use every color in this lexicon as a feature in the vector. For example, COLOR.RED is a feature evoked by the word *blood*.

Part of Speech Tags. The Penn Treebank (Marcus *et al.*, 1993) annotates naturally occurring text for linguistic structure. It contains syntactic parse trees and POS tags for every word in the corpus. We collect all the possible POS tags that a word is annotated with and use it as features in the non-distributional vectors. For example, *love* has PTB.NOUN, PTB.VERB as features.

Synonymy & Antonymy. We use Roget's thesaurus (Roget, 1852) to collect sets of synonymous words.[4] For every word, its synonymous word is used as a feature in the linguistic vector. For example, *adoration* and *affair* have a feature SYNO.LOVE, *admissible* has a feature SYNO.ACCEPTABLE. The synonym lexicon contains 25,338 words after removal of multiword phrases. In a similar manner, we also use antonymy relations between words as features in the word vector. The antonymous words for a given word were collected from Ordway (1913).[5] An example would be of *impartiality*, which has features ANTO.FAVORITISM and ANTO.INJUSTICE. The antonym lexicon has 10,355 words. These features are different from those induced from

[4] http://www.gutenberg.org/ebooks/10681

[5] https://archive.org/details/synonymsantonyms00ordwiala

Vector	Length	Corpus Size	WS-353	MEN	SimLex	Senti	NP	QVEC
SG	300	300 billion	65.6	67.8	43.6	**81.5**	80.1	34.6
Glove	300	6 billion	60.5	**73.7**	36.9	77.7	77.9	34.2
LSA	300	1 billion	**67.3**	70.5	49.6	81.1	79.7	35.0
Ling Sparse	172,418	–	44.6	45.3	56.6	79.4	**83.3**	–
Ling Dense	300	–	45.4	43.8	**57.8**	75.4	76.2	**37.9**
SG $\oplus$ Ling Sparse	172,718	–	67.1	69.1	55.5	82.4	82.8	–

Table 4.3: Performance of different type of word vectors on evaluation tasks reported by Spearman's correlation (first 3 columns), Accuracy (next 2 columns), and QVEC score. Bold shows the best performance for a task.

WordNet as the former encode word-word relations whereas the latter encode word-synset relations.

After collecting features from the various linguistic resources described above we obtain non-distributional word vectors of length 172,418 dimensions. These vectors are 99.9% sparse i.e, each vector on an average contains only 34 non-zero features out of 172,418 total features. On average a linguistic feature (vector dimension) is active for 15 word types. The non-distributional word vectors contain 119,257 unique word types. Table 4.2 shows non-distributional vectors for some of the words.

4.1.2 Evaluation

In order to make our non-distributional vectors comparable to publicly available distributional word vectors, we perform singular value decompostion (SVD) on the linguistic matrix to obtain word vectors of lower dimensionality. If $\mathbf{L} \in \{0,1\}^{N \times D}$ is the linguistic matrix with N word types and D linguistic features, then we can obtain $\mathbf{U} \in \mathbb{R}^{N \times K}$ from the truncated SVD of $\mathbf{L}$ as follows: $\mathbf{L} = \mathbf{U\Sigma V}^\top$, with K being the desired length of the lower dimensional space. We compare both sparse and dense non-distributional vectors to three widely used distributional word vector models. The first two are the pre-trained Skip-Gram (Mikolov *et al.*, 2013a)[6] and Glove (Pennington *et al.*, 2014)[7] word vectors each of length 300, trained on 300 billion and 6 billion words respectively. We used truncated SVD to obtain word vectors trained on 1 billion words of Wikipedia with vector length 300 and context window of 5 words (cf. appendix A).

Table 4.3 shows the performance of different word vector types on the evaluation tasks. It can be seen that although Skip-Gram, Glove & LSA perform better than non-distributional vectors on WS-353, the non-distributional vectors outperform them by a huge margin on SimLex. On sentiment analysis, non-distributional vectors are competitive with Skip-Gram vectors and on the NP-bracketing task they outperform all distributional vectors with a statistically significant margin ($p < 0.05$, McNemar's test (Dietterich, 1998)). On QVEC, the non-distributional

[6] `https://code.google.com/p/word2vec`

[7] `http://www-nlp.stanford.edu/projects/glove/`

dense vectors perform better than all the distributional vectors. This can be expected as the non-distributional vectors are constructed with features extracted from WordNet, which are used in the QVEC linguistic matrix.[8] We append the sparse non-distributional vectors to Skip-Gram vectors and evaluate the resultant vectors as shown in the bottom row of Table 4.3. The combined vector outperforms Skip-Gram on all tasks, showing that non-distributional vectors contain useful information orthogonal to distributional information.

4.1.3 Discussion

It is evident from the results that non-distributional vectors are either competitive or better to state-of-the-art distributional vector models. Non-distributional word vectors are sparse, which makes them computationally easy to work with, despite the large number of dimensions. In contrast to distributional word vectors, non-distributional word vectors have interpretable dimensions as every dimension is a linguistic property. Non-distributional word vectors require no training as there are no parameters to be optimized, meaning they are computationally economical.

An obvious shortcoming of non-distributional word vectors is that their coverage is limited to only words that have been observed in the available lexicons. This limits the extension of obtaining non-distributional word vectors to different languages, and good quality non-distributional word vectors may only be obtained for languages with rich linguistic resources. However, it's worth noting that such resources do exist in many major languages like English, French, German, Chinese. and thus non-distributional vectors can be constructed for these languages in future. Nonetheless, we have established that word lexicons can be used to construct sparse, and interpretable word vectors that are competitive to current state-of-the-art models of distributional word representations.

4.2 Lexicons as Additional Evidence

Having established the importance of lexicons as sources of information in obtaining word vector representations in the previous section, in this section we present a method that can use such rich information present in word lexicons to improve the quality of distributional word vectors. Even though the existing models of distributional word vectors have shown to be extremely effective in capturing word meaning, we argue that the information contained in word lexicons can still provide guidance to such distributional word vectors to enhance their quality.

Word lexicons provide information about the relations between a word and other words. Such kind of relational information can be useful in identifying words which should have

[8] QVEC can only be used to compare vectors with equal dimensionality in the version presented in this thesis.

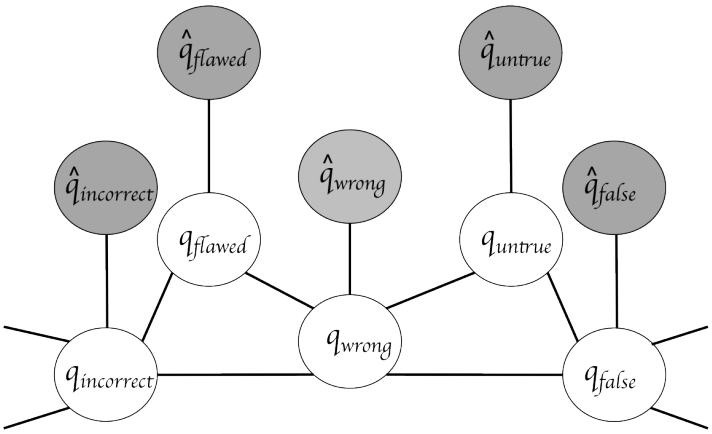

Figure 4.1: Word graph with edges between related words showing the observed and the inferred word vector representations.

similar (or different) word vectors. For exmaple, if we know that words like *false* and *incorrect* are synonyms, then we would expect them to have word vectors that lie close together in the vector space. We want to enforce such constraints on word vectors using information obtained from word lexicons in order to improve the quality of word vectors.

Recent work has shown that by either changing the objective of word vector training algorithms in neural language models (Yu and Dredze, 2014; Xu *et al.*, 2014; Bian *et al.*, 2014) or by relation-specific augmentation of the co-occurence matrix in spectral word vector models (Yih *et al.*, 2012; Chang *et al.*, 2013) to incorporate semantic knowledge, the quality of word vectors can be improved. However, these methods are limited to particular methods for constructing vectors. We present a graph-based learning technique for leveraging lexical relational resources to obtain higher quality semantic vectors, which we call "retrofitting" (Faruqui *et al.*, 2015). In contrast to previous work, retrofitting is applied as a *post-processing step* by running belief propagation on a graph constructed from lexicon-derived relational information to update word vectors (§4.2). The vectors obtained after applying retrofitting are called retrofitted vectors. Intuitively, this method encourages the retrofitted vectors to be:

- similar to retrofitted vectors of related word types
- similar to their purely distributional vectors

Thus, we extend the distributional hypothesis by showing that in addition to monolingual word co-occurrences, the words related to a given word in semantic lexicons also help in evidencing the meaning of the word.

4.2.1 Retrofitting

Let $V = \{w_1, \ldots, w_n\}$ be a **vocabulary**, i.e, the set of word types, and Ω be an **ontology** that encodes semantic relations between words in V. We represent Ω as an undirected graph (V, E) with one vertex for each word type and edges $(w_i, w_j) \in E \subseteq V \times V$ indicating a semantic relationship of interest.

The matrix $\hat{\mathbf{Q}}$ will be the collection of vector representations $\hat{q}_i \in \mathbb{R}^d$, for each $w_i \in V$, learned using a standard data-driven technique, where d is the length of the word vectors. Our objective is to learn the matrix $\mathbf{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_n)$ such that the columns are both close (under a distance metric) to their counterparts in $\hat{\mathbf{Q}}$ *and* to adjacent vertices in Ω. Figure 4.1 shows a small word graph with such edge connections; white nodes are labeled with the $\mathbf{Q}$ vectors to be retrofitted (and correspond to V_Ω); shaded nodes are labeled with the corresponding vectors in $\hat{\mathbf{Q}}$, which are observed. The graph can be interpreted as a Markov random field (Kindermann and Snell, 1980).

The distances between a pair of vectors is defined to be the Euclidean distance. Since we want the inferred word vector to be close to the observed value $\hat{\mathbf{q}}_i$ and close to its neighbors $\mathbf{q}_j, \forall j$ such that $(i, j) \in E$, the objective to be minimized becomes:

$$\Psi(\mathbf{Q}) = \sum_{i=1}^{n} \left[\alpha_i \|\mathbf{q}_i - \hat{\mathbf{q}}_i\|^2 + \sum_{(i,j)\in E} \beta_{ij} \|\mathbf{q}_i - \mathbf{q}_j\|^2 \right]$$

where α and β values control the relative strengths of associations.

In this case, we first train the word vectors independent of the information in the semantic lexicons and then retrofit them. Ψ is convex in $\mathbf{Q}$ and its solution can be found by solving a system of linear equations. However, since there are a very large number of variables ($n \times d$), directly solving this equation is expensive, so we use an efficient iterative updating method (Bengio *et al.*, 2006; Subramanya *et al.*, 2010; Das and Petrov, 2011; Das and Smith, 2011). We take the first derivative of Ψ with respect to one $\mathbf{q}_i$ vector, and by equating it to zero arrive at the following update rule:

$$\mathbf{q}_i = \frac{\sum_{j:(i,j)\in E} \beta_{ij} \mathbf{q}_j + \alpha_i \hat{\mathbf{q}}_i}{\sum_{j:(i,j)\in E} \beta_{ij} + \alpha_i} \tag{4.1}$$

In practice, running this procedure for 10 iterations converges to changes in Euclidean distance of adjacent vertices of less than 10^{-2}. The vectors in $\mathbf{Q}$ are initialized to be equal to the vectors in $\hat{\mathbf{Q}}$. The retrofitting approach described above can be applied to word vector representations obtained from any model as the updates in Eq. 4.1 are agnostic to the original vector training model objective.

Retrofitting, in its current form as described in this thesis, is designed only to work with words having similar meaning and can only pull word vectors with similar meaning closer

Lexicon	Words	Edges
PPDB	102,902	374,555
WordNet$_{syn}$	148,730	304,856
WordNet$_{all}$	148,730	934,705
FrameNet	10,822	417,456

Table 4.4: Size of the graphs obtained from different lexicons.

in the vector space. However, it has been extended to incorporate other relation types which signify change in meaning, such an antonymous relations as described in Mrkšić *et al.* (2016), a method they call "counter-fitting". Similar to retrofitting, counter-fitting pulls word vectors of words with similar meaning together but also includes another term in the objective which pushes away word vectors of antonymous words.

4.2.2 Semantic Lexicons during Learning.

Our proposed approach is reminiscent of recent work on improving word vectors using lexical resources (Yu and Dredze, 2014; Bian *et al.*, 2014; Xu *et al.*, 2014) which alters the learning objective of the original vector training model with a prior (or a regularizer) that encourages semantically related vectors (in Ω) to be close together, except that our technique is a applied after learning, rather than in conjunction with learning. We describe the prior approach here since it will serve as a baseline. Here semantic lexicons play the role of a prior on $\mathbf{Q}$ which we define as follows:

$$p(\mathbf{Q}) \propto \exp\left(-\gamma \sum_{i=1}^{n} \sum_{j:(i,j)\in E} \beta_{ij} \|\mathbf{q}_i - \mathbf{q}_j\|^2\right) \tag{4.2}$$

Here, γ is a hyper-parameter that controls the strength of the prior. As in the retrofitting objective, this prior on the word vector parameters forces words connected in the lexicon to have close vector representations as did $\Psi(\mathbf{Q})$ (with the role of $\hat{\mathbf{Q}}$ being played by cross entropy of the empirical distribution).

This prior can be incorporated during learning through maximum a posteriori (MAP) estimation. Since there is no closed form solution of the estimate, we consider two iterative procedures. In the first, we use the sum of gradients of the log-likelihood (given by the extant vector learning model) and the log-prior probability (from Eq. 4.2), with respect to $\mathbf{Q}$ for learning. Since computing gradient of Eq. 4.2 is linear in the vocabulary size n, we use lazy updates (Carpenter, 2008) for every k words during training. We call this the **lazy** method of MAP. The second technique applies stochastic gradient ascent to the log-likelihood, and after every k words applies the update in Eq. 4.1. We call this the **periodic** method.

4.2.3 Graph Construction

We use three different semantic lexicons to evaluate their utility in improving the word vectors. We construct word graphs using these lexicons which are then used for retrofitting. Table 5.2 shows the size of the graphs obtained from these lexicons.

PPDB. The paraphrase database Ganitkevitch *et al.* (2013) is a semantic lexicon containing more than 220 million paraphrase pairs of English.[9] Of these, 8 million are lexical (single word to single word) paraphrases. The key intuition behind the acquisition of its lexical paraphrases is that two words in one language that align, in parallel text, to the same word in a different language, should be synonymous. For example, if the words *jailed* and *imprisoned* are translated as the same word in another language, it may be reasonable to assume they have the same meaning. In our experiments, we instantiate an edge in E for each lexical paraphrase in PPDB. The lexical paraphrase dataset comes in different sizes ranging from S to XXXL, in decreasing order of paraphrasing confidence and increasing order of size. We chose XL for our experiments. We want to give higher edge weights (α_i) connecting the retrofitted word vectors (q) to the purely distributional word vectors ($\hat{q}$) than to edges connecting the retrofitted vectors to each other (β_{ij}), so all α_i are set to 1 and β_{ij} to be degree$(i)^{-1}$ (with i being the node the update is being applied to).[10]

WordNet. We extract pairs of words from WordNet which are connected by synonymy, hypernymy, or hyponymy relations. For example, the word *dog* is a synonym of *canine*, a hypernym of *puppy* and a hyponym of *animal*. We perform two different experiments with WordNet: (1) connecting a word only to synonyms, and (2) connecting a word to synonyms, hypernyms and hyponyms.[11] We refer to these two graphs as WN_{syn} and WN_{all}, respectively. In both settings, all α_i are set to 1 and β_{ij} to be degree$(i)^{-1}$.

FrameNet. In predicate-argument semantics, a frame can be realized on the surface by many different word type. For example, the frame Cause_change_of_position_on_a_scale is associated with *push*, *raise*, and *growth* (among many others). In our use of FrameNet, two words that group together with any frame are given an edge in E. We refer to this graph as FN. All α_i are set to 1 and β_{ij} to be degree$(i)^{-1}$.

[9] `http://www.cis.upenn.edu/~ccb/ppdb`

[10] In principle, these hyperparameters can be tuned to optimize performance on a particular task, which we leave for future work.

[11] We ignore the word senses present in WordNet, thus merging all senses of the word together. Sense-specific retrofitting was extended in Jauhar *et al.* (2015).

Lexicon	MEN	SimLex	WS-353	SYN-REL	Senti
Glove	73.7	37.0	60.5	67.0	79.6
+PPDB	1.4	**12.5**	-1.2	-0.4	**1.6**
+WN$_{syn}$	0.0	7.3	0.5	-12.4	0.7
+WN$_{all}$	**2.2**	10.0	**0.7**	-8.4	0.5
+FN	-3.6	-0.9	-5.3	-7.0	0.0
SG	67.8	43.6	65.6	73.9	81.2
+PPDB	**5.4**	**11.0**	**4.4**	-2.3	**0.9**
+WN$_{syn}$	0.7	3.2	0.0	-13.6	0.7
+WN$_{all}$	2.5	7.6	1.9	-10.7	-0.3
+FN	-3.2	-3.0	-4.9	-7.3	0.5
GC	31.3	9.7	62.3	10.9	67.8
+PPDB	**7.0**	**12.1**	2.0	**5.3**	**1.1**
+WN$_{syn}$	3.6	6.7	0.6	-1.7	0.0
+WN$_{all}$	6.7	7.5	**2.3**	-0.6	0.2
+FN	1.8	3.8	0.0	-0.6	0.2
Multi	75.8	28.7	68.1	45.5	81.0
+PPDB	**3.8**	**12.9**	**6.0**	**4.3**	0.6
+WN$_{syn}$	1.2	6.9	2.2	-12.3	**1.4**
+WN$_{all}$	2.9	9.3	4.3	-10.6	**1.4**
+FN	1.8	2.5	0.0	-0.6	0.2

Table 4.5: Absolute performance changes with retrofitting. Spearman's correlation (3 left columns) and accuracy (2 right columns) on different tasks. Bold indicates highest improvement for a vector type.

4.2.4 Evaluation

We use retrofitting to improve four different pre-trained vectors: Glove, Skip-Gram (SG), Global Context (GC) and Cross-lingually enriched (Multi, obtained as part of this thesis, details in §6.2). These vectors are described in appendix A.

Retrofitting Results. Table 4.5 shows the absolute changes in performance on different tasks (as columns) with different semantic lexicons (as rows). All of the lexicons offer high improvements on the word similarity tasks (the first three columns). For the extrinsic sentiment analysis task, we get improvements using all the lexicons and gain 1.4% (absolute) in accuracy for the Multi vectors over the baseline. This increase is statistically significant ($p < 0.01$) according to McNemar's test.

We get improvements on all tasks except for SYN-REL, over Glove and SG vectors, which were trained on billions of tokens. For higher baselines (Glove and Multi) we get smaller improvements as compared to lower baseline scores (SG and GC). We believe that FrameNet does not perform as well as the other lexicons because its frames group words based on abstract concepts; often words with seemingly distantly related meanings (e.g., *push* and *growth*) can evoke the same frame. Interestingly, we almost never improve on the SYN-REL task, es-

Vectors	QVEC
SG	34.6
SG + PPDB	38.5
SG + WordNet$_{sym}$	37.9
SG + WordNet$_{all}$	**41.0**
SG + FrameNet	37.1

Table 4.6: QVEC score for intrinsic evaluation of skip-gram vectors before and after retrofitting with different lexicons (higher is better).

pecially with higher baselines, this can be attributed to the fact that SYN-REL is inherently a syntactic task and during retrofitting we are including semantic information in the vectors. In summary, we find that PPDB gives the best graph among those tested, closely followed by WN$_{all}$ and retrofitting gives consistent gains across tasks and vectors.

Intrinsic evaluation with Qvec. We proposed QVEC as an intrinsic evaluation metric for word vectors in section 3.3. QVEC measures how well the dimensions in distributional word vectors quantify the linguistic content of the words. Table 4.6 shows the QVEC score for skip-gram vectors, before and after retrofitting. Using retrofitting the score increases using all the semantic lexicons, and the highest improvement is obtained for retrofitting with WordNet$_{all}$, followed by PPDB. This is expected as the linguistic properties used inside QVEC contain only supersenes at the moment, which are derived from WordNet.

Semantic Lexicons during Learning. To incorporate lexicon information during training, and compare its performance against retrofitting, we train log-bilinear (LBL) vectors (Mnih and Teh, 2012). These vectors are trained to optimize the log-likelihood of a language model which predicts a word token w's vector given the set of words in its context (h), also represented as vectors:

$$p(w \mid h; \mathbf{Q}) \propto \exp\left(\sum_{i \in h} \mathbf{q}_i^\top \mathbf{q}_j + b_j\right) \tag{4.3}$$

We optimize the above defined posterior along with the prior defined in eq. 4.2 using the two **lazy** and **periodic** techniques mentioned in §4.2. Since it is costly to compute the partition function over the whole vocabulary, we use *noise contrastive estimation* (NCE) to estimate the parameters of the model (Mnih and Teh, 2012) using AdaGrad (Duchi *et al.*, 2010) with a learning rate of 0.05.

We train vectors of length 100 on the WMT-2011 news corpus, which contains 360 million words and use PPDB as the semantic lexicon as it performed reasonably well in the retrofitting experiments. For the **lazy** method we update the prior every $k = 100{,}000$ words[12] and test for different values of prior strength $\gamma \in \{1, 0.1, 0.01\}$. For the **periodic** method, we update

[12] k = 10000, 50000 yielded similar results.

Method	k/γ	MEN	WS-353	SYN-REL	Senti
LBL (Baseline)	$k = \infty, \gamma = 0$	58.0	53.6	31.5	72.5
LBL + Lazy	$\gamma = 1$	-0.4	0.6	0.6	1.2
	$\gamma = 0.1$	0.7	0.4	0.7	0.8
	$\gamma = 0.01$	0.7	1.7	1.9	0.4
LBL + Periodic	$k = 100$M	3.8	3.6	4.8	1.3
	$k = 50$M	3.4	4.4	0.6	**1.9**
	$k = 25$M	0.5	2.7	-3.7	0.8
LBL + Retrofitting	–	**5.7**	**5.5**	**14.7**	0.9

Table 4.7: Absolute performance changes for including PPDB information while training LBL vectors. Spearman's correlation (2 left columns) and accuracy (2 right columns) on different tasks. Bold indicates highest improvement.

Corpus	Vector Training	MEN	WS-353	SYN-REL	Senti
WMT-11	CBOW	55.2	54.7	40.8	74.1
	Yu and Dredze (2014)	50.1	53.7	29.9	71.5
	CBOW + Retrofitting	**60.5**	**58.4**	**52.5**	**75.7**
Wikipedia	SG	**76.1**	67.8	40.3	73.1
	Xu *et al.* (2014)	–	68.3	44.4	–
	SG + Retrofitting	65.7	**69.0**	**49.9**	**74.6**

Table 4.8: Comparison of retrofitting for semantic enrichment against Yu and Dredze (2014), Xu *et al.* (2014). Spearman's correlation (2 left columns) and accuracy (2 right columns) on different tasks.

the word vectors using Eq. 4.1 every $k \in \{25, 50, 100\}$ million words. See Table 4.7. For **lazy**, $\gamma = 0.01$ performs best, but the method is in most cases not highly sensitive to γ's value. For **periodic**, which overall leads to greater improvements over the baseline than **lazy**, $k = 50$M performs best, although all other values of k also outperform the the baseline. It can be seen that retrofitting performs either better or competitively to using semantic knowledge during training.

Comparisons to Prior Work. Two previous models that leverage semantic information from lexicons (Yu and Dredze, 2014; Xu *et al.*, 2014) have shown that the quality of word vectors obtained using `word2vec` tool can be improved by using semantic knowledge from lexicons. Both these models use constraints among words as a regularization term on the training objective during training, and their method can only be applied for improving the quality of SG and CBOW vectors produced by the `word2vec` tool.

We first compare our model to Yu and Dredze (2014). We train word vector using their joint model training code[13] while using exactly the same training settings as specified in their best model: CBOW model, vector length 100 and PPDB for enrichment. The results are shown in the top half of Table 4.8 where our model consistently outperforms the baseline and their

[13] `https://github.com/Gorov/JointRCM`

Language	Task	SG	Retrofitted SG
German	RG-65	53.4	**60.3**
French	RG-65	46.7	**60.6**
Spanish	MC-30	54.0	**59.1**

Table 4.9: Spearman's correlation for word similarity evaluation using the using original and retrofitted SG vectors.

model. Next, we compare our model to Xu *et al.* (2014). This model extracts categorical and relational knowledge among words from Freebase[14] and uses it as a constraint while training. We train the SG model by using exactly the same settings as described in their system (vector length 300) and on the same corpus: monolingual English Wikipedia text.[15] We compare the performance of our retrofitting vectors on the SYN-REL and WS-353 task against the best model[16] reported in their paper. As shown in the lower half of Table 4.8, our model outperforms their model by an absolute 5.5 points absolute on the SYN-REL task and obtains a gain on the WS-353 task as well.

Multilingual Evaluation. We tested our method on three additional languages: German, French, and Spanish. We used the Universal WordNet (de Melo and Weikum, 2009), an automatically constructed multilingual lexical knowledge base based on WordNet.[17] It contains words connected via different lexical relations to other words both within and across languages. We construct separate graphs for different languages (i.e., only linking words to other words in the same language) and apply retrofitting to each. Since not many word similarity evaluation benchmarks are available for languages other than English, we tested our baseline and improved vectors on one benchmark per language.

We used RG-65 (Gurevych, 2005), RG-65 (Joubarne and Inkpen, 2011) and MC-30 (Hassan and Mihalcea, 2009) for German, French and Spanish, respectively. These benchmarks were created by translating the corresponding English benchmarks word by word manually. We trained SG vectors for each language of length 300 on a corpus of 1 billion tokens, each extracted from Wikipedia, and evaluate them on word similarity on the benchmarks before and after retrofitting. Table 4.9 shows that we obtain high improvements which strongly indicates that our method generalizes across these languages.

4.2.5 Analysis

Retrofitting vs. vector length. Upon increasing vector length, the vector might be able to capture higher orders of semantic information and retrofitting might have a diminishing effect.

[14] `https://www.freebase.com/`

[15] `http://mattmahoney.net/dc/enwik9.zip`

[16] Their best model is named RC-NET in their paper.

[17] `http://www.mpi-inf.mpg.de/yago-naga/uwn`

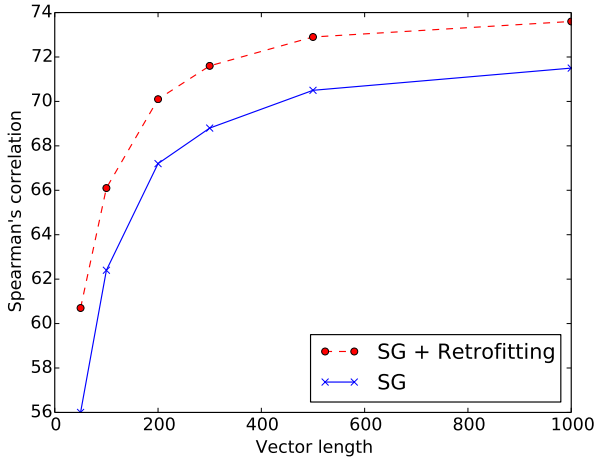

Figure 4.2: Spearman's correlation on SG and SG + Retrofitting vectors on **MEN** word similarity task.

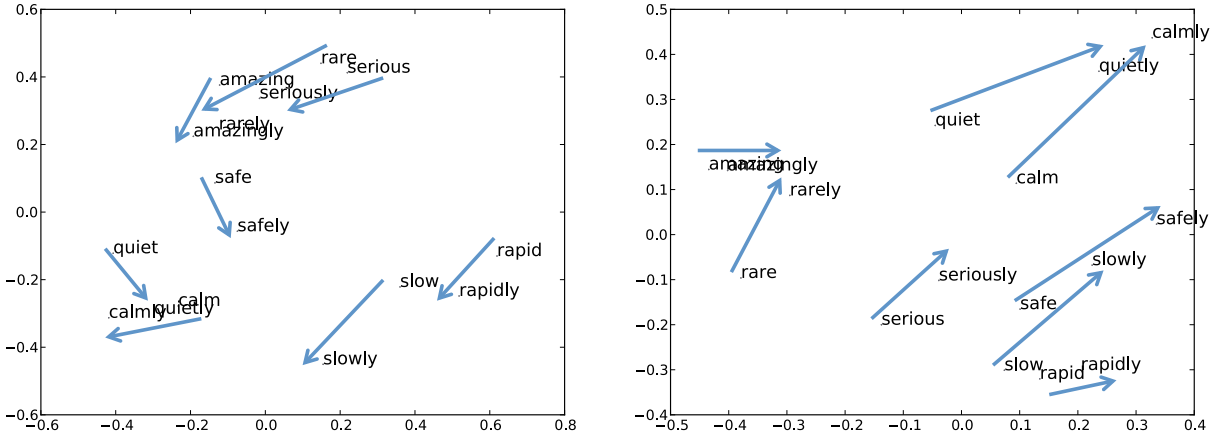

Figure 4.3: Two-dimensional PCA projections of 100-dimensional **SG** vectors of syntactic analogy "adjective to adverb" relation, before (left) and after (right) retrofitting.

To confirm the robustness of retrofitting, we train SG vectors on 1 billion English tokens for vector lengths ranging from 50 to 1000 and evaluate it on the **MEN** word similarity task. We retrofit these vectors to PPDB and evaluate those on the same task. Figure 4.2 shows that by retrofitting we observe consistent improvement in vector quality for different vector lengths.

Qualitative visualization. We randomly pick eight word pairs that follow the "adjective to adverb" relation from the SYN-REL task. We then take a two-dimensional PCA projection of these word vectors and plot them in $\mathbb{R}^2$. In figure 4.3 we plot 100 dimensional SG vectors before (left) and after (right) retrofitting. It can be seen that in the first case the direction of the analogy vectors is not consistent but after retrofitting all the analogy vectors are aligned

in the same direction.

4.3 Conclusion

In this chapter, we showed that information stored in word lexicons is useful in obtaining word vectors, and improving existing distributional word vectors. We first showed that we can construct non-distributional word vectors by collecting information stored in word lexicons. These word vectors are highly sparse and binary, where every dimension represents a linguistic attribute. Thus, such vectors are fully interpretable. When compared against standard distributional word vector models, these vectors exhibited comparable performance on a variety of evaluation benchmarks.

Having shown the importance of word lexicons on constructing word vector representations, we next presented a method called retrofitting, that can improve the quality of distributional word vectors by incorporating information stored in semantic lexicons. Retrofitting is used as a post-processing step to improve vector quality and is simpler to use than other approaches that use semantic information while training. It can be used for improving vectors obtained from any word vector training model and performs better than current approaches to semantic enrichment of word vectors. In summary, we have shown how word vector representations obtained using monolingual distributional information can be further improved by using evidence from a word's neighborhood in a semantic lexicon. Firth might paraphrase this as *"You shall know a word by its introduction from a lexicographer"*.

Chapter 5

Morphology

In most languages many words can be related to other words by rules that collectively describe the grammar for that language. For example, English speakers recognize that the words *play* and *played* are closely related, differentiated only by the inflection corresponding to the past tense marker suffix "-ed". Such relationships between words are a byproduct of the compositional structure of the morphemes in a word (Nida, 1949; Bauer, 2003). These morphemes present in a word combine together to generate the meaning of a word. Inflection is the word-formation mechanism to express different grammatical categories such as tense, mood, voice, aspect, person, gender, number, case, *inter alia*. Inflectional morphology is often realized by the concatenation of bound morphemes (prefixes and suffixes) to a root form or stem, but nonconcatenative processes such as ablaut and infixation are found in many languages as well (Stump, 2001). Thus, the morphological structure of a word provides informative signals about its meaning. Firth might paraphrase this as *"You shall know a word by its morphological structure"*.[1]

Till now in the thesis, we have explored what information about a word's meaning can be obtained by looking at its neighboring words in its distributional context, and by looking at its neighboring words in semantic lexicons. In other words, we have looked for evidence of a word's meaning outside the word. In this chapter we instead will look inside the word i.e., we will look at the morphological structure of the word to gain evidence about the word's meaning. In particular, we will be asking the following questions: Does inflectional morphology provide information about word's meaning? Is this information useful in addition to distributional information, or is it redundant? How can we use such source of information to obtain word representations?

[1]This chapter contains work that was previously published in Faruqui *et al.* (2016).

played	POS:Verb, Tense:Past, VForm:Fin, VForm:Part, Voice:Pass, Mood:Ind
playing	POS:Verb, POS:Noun, Tense:Pres, VForm:Ger, Num:Sing, VForm:part
awesome	POS:Adj, Degree:Pos
Paris	POS:Propn, Num:Sing

Table 5.1: A sample English morpho-syntactic lexicon.

5.1 Morpho-syntactic Lexicons

There exist manually constructed lexicons that contain information about the morphological attributes and syntactic roles of words in a given language. These lexicons are generally called "morpho-syntactic lexicons". A typical lexicon contains all possible attributes that can be displayed by a word. Table 5.1 shows some entries in a sample English morpho-syntactic lexicon. The word *playing* can act both as POS:Verb, and POS:Noun in the language, and thus both the attributes are present in the lexicon. A morpho-syntactic lexicon, by definition, contains all possible attributes, and only a subset of these are displayed by a word in a particular context. As morpho-syntactic lexicons contain rich linguistic information, they are useful as features in downstream NLP tasks like machine translation (Nießen and Ney, 2004; Minkov *et al.*, 2007; Green and DeNero, 2012), part of speech tagging (Schmid, 1994; Denis *et al.*, 2009; Moore, 2015), dependency parsing (Goldberg *et al.*, 2009), language modeling (Arisoy *et al.*, 2010) and morphological tagging (Müller and Schuetze, 2015) *inter alia*. However, there are three major factors that limit the use of such lexicons in real world applications:

1. They are often constructed manually and are expensive to obtain (Kokkinakis *et al.*, 2000; Dukes and Habash, 2010)

2. They are currently available for only a few languages

3. Size of the available lexicons is generally small

We present a method that takes as input a small seed lexicon (like table 5.1), containing a few thousand annotated words, and outputs an automatically constructed lexicon which contains morpho-syntactic attributes (henceforth referred to as attributes) for a large number of words of a given language. We model the problem of morpho-syntactic lexicon generation as a graph-based semi-supervised learning problem (Zhu *et al.*, 2005; Bengio *et al.*, 2006; Subramanya and Talukdar, 2014). We construct a graph where nodes represent word types and the goal is to label them with attributes. The seed lexicon provides attributes for a subset of these nodes. Nodes are connected to each other through edges that denote features shared between them or surface morphological transformation between them. Our model relies on the principle of compositionality for morphology according to which the morphemes in a word collectively assign meaning to a word. Thus, our model exploits how similar or different the

morphemes between a pair of words are to identify how the attribute distribution of the two words' should be related.

Our entire framework of lexicon generation, including the label propagation algorithm and the feature extraction module is language independent. We only use word-level morphological, semantic and syntactic relations between words that can be induced from unannotated corpora in an unsupervised manner. One particularly novel aspect of our graph-based framework is that edges are featurized. Some of these features measure similarity, e.g., singular nouns tend to occur in similar distributional contexts as other singular nouns, but some also measure transformations from one inflection to another, e.g., adding a "ed" suffix could indicate flipping the TENSE:PRES attribute to TENSE:PAST (in English). For every attribute to be propagated, we learn weights over features on the edges separately. This is in contrast to traditional label propagation, where edges indicate similarity exclusively (Zhu *et al.*, 2005). In spirit our model learns how transformations and shared features between words correspond to the variation of attributes between them, similar to how Eisner (2002) learn probabilities for PCFG rules in terms of transformations between them.

To evaluate the model, we construct lexicons in 11 languages of varying morphological complexity. We perform intrinsic evaluation of the quality of generated lexicons obtained from either the universal dependency treebank or created manually by humans (§5.4). We show that these automatically created lexicons provide useful features in two extrinsic NLP tasks which require identifying the contextually plausible morphological and syntactic roles: morphological tagging (Hajič and Hladká, 1998b; Hajič, 2000) and syntactic dependency parsing (Kübler *et al.*, 2009). We obtain an average of 15.4% and 5.3% error reduction across 11 languages for morphological tagging and dependency parsing respectively on a set of publicly available treebanks (§5.5). We anticipate that the lexicons thus created will be useful in a variety of NLP problems.

5.2 Graph Construction

The approach we take propagates information over lexical graphs (§5.3). In this section we describe how to construct the graph that serves as the backbone of our model. We construct a graph in which nodes are word types and directed edges are present between nodes that share *one or more* features. Edges between nodes denote that there might be a relationship between the attributes of the two nodes, which we intend to learn. As we want to keep our model language independent, we use edge features that can be induced between words without using any language specific tools. To this end, we describe three features in this section that can be obtained using unlabeled corpora for any given language.[2] Fig. 5.1 shows a subgraph

[2]Some of these features can cause the graph to become very dense making label propagation prohibitive. We keep the size of the graph in check by only allowing a word node to be connected to at most 100 other (randomly

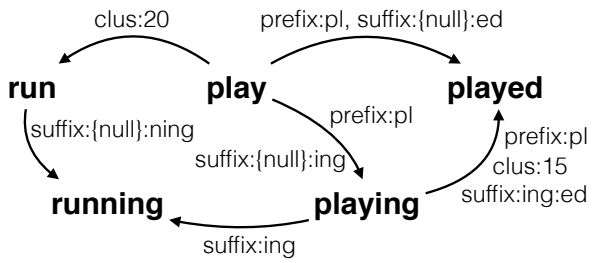

Figure 5.1: A subgraph from the complete graph of English showing different kinds of features shared on the edges between words. Some possible features/edges have been removed for enhancing clarity.

of the full graph constructed for English.

Word Clusters. Previous work has shown that unlabeled text can be used to induce unsupervised word clusters which can improve the performance of many NLP tasks in different languages (Clark, 2003; Koo *et al.*, 2008; Turian *et al.*, 2010; Faruqui and Padó, 2010; Täckström *et al.*, 2012a; Owoputi *et al.*, 2013). Word clusters capture semantic and syntactic similarities between words, for example, *play* and *run* are present in the same cluster. We obtain word clusters by using exchange clustering algorithm (Kneser and Ney, 1993b; Martin *et al.*, 1998; Uszkoreit and Brants, 2008) to cluster words present in a large unlabeled corpus. As in Täckström *et al.* (2012a), we use one year of news articles scrapped from a variety of sources and cluster only the most frequent 1M words into 256 different clusters. An edge was introduced for every word pair sharing the same word cluster and a feature for the cluster is fired. Thus, there are 256 possible cluster features on an edge, though in our case only a single one can fire.

Suffix & Prefix. Suffixes are often strong indicators of the morpho-syntactic attributes of a word (Ratnaparkhi, 1996; Clark, 2003). For example, in English, *-ing* denotes gerund verb forms like, *studying*, *playing* and *-ed* denotes past tense like *studied*, *played* etc. Prefixes like *un-*, *in-* often denote adjectives. Thus we include both 2-gram and 3-gram suffix and prefix as edge features.[3] We introduce an edge between two words sharing a particular suffix or prefix feature.

Morphological Transformations. Soricut and Och (2015) presented an unsupervised method of inducing prefix- and suffix-based morphological transformations between words using word embeddings. In their method, statistically, most of the transformations are induced between words with the same lemma (without using any prior information about the word

selected) word nodes sharing one particular feature. This reduces edges while still keeping the graph connected.

[3]We only include those suffixes and prefixes which appear at least twice in the seed lexicon.

lemma). For example, their method induces the transformation between *played* and *playing* as *suffix:ed:ing*. This feature indicates TENSE:PAST to turn off and TENSE:PRES to turn on.[4] We train the morphological transformation prediction tool of Soricut and Och (2015) on the news corpus (same as the one used for training word clusters) for each language. An edge is introduced between two words that exhibit a morphological transformation feature from one word to another as indicated by the tool's output.

Motivation for the Model. To motivate our model, consider the words *played* and *playing*. They have a common attribute POS:VERB but they differ in tense, showing TENSE:PAST and TENSE:PRES resp. Typical graph-propagation algorithms model similarity (Zhu *et al.*, 2005) and thus propagate all attributes along the edges. However, we want to model if an attribute should propagate or change across an edge. For example, having a shared cluster feature, is an indication of similar POS tag (Clark, 2003), but a surface morphological transformation feature like *suffix:ed:ing* possibly indicates a change in the tense of the word. Thus, we will model attributes propagation/transformation as a function of the features shared on the edges between words. The features described in this section are especially suitable for languages that exhibit concatenative morphology, like English, German, Greek etc. and might not work very well with languages that exhibit non-concatenative morphology i.e., where root modification is highly frequent like in Arabic and Hebrew. However, it is important to note that our framework is not limited to just the features described here, but can incorporate any arbitrary information over word pairs.

5.3 Graph-based Label Propagation

We now describe our model. Let $\mathcal{W} = \{w_1, w_2, \ldots, w_{|\mathcal{W}|}\}$ be the vocabulary with $|\mathcal{W}|$ words and $\mathcal{A} = \{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ be the set of lexical attributes that words in $\mathcal{W}$ can express; e.g. $\mathcal{W} = \{played, playing, \ldots\}$ and $\mathcal{A} = \{\text{NUM:SING}, \text{NUM:PLUR}, \text{TENSE:PAST}, \ldots\}$. Each word type $w \in \mathcal{W}$ is associated with a vector $\boldsymbol{a}_w \in [-1, 1]^{|\mathcal{A}|}$, where $a_{i,w} = 1$ indicates that word w has attribute i and $a_{i,w} = -1$ indicates its absence; values in between are treated as degrees of uncertainty. For example, $\text{TENSE:PAST}_{played} = 1$ and $\text{TENSE:PAST}_{playing} = -1$.[5]

The vocabulary $\mathcal{W}$ is divided into two disjoint subsets, the labeled words $\mathcal{L}$ for which we know their $\boldsymbol{a}_w$'s (obtained from seed lexicon)[6] and the unlabeled words $\mathcal{U}$ whose attributes are unknown. In general $|\mathcal{U}| \gg |\mathcal{L}|$. The words in $\mathcal{W}$ are organized into a directed graph with edges $\mathcal{E}$ between words. Let, vector $\boldsymbol{\phi}(w, v) \in [0, 1]^{|\mathcal{F}|}$ denote the features on the directed edge between words w and v, with 1 indicating the presence and 0 the absence of feature $f_k \in \mathcal{F}$,

[4] Our model will learn the following transformation: TENSE:PAST: 1 → -1, TENSE:PRESENT: -1 → 1 (§5.3).

[5] We constrain $a_{i,w} \in [-1, 1]$ as its easier to model the flipping of an attribute value from -1 to 1 as opposed to $[0, 1]$. This will be clearer soon.

[6] We use labeled, seed, and training lexicon to mean the same thing interchangeably.

where, $\mathcal{F} = \{f_1, f_2, \ldots, f_{|\mathcal{F}|}\}$ are the set of possible binary features shared between two words in the graph. For example, the features on edges between *playing* and *played* from Fig. 5.1 are:

$$\phi_k(playing, played) = \begin{cases} 1, & \text{if} f_k = \text{suffix:ing:ed} \\ 1, & \text{if} f_k = \text{prefix:pl} \\ 0, & \text{if} f_k = \text{suffix:ly} \\ \ldots & \end{cases}$$

Our model can be seen as an Ising model (Ising, 1925). Ising model is a mathematical model of ferromagnetism in statistical mechanics. The model consists of discrete variables that represent magnetic dipole moments of atomic spins that can be in one of two states (+1 or -1). The energy of an Ising model, which determines its stability is defined in terms of the interactions between the spins of the consecutive atoms. In our model, words are atoms, and the value of the attribute $a_{i,w}$ is the spin of the word. We posit that the attribute a_i of a word only interacts with the same attribute a_i of its neighbours in the model. Thus, our model is equivalent to $|\mathcal{A}|$ independent Ising models, every Ising model representing one single attribute. The energy of an Ising model i, with attribute distribution $\sigma_i = \langle a_{i,w} | w \in \mathcal{W} \rangle$ is given by:

$$H(\sigma_i) = -\sum_{w} \sum_{v \in \mathcal{N}(w)} J_{w,v} a_{i,w} a_{i,v} \tag{5.1}$$

where, $\mathcal{N}(w)$ are the neighbors of w in the graph, and $J_{w,v}$ is the strength of interaction or the edge weight between w and v. The probability of observing a particular attribute distribution σ_i on the graph then is:

$$P(\sigma_i) = \frac{e^{-H(\sigma_i)}}{Z} \tag{5.2}$$

where, Z is the normalization factor. Obtaining the σ_i which can maximize $P(\sigma_i)$ is an NP-complete problem (Cipra, 2000). However, the naïve mean field approximation, which focuses on one spin and assumes that the most important contribution to the interactions of such spin with neighbouring spins is determined by the mean field due to its neighboring spins, can approximate the behavior of Ising models in equilibrium. The naïve mean field approximation of the Ising model is given by $a_{i,w} = \tanh(\sum_{v \in \mathcal{N}(w)} J_{w,v} a_{i,v})$ (Opper and Winther, 2001; Wang *et al.*, 2007).

In our model, we define the strength of interaction or the edge weight between two nodes by $J_{w,v} = \boldsymbol{\phi}(w, v) \cdot \boldsymbol{\theta_i}$. Here, $\boldsymbol{\theta_i} \in \mathbb{R}^{|\mathcal{F}|}$ is weight vector of the edge features for estimating attribute a_i, and '$\cdot$' represents dot product betwen two vectors. The set of such weights $\boldsymbol{\theta} \in \mathbb{R}^{|\mathcal{A}| \times |\mathcal{F}|}$ for all attributes are the model parameters that we learn. Now we can use the

following iterative update, similar to the retrofitting rule:

$$\hat{a}_{i,w} = \tanh\left(\sum_{v \in \mathcal{N}(w)} (\boldsymbol{\phi}(w,v) \cdot \boldsymbol{\theta_i}) \times a_{i,v}\right) \tag{5.3}$$

where, $\hat{a}_{i,w}$ is the empirical estimate of the attribute. Intuitively, one can view the node to node message function from v to w: $\boldsymbol{\phi}(w,v) \cdot \boldsymbol{\theta_i} \times a_{i,v}$ as either:

1. Supporting the value $a_{i,v}$ when $\boldsymbol{\phi}(w,v) \cdot \boldsymbol{\theta_i} > 0$
2. Inverting $a_{i,v}$ when $\boldsymbol{\phi}(w,v) \cdot \boldsymbol{\theta_i} < 0$
3. Dampening or neutering $a_{i,v}$ when $\boldsymbol{\phi}(w,v) \cdot \boldsymbol{\theta_i} \approx 0$

Returning to our motivation, if $w = \textit{played}$ and $v = \textit{playing}$, a feature indicating the suffix substitution *suffix:ed:ing* should have a highly negative weight for TENSE:PAST, indicating a change in value. This is because TENSE:PAST = -1 for *playing*, and a negative value of $\boldsymbol{\phi}(w,v) \cdot \boldsymbol{\theta_i}$ will push it to positive for *played*.

It should be noted that this framework for constructing lexicons does not explicitly distinguish between morpho-syntactic paradigms, but simply identifies all possible attribute-values a word can take. If we consider an example like "games" and two attributes, the syntactic part-of-speech, POS, and number, NUM, games can either be {POS:VERB, NUM:SING}, as in *John games the system*; or {POS:NOUN, NUM:PLUR}, as in *The games have started*. Our framework will merely return that all the above attribute-values are possible, which implies that the singular noun and plural verb interpretations are valid.

Our framework has three critical components, each described below: (1) model estimation, i.e., learning $\boldsymbol{\theta}$; (2) label propagation to $\mathcal{U}$; and optionally (3) paradigm projection to known valid morphological paradigms. The overall procedure is illustrated in figure 5.2 and made concrete in algorithm 1.

5.3.1 Edge Feature Weights Estimation

We estimate all individual elements of an attribute vector using eq. 5.3. We define loss as the squared loss between the empirical and observed attribute vectors on every labeled node in the graph, thus the total loss can be computed as:

$$\sum_{w \in \mathcal{L}} \|\mathbf{a}_w - \hat{\mathbf{a}}_w\|_2^2 \tag{5.4}$$

We train the edge feature weights $\boldsymbol{\theta}$ by minimizing the loss function in eq. 5.4. In this step, we only use labeled nodes and the edge connections between labeled nodes. As such, this is strictly a supervised learning setup. We minimize the loss function using online adaptive

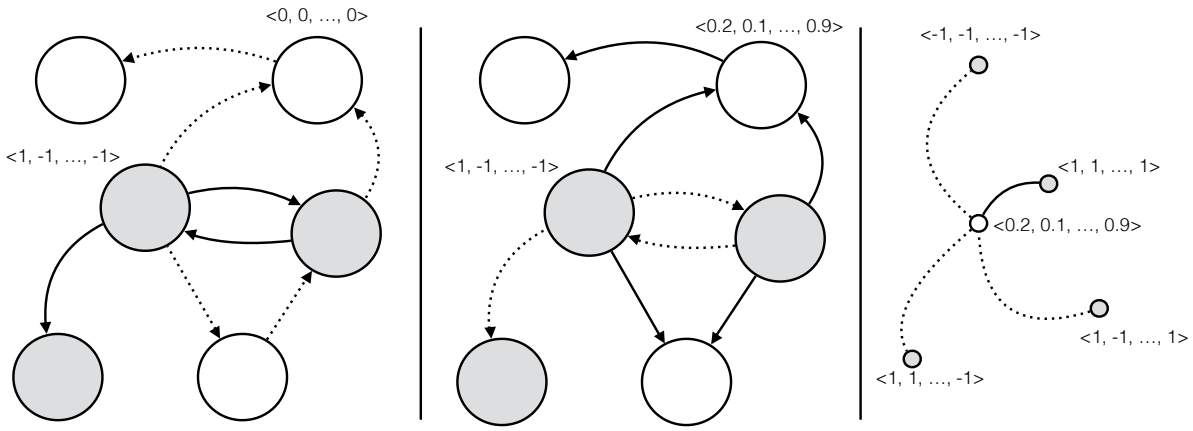

Figure 5.2: Word graph with edges between words showing the labeled (grey) and the unlabeled (white) word nodes. Only nodes connected via solid edges are visible to each other, dotted edges block visibility. This figure demonstrates interaction between nodes during model estimation (left), label propagation (center), and paradigm projection (right). Attribute-value vectors of the words are shown in angled brackets. The solid edge in the right figure shows the closest attribute paradigm to which the empirical vector is projected.

gradient descent (Duchi *et al.*, 2010) with ℓ_2 regularization on the feature weights $\boldsymbol{\theta}$. This is the first step in Algorithm 1 (lines 1–5).

5.3.2 Label Propagation

In the second step, we use the learned weights of the edge features to estimate the attribute values over unlabeled nodes iteratively. The attribute vector of all unlabeled words is initialized to null, $\forall w \in \mathcal{U}, \boldsymbol{a}_w = \langle 0, 0, \ldots, 0 \rangle$. In every iteration, an unlabeled node estimates its empirical attributes by looking at the corresponding attributes of its labeled and unlabeled neighbors using eq. 5.3, thus this is the semi-supervised step. We stop after the squared Euclidean distance between the attribute vectors at two consecutive iterations for a node becomes less than 0.1 (averaged over all unlabeled nodes). This is the second step in Algorithm 1 (lines 6–9). After convergence, we can directly obtain attributes for a word by thresholding: a word w is said to possess an attribute a_i if $a_{i,w} > 0$.

5.3.3 Paradigm Projection

Since a word can be labeled with multiple lexical attributes, this is a multi-label classification problem. For such a task, several advanced methods that take into account the correlation between attributes have been proposed (Ghamrawi and McCallum, 2005; Tsoumakas and Katakis, 2006; Fürnkranz *et al.*, 2008; Read *et al.*, 2011), here we have adopted the *binary relevance method* which trains a classifier for every attribute independently of the other attributes,

Data: $\mathcal{W}$, $\mathcal{L}$, $\mathcal{U}$, $\mathcal{A}$, $\mathcal{F}$, $\mathcal{P}$
Result: $\boldsymbol{\theta}$, labeled $\mathcal{U}$

```
1  model optimization
2  while convergence do
3      for w ∈ L do
4          loss ← ‖a_w − â_w‖_2^2
5          Update θ using ∂loss/∂θ
6  label propagation
7  while convergence do
8      for w ∈ U do
9          a_w ← â_w
10 paradigm projection
11 for w ∈ U do
12     mindist ← ∞, closest ← ∅
13     for p ∈ P do
14         dist ← ‖a_w − p‖_2^2
15         if dist < mindist then
16             mindist ← dist, closest ← p
17     a_w ← closest
```

Algorithm 1: Graph-based semi-supervised label propagation algorithm.

for its simplicity (Godbole and Sarawagi, 2004; Zhang and Zhou, 2005).

However, as the decision for the presence of an attribute over a word is independent of all the other attributes, the final set of attributes obtained for a word in §5.3.2 might not be a valid paradigm.[7] For example, a word cannot only exhibit the two attributes POS:Noun and Tense:Past, since the presence of the tense attribute implies POS:Verb should also be true. Further, we want to utilize the inherent correlations between attribute labels to obtain better solutions. We thus present an alternative, simpler method to account for this problem. To ensure that we obtain a valid attribute paradigm, we project the empirical attribute vector obtained after propagation to the space of all valid paradigms.

We first collect all observed and thus valid attribute paradigms from the seed lexicon ($\mathcal{P} = \{\boldsymbol{a}_w | w \in \mathcal{L}\}$). We replace the empirical attribute vector obtained in §5.3.2 by a valid attribute paradigm vector which is nearest to it according to Euclidean distance. This projection step is inspired from the decoding step in label-space transformation approaches to multilabel classification (Hsu *et al.*, 2009; Ferng and Lin, 2011; Zhang and Schneider, 2011). This is the last step in Algorithm 1 (lines 10–17). We investigate for each language if paradigm projection is helpful (§5.4.1).

[7] A paradigm is defined as a set of attributes.

	$\lvert\mathcal{L}\rvert$ (k)	$\lvert\mathcal{W}\rvert$ (k)	$\lvert\mathcal{E}\rvert$ (m)	$\lvert\mathcal{A}\rvert$	$\lvert\mathcal{P}\rvert$	Prop (k)
eu	3.4	130	13	83	811	118
bg	8.1	309	27	57	53	285
hr	6.9	253	26	51	862	251
cs	58.5	507	51	122	4,789	403
da	5.9	259	26	61	352	246
en	4.1	1,006	100	50	412	976
fi	14.4	372	30	100	2,025	251
el	3.9	358	26	40	409	236
hu	1.9	451	25	85	490	245
it	8.5	510	28	52	568	239
sv	4.8	305	26	41	265	268

Table 5.2: Graph statistics for different languages, showing the approximate number of labeled seed nodes ($|\mathcal{L}|$), labeled and unlabeled nodes ($|\mathcal{W}|$), edges between words ($|\mathcal{E}|$), the number of unique attributes ($|\mathcal{A}|$), attribute paradigms ($|\mathcal{P}|$) and size of the constructed lexicon (Prop). k: thousands, m: millions.

5.4 Intrinsic Evaluation

To ascertain how our graph-propagation framework predicts morphological attributes for words, we provide an intrinsic evaluation where we compare predicted attributes to gold lexicons that have been either read off from a treebank or annotated manually.

5.4.1 Dependency Treebank Lexicons

The universal dependency treebank (McDonald *et al.*, 2013; De Marneffe *et al.*, 2014; Agić *et al.*, 2015) contains dependency annotations for sentences and morpho-syntactic annotations for words in context for a number of languages. A word can display different attributes depending on its role in a sentence. In order to create morpho-syntactic lexicon for every language, we take the union of all the attributes that the word realizes in the entire treebank. It is possible that this lexicon might not contain all realizable attributes if a particular attribute or paradigm is not seen in the treebank (we address this issue in §5.4.2). The utility of evaluating against treebank derived lexicons is that it allows us to evaluate on a large set of languages. In particular, in the universal dependency treebanks v1.1 (Agić *et al.*, 2015), 11 diverse languages contain the morphology layer, including Romance, Germanic and Slavic languages plus isolates like Basque and Greek.

We use the train/dev/test set of the treebank to create training (seed),[8] development and test lexicons for each language. We exclude words from the dev and test lexicon that have been seen in seed lexicon. For every language, we create a graph with the features described

[8]We only include those words in the seed lexicon that occur at least twice in the training set of the treebank.

	Clus	Suffix	Prefix	MorphTrans	Proj
eu	✓			✓	✓
bg	✓			✓	
hr	✓	✓			✓
cs	✓	✓	✓	✓	✓
da	✓			✓	✓
en	✓			✓	✓
fi	✓			✓	
el	✓	✓		✓	✓
hu	✓			✓	✓
it	✓			✓	✓
sv	✓	✓			✓

Table 5.3: Features selected and the decision of paradigm projection (Proj) tuned on the development lexicon for each language. ✓denotes a selected feature.

in §5.2 with words in the seed lexicon as labeled nodes. The words from development and test set are included as unlabeled nodes for the propagation stage.[9] Table 5.2 shows statistics about the constructed graph for different languages.[10]

We perform feature selection and hyperparameter tuning by optimizing prediction on words in the development lexicon and then report results on the test lexicon. Our feature selection is done on the template level, i.e., if we decide to not use suffixes, the entire set of suffixes are removed. Thus the ablation experiments are done for word clusters, suffix, prefix, and morphological transformation feature templates. The decision whether paradigm projection (§5.3.3) is useful or not is also taken by tuning performance on the development lexicon. Table 5.3 shows the features that were selected for each language. Now, for every word in the test lexicon we obtain predicted lexical attributes from the graph. For a given attribute, we count the number of words for which it was correctly predicted (true positive), wrongly predicted (false positive) and not predicted (false negative). Aggregating these counts over all attributes ($\mathcal{A}$), we compute the micro-averaged F_1 score and achieve 74.3% on an average across 11 languages (cf. table 5.4). Note that this systematically underestimates performance due to the effect of missing attributes/paradigms that were not observed in the treebank.

Propagated Lexicons. The last column in Table 5.2 shows the number of words in the propagated lexicon, and the first column shows the number of words in the seed lexicon. The ratio of the size of propagated and seed lexicon is different across languages, which presumably depends on how densely connected each language's graph is. For example, for English the

[9]Words from the news corpus used for word clustering are also used as unlabeled nodes.

[10]Note that the size of the constructed lexicon (cf. Table 5.2) is always less than or equal to the total number of unlabeled nodes in the graph because some unlabeled nodes are not able to collect enough mass for acquiring an attribute i.e, $\forall a \in \mathcal{A} : a_w < 0$ and thus they remain unlabeled (cf. §5.3.2).

propagated lexicon is around 240 times larger than the seed lexicon, whereas for Czech, its 8 times larger. We can individually tune how densely connected graph we want for each language depending on the seed size and feature sparsity, which we leave for future work.

Selected Edge Features. The features most frequently selected across all the languages are the word cluster and the surface morphological transformation features. This essentially translates to having a graph that consists of small connected components of words having the same lemma (discovered in an unsupervised manner) with semantic links connecting such components using word cluster features. Suffix features are useful for highly inflected languages like Czech and Greek, while the prefix feature is only useful for Czech. Overall, the selected edge features for different languages correspond well to the morphological structure of these languages (Dryer, 2013).

Corpus Baseline. We compare our results to a corpus-based method of obtaining morpho-syntactic lexicons. We hypothesize that if we use a morphological tagger of reasonable quality to tag the entire wikipedia corpus of a language and take the union of all the attributes for a word type across all its occurrences in the corpus, then we can acquire all possible attributes for a given word. Hence, producing a lexicon of reasonable quality. Moore (2015) used this technique to obtain a high quality tag dictionary for POS-tagging. We thus train a morphological tagger (detail in §5.5.1) on the training portion of the dependency treebank and use it to tag the entire Wikipedia corpus. For every word, we add an attribute to the lexicon if it has been seen at least k times for the word in the corpus, where $k \in [2, 20]$. This threshold on the frequency of the word-attribute pair helps prevent noisy judgements. We tune k for each language on the development set and report results on the test set in Table 5.4. We call this method the Corpus baseline. It can be seen that for every language we outperform this baseline, which on average has an F_1 score of 67.1%.

5.4.2 Manually Curated Lexicons

We have now showed that its possible to automatically construct large lexicons from smaller seed lexicons. However, the seed lexicons used in §5.4.1 have been artifically constructed from aggregating attributes of word types over the treebank. Thus, it can be argued that these constructed lexicons might not be complete i.e., the lexicon might not exhibit all possible attributes for a given word. On the other hand, manually curated lexicons are unavailable for many languages, inhibiting proper evaluation.

To test the utility of our approach on manually curated lexicons, we investigate publicly available lexicons for Finnish (Pirinen, 2011), Czech (Hajič and Hladká, 1998a) and Hungarian (Trón *et al.*, 2006). We eliminate numbers and punctuation from all lexicons. For each of these

	words	Corpus	Propagation
eu	3409	54.0	57.5
bg	2453	66.4	73.6
hr	1054	69.7	71.6
cs	14532	79.1	80.8
da	1059	68.2	78.1
en	3142	57.2	72.0
fi	2481	58.2	68.2
el	1093	72.2	82.4
hu	1004	64.9	70.9
it	1244	78.8	81.7
sv	3134	69.8	80.7
avg.	3146	67.1	74.3

Table 5.4: Micro-averaged F_1 score (%) for prediction of lexical attributes on the test set using our propagation algorithm (Propagation) and the corpus-based baseline (Corpus). Also, shown are the no. of words in test set.

	words	F_1
cs	115,218	87.5
fi	39,856	71.9
hu	135,386	79.7
avg.	96,820	79.7

Table 5.5: Micro-averaged F_1 score (%) for prediction of lexical attributes on the test lexicon of human-curated lexicons.

languages, we select 10000 words for training and the rest of the word types for evaluation. We train models obtained in §5.4.1 for a given language using suffix, Brown and morphological transformation features with paradigm projection. The only difference is the source of the seed lexicon and test set. Results are reported in Table 5.5 averaged over 10 different randomly selected seed set for every language. For each language we obtain more than 70% F_1 score and on an average obtain 79.7%. The F_1 score on human curated lexicons is higher for each language than the treebank constructed lexicons, in some cases as high as 9% absolute. This suggests that the average 74.3% F_1 score across all 11 languages is likely underestimated.

5.5 Extrinsic Evaluation

We now show that the automatically generated lexicons provide informative features that are useful in two downstream NLP tasks: morphological tagging (§5.5.1) and syntactic dependency parsing (§5.5.2).

word	exchange cluster*
lowercase(word)	capitalization
{1,2,3}-g suffix*	digit
{1,2,3}-g prefix*	punctuation

Table 5.6: Features used to train the morphological tagger on the universal dependency treebank. * on for word offsets {-2, -1, 0, 1, 2}. Conjunctions of the above are also included.

	None	Seed	Propagation
eu	84.1	84.4	**85.2**
bg	94.2	94.6	**95.9**
hr	92.5	**93.6**	93.2
cs	96.8	97.1	97.1
da	96.4	97.1	**97.3**
en	94.4	94.7	**94.8**
fi	92.8	93.6	**94.0**
el	93.4	**94.6**	94.2
hu	91.7	92.3	**93.5**
it	96.8	97.1	97.1
sv	95.4	96.5	96.5
avg.	93.5	94.2	**94.5**

Table 5.7: Macro-averaged F_1 score (%) for morphological tagging: without using any lexicon (None), with seed lexicon (Seed), with propagated lexicon (Propagation).

5.5.1 Morphological Tagging

Morphological tagging is the task of assigning a morphological reading to a token in context. The morphological reading consists of features such as part of speech, case, gender, person, tense etc. (Oflazer and Kuruöz, 1994; Hajič and Hladká, 1998b). The model we use is a standard atomic sequence classifier that classifies the morphological bundle for each word independent of the others (with the exception of features derived from these words). Specifically, we use a linear SVM model classifier with hand tuned features. This is similar to commonly used analyzers like SVMTagger (Màrquez and Giménez, 2004) and MateTagger (Bohnet and Nivre, 2012).

Our taggers are trained in a language independent manner (Hajič, 2000; Smith *et al.*, 2005; Müller *et al.*, 2013). The list of features used in training the tagger are listed in Table 5.6. In addition to the standard features, we use the morpho-syntactic attributes present in the lexicon for every word as features in the tagger. As shown in Müller and Schuetze (2015), this is typically the most important feature for morphological tagging, even more useful than clusters or word embeddings. While predicting the contextual morphological tags for a given word, the morphological attributes present in the lexicon for the current word, the previous word and the next word are used as features.

	None	Seed	Propagation
eu	60.5	62.3	**62.9**
bg	78.3	78.8	**79.3**
hr	72.8	74.7	74.7
cs	78.3	78.4	78.4
da	67.5	69.4	**70.1**
en	74.4	74.1	**74.4**
fi	66.1	67.4	**67.9**
el	75.0	75.6	**75.8**
hu	67.6	69.0	**71.1**
it	82.4	82.8	**83.1**
sv	69.7	70.1	70.1
avg.	72.0	73.0	**73.5**

Table 5.8: Labeled accuracy score (LAS, %) for dependency parsing: without using any lexicon (None), with seed (Seed), with propagated lexicon (Propagation). Bold indicates best result for each language.

We use the same 11 languages from the universal dependency treebanks (Agić *et al.*, 2015) that contain morphological tags to train and evaluate the morphological taggers. We use the pre-specified train/dev/test splits that come with the data. Table 5.7 shows the macro-averaged F_1 score over all attributes for each language on the test lexicon. The three columns show the F_1 score of the tagger when no lexicon is used; when the seed lexicon derived from the training data is used; and when label propagation is applied.

Overall, using lexicons provides a significant improvement in accuracy, even when just using the seed lexicon. For 9 out of 11 languages, the highest accuracy is obtained using the lexicon derived from graph propagation. In some cases the gain is quite substantial, e.g., 94.6% $\rightarrow$ 95.9% for Bulgarian. Overall there is 1.0% and 0.3% absolute improvement over the baseline and seed respectively, which corresponds roughly to a 15% and 5% relative reduction in error. It is not surprising that the seed lexicon performs on par with the derived lexicon for some languages, as it is derived from the training corpus, which likely contains the most frequent words of the language.

5.5.2 Dependency Parsing

We train dependency parsers for the same 11 universal dependency treebanks that contain the morphological layer (Agić *et al.*, 2015). We again use the supplied train/dev/test split of the dependency treebank to develop the models. Our parsing model is the transition-based parsing system of Zhang and Nivre (2011) with identical features and a beam of size 8.

We augment the features of Zhang and Nivre (2011) in two ways: using the context-independent morphological attributes present in the different lexicons; and using the corre-

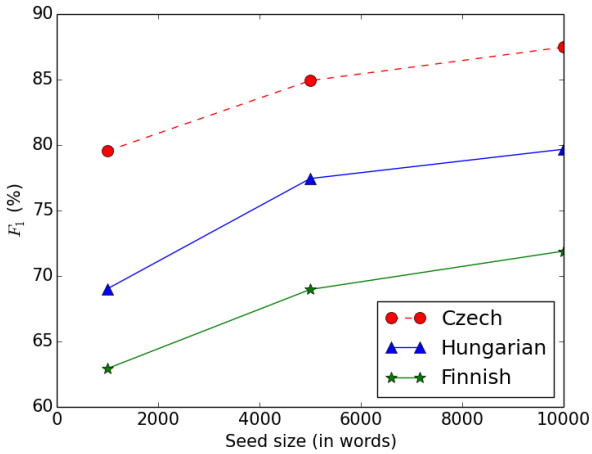

Figure 5.3: Micro-average F_1 score for predicting the morpho-syntactic attributes on the test lexicon while using varying seed sizes for cs, hu and fi.

sponding morphological taggers from §5.5.1 to generate context-dependent attributes. For each of the above two kinds of features, we fire the attributes for the word on top of the stack and the two words on at the front of the buffer. Additionally we take the cross product of these features between the word on the top of the stack and at the front of the buffer.

Table 5.8 shows the labeled accuracy score (LAS) for all languages. Overall, the generated lexicon gives an improvement of absolute 1.5% point over the baseline (5.3% relative reduction in error) and 0.5% over the seed lexicon on an average across 11 languages. Critically this improvement holds for 10/11 languages over the baseline and 8/11 languages over the system that uses seed lexicon only.

5.6 Further Analysis

In this section we further investigate our model and results in detail.

Size of seed lexicon. We first test how the size of the seed lexicon affects performance of attribute prediction on the test set (intrinsic evaluation for prediction of morpho-syntactic attributes). We use the manually constructed lexicons described in §5.4.2 for experiments. For each language, instead of using the full seed lexicon of 10000 words, we construct subsets of this lexicon by taking 1000 and 5000 randomly sampled words. We then train models obtained in §5.4.1 on these lexicons and plot the performance on the test set in Figure 5.3. On average across three languages, we observe that the absolute performance improvement from 1000 to 5000 seed words is ≈10% whereas it reduces to ≈2% from 5000 to 10000 words.

VFORM:GER		NUM:PLUR	
Clus:105 (ex.: playing, running)	+	Clus:19 (ex.: movies, tickets)	+
Clus:77 (ex.: eating, feeding)	+	Clus:97 (ex.: guardsmen, women)	+
suffix:ing:{null}	-	suffix:ies:y	-
suffix:ping:{null}	-	suffix:gs:g	-
suffix:ing:er	-	suffix:ons:on	-

Table 5.9: Highest (upper half) and lowest (lower half) weighted features (with their sign) for predicting a given attribute of English words.

	Word	Attributes
en	study (seed)	POS:Verb, VForm:Fin, Mood:Ind, Tense:Pres, Num:Sing, POS:Noun
	studied	POS:Verb, VForm:Fin, Mood:Ind, Tense:Past, VForm:Part
	taught	POS:Verb, VForm:Fin, Mood:Ind, Tense:Past, VForm:Part, Voice:Pass
it	tavola (seed)	POS:Noun, Gender:Fem, Num:Sing
	tavoli	POS:Noun, Gender:Masc, Num:Plur
	divano	POS:Noun, Gender:Masc, Num:Sing

Table 5.10: Attributes induced for words which are semantically or syntactically related to a word in the seed lexicon for English and Italian.

Feature analysis. Table 5.9 shows the highest and the lowest weighted features for predicting a given attribute of English words. The highest weighted features for both VFORM:GER and NUM:PLUR are word clusters, indicating that word clusters exhibit strong syntactic and semantic coherence. More interestingly, it can be seen that for predicting VFORM:GER i.e., continuous verb forms, the lowest weighted features are those morphological transformations that substitute "ing" with something else. Thus, if there exists an edge between the words *studying* and *study*, containing the feature suffix:ing:{null}, the model would correctly predict that *studying* is VFORM:GER as *study* is not so and the negative feature weight can flip the label values. The same observation holds true for NUM:PLUR.

Prediction examples. Table 5.10 shows examples of predictions made by our model for English and Italian. For each language, we first select a random word from the seed lexicon, then we pick one syntactic and one semantically related word to the selected word from the set of unlabeled words. For e.g., in Italian *tavola* means *table*, whereas *tavoli* is the plural form and *divano* means *sofa*. We correctly identify attributes for these words.

5.7 Conclusion

A word is composed of sub-word units called morphemes which compose together to assign meaning to the word. These morphemes thus provide informative signals about the word's meaning. In this chapter, we have shown that morphology of the word provides evidence

about its meaning and such evidence is useful in addition to the distributional information of the word. We have presented a graph-based semi-supervised method to construct large annotated morpho-syntactic lexicons from small seed lexicons. In addition to distributional evidence in the form of word clusters, we use the internal morphological structure of a word to predict its morpho-syntactic attributes. Our method is language independent, and can be used for expanding both human constructed and dependency treebank lexicons. We have constructed lexicons for 11 different languages, and showed that the lexicons thus constructed help improve performance in morphological tagging and dependency parsing, when used as features.

Chapter 6

Translational Context

Apart from the monolingual context of a word, the context of a word's translation across different languages has been shown to provide evidence of its meaning (Resnik and Yarowsky, 1999; Diab, 2003). The translational context of a word can be obtained from a parallel corpus. In a parallel corpus, every sentence in one language has a corresponding translated sentence in another language. Bannard and Callison-Burch (2005) show that if different words or phrases often translate into a single word or phrase type in a different language, this is good evidence that they are synonymous. For example, Figure 6.1 shows an English sentence which has two possible translations in Arabic. Although, the two verbs *ylEb* and *ylEbwn*[1] mean the same in English, they have different surface forms and different positions with respect to other words in the sentence. The cross-lingual pivoting here helps us identify that these two words are synonymous and should belong to the same word cluster. Firth might paraphrase this as *"You shall know a word by how it translates"*.

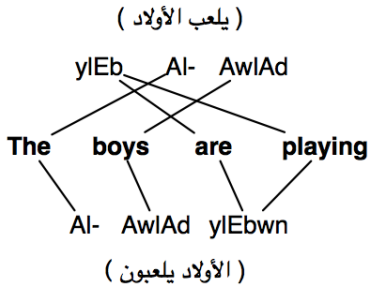

Figure 6.1: A 2 sentence Arabic–English parallel corpus with word alignments.

How can we exploit information like this when constructing word meaning representations? We present models to incorporate this translational information into word meaning representations that are constructed primarily using monolingual distributional information.

[1]These words are transcribed in Buckwalter encoding `http://languagelog.ldc.upenn.edu/myl/ldc/morph/buckwalter.html`

Specifically, we explore incorporating translational information into distributional word clusters (§6.1), and word vectors (§6.2).[2]

6.1 Word Clusters

A word cluster is a group of words that ideally captures syntactic, semantic, and distributional regularities among the words belonging to the group. Word clustering is widely used to reduce the number of parameters in statistical models which leads to improved generalization (Brown *et al.*, 1992; Kneser and Ney, 1993a; Clark, 2003; Koo *et al.*, 2008; Turian *et al.*, 2010), and multilingual clustering has been proposed as a means to improve modeling of translational correspondences and to facilitate projection of linguistic resource across languages (Och, 1999; Täckström *et al.*, 2012b). In this chapter, we show that generally more informative clusters can be learned when evidence from multiple languages is considered while creating the word clusters.

We propose a novel bilingual word clustering objective in which the first term deals with each language independently and ensures that the data is well-explained by the clustering in a sequence model. The second term ensures that the cluster alignments induced by a word alignment have high mutual information across languages. Since the objective consists of terms representing the entropy of monolingual data (for each language) and parallel bilingual data, it is particularly attractive for the usual situation in which there is much more monolingual data available than parallel data. Because of its similarity to the variation of information metric (Meilă, 2003), we call this bilingual term in the objective the **aligned variation of information**.

6.1.1 Monolingual Objective

A word clustering $\mathcal{C}$ is a partition of a vocabulary $\Sigma = \{x_1, x_2, \ldots, x_{|\Sigma|}\}$ into K disjoint subsets, $C_1, C_2, \ldots, C_K$. That is, $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$; $C_i \cap C_j = \emptyset$ for all $i \neq j$ and $\bigcup_{k=1}^{K} C_k = \Sigma$. We use the average surprisal in a probabilistic sequence model to define the monolingual clustering objective. Let c_i denote the word class of word w_i. Our objective assumes that the probability of a word sequence $\mathbf{w} = \langle w_1, w_2, \ldots, w_M \rangle$ is

$$p(\mathbf{w}) = \prod_{i=1}^{M} p(c_i \mid c_{i-1}) \times p(w_i \mid c_i),$$

where c_0 is a special start symbol. The term $p(c_i \mid c_{i-1})$ is the probability of class c_i following class c_{i-1}, and $p(w_i \mid c_i)$ is the probability of class c_i emitting word w_i. Using the MLE estimates after taking the negative logarithm, this term reduces to the following as shown in

[2]Previously published in Faruqui and Dyer (2013) and Faruqui and Dyer (2014b).

(Brown *et al.*, 1992):

$$H(\mathcal{C}; \mathbf{w}) = 2 \sum_{k=1}^{K} \frac{\#(C_k)}{M} \log \frac{\#(C_k)}{M} - \sum_i \sum_{j \neq i} \frac{\#(C_i, C_j)}{M} \log \frac{\#(C_i, C_j)}{M}$$

where $\#(C_k)$ is the count of C_k in the corpus $\mathbf{w}$ under the clustering $\mathcal{C}$, $\#(C_i, C_j)$ is the count of the number of times that cluster C_i precedes C_j and M is the size of the corpus. Using the monolingual objective to cluster, we solve the following search problem:

$$\hat{\mathcal{C}} = \arg\min_{\mathcal{C}} H(\mathcal{C}; \mathbf{w}). \tag{6.1}$$

This is a computationally hard combinatorial optimization problem, and finding the clustering that can maximize the average mutual information is impractical. However, starting from an initial estimate, a greedy hill-climbing word exchange algorithm can often produce good results (Brown *et al.*, 1992; Martin *et al.*, 1995).

6.1.2 Bilingual Objective

Now let us suppose we have a second language with vocabulary $\Omega = \{y_1, y_2, \ldots, y_{|\Omega|}\}$, which is clustered into K disjoint subsets $\mathcal{D} = \{D_1, D_2, \ldots, D_K\}$, and a corpus of text in the second language, $\mathbf{v} = \langle v_1, v_2, \ldots, v_N \rangle$. Obviously we can cluster both languages using the monolingual objective above:

$$\hat{\mathcal{C}}, \hat{\mathcal{D}} = \arg\min_{\mathcal{C}, \mathcal{D}} H(\mathcal{C}; \mathbf{w}) + H(\mathcal{D}; \mathbf{v}). \tag{6.2}$$

This joint minimization for the clusterings for both languages clearly has no benefit since the two terms of the objective are independent. We must alter the object by further assuming that we have *a priori* beliefs that some of the words in $\mathbf{w}$ and $\mathbf{v}$ have the same meaning.

To encode this belief, we introduce the notion of a weighted vocabulary alignment $\mathcal{A}$, which is a function on pairs of words in vocabularies Σ and Ω to a value greater than or equal to 0, i.e., $\mathcal{A} : \Sigma \times \Omega \mapsto \mathbb{R}_{\geq 0}$. For concreteness, $\mathcal{A}(x, y)$ will be the number of times that x is aligned to y in a word aligned parallel corpus. By abuse of notation, we write marginal weights $\mathcal{A}(x) = \sum_{y \in \Omega} \mathcal{A}(x, y)$ and $\mathcal{A}(y) = \sum_{x \in \Sigma} \mathcal{A}(x, y)$. We also define the set marginals $\mathcal{A}(C, D) = \sum_{x \in C} \sum_{y \in D} \mathcal{A}(x, y)$. Using this weighted vocabulary alignment, we state an objective that encourages clusterings to have high average mutual information when alignment links are followed; that is, on average how much information does knowing the cluster of a word $x \in \Sigma$ impart about the clustering of $y \in \Omega$, and vice-versa? We call this quantity the **aligned variation of information** (AVI).

$$\text{AVI}(\mathcal{C}, \mathcal{D}; \mathcal{A}) = \mathbb{E}_{\mathcal{A}(x,y)} \left[-\log p(c_x \mid d_y) - \log p(d_y \mid c_x) \right] \tag{6.3}$$

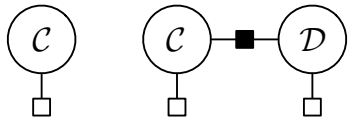

Figure 6.2: Factor graphs of the monolingual (left) & proposed bilingual clustering problem (right).

Writing out the expectation and gathering terms, we obtain

$$\mathrm{AVI}(\mathcal{C},\mathcal{D};\mathcal{A}) = -\sum_{x\in\Sigma}\sum_{y\in\Omega}\frac{\mathcal{A}(x,y)}{\mathcal{A}(\cdot,\cdot)}\times\left[2\log\frac{\mathcal{A}(C,D)}{\mathcal{A}(\cdot,\cdot)} - \log p(C) - \log p(D)\right],$$

where it is assumed that $0\log x = 0$. Our bilingual clustering objective can therefore be stated as the following search problem over a linear combination of the monolingual and bilingual objectives:

$$\arg\min_{\mathcal{C},\mathcal{D}} \overbrace{H(\mathcal{C};\mathbf{w}) + H(\mathcal{D};\mathbf{v})}^{\text{monolingual}} + \overbrace{\beta\mathrm{AVI}(\mathcal{C},\mathcal{D})}^{\beta\times\text{bilingual}}. \tag{6.4}$$

Figure 6.2 shows the factor graph representation of our clustering models.

Aligned Variation of Information

Intuitively, we can imagine sampling a random alignment from the distribution obtained by normalizing $\mathcal{A}(\cdot,\cdot)$. AVI gives us a measure of how much information do we obtain, on average, from knowing the cluster in one language about the clustering of a linked element chosen at random proportional to $\mathcal{A}(x,\cdot)$ (or conditioned the other way around). In the following sections, we denote $\mathrm{AVI}(\mathcal{C},\mathcal{D};\mathcal{A})$ by $\mathrm{AVI}(\mathcal{C},\mathcal{D})$. To further understand AVI, we remark that AVI reduces to the VI metric when the alignment maps words to themselves in the same language. As a proper metric, VI has a number of attractive properties, and these can be generalized to AVI (without restriction on the alignment map), namely:

- *Non-negativity:* $\mathrm{AVI}(C,D) \geq 0$
- *Symmetry:* $\mathrm{AVI}(C,D) = \mathrm{AVI}(D,C)$
- *Triangle inequality:* $\mathrm{AVI}(C,D) + \mathrm{AVI}(D,E) \geq \mathrm{AVI}(C,E)$
- *Identity of indiscernibles:* $\mathrm{AVI}(C,D) = 0$ iff $C \equiv D$.[3]

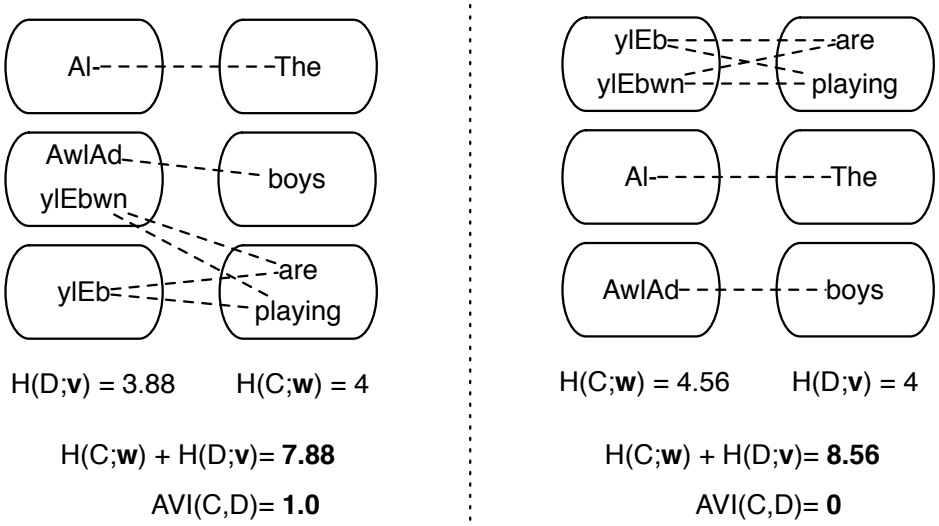

Figure 6.3: A 3-class clustering that maximizes the monolingual objective ($\beta = 0$; center); and a 3-class clustering that maximizes the joint monolingual and bilingual objective (any $\beta > 0.68$; right).

Example

Figure 6.3 provides an example illustrating the difference between the bilingual vs. monolingual clustering objectives. We compare two different clusterings of a two-sentence Arabic-English parallel corpus (the English half of the corpus contains the same sentence, twice, while the Arabic half has two variants with the same meaning). While English has a relatively rigid SVO word order, Arabic can alternate between the traditional VSO order and an more modern SVO order. Since our monolingual clustering objective relies exclusively on the distribution of clusters before and after each token, flexible word order alternations like this can cause unintuitive results. To further complicate matters, verbs can inflect differently depending on whether their subject precedes or follows them (Haywood and Nahmad, 1999), so a monolingual model, which knows nothing about morphology and may only rely on distributional clues, has little chance of performing well without help. This is indeed what we observe in the monolingual objective optimal solution (center), in which `AwlAd` (*boys*) and `yElbwn` (*play*+PRES + 3PL) are grouped into a single class, while `yElb` (*play*+PRES + 3SG) is in its own class. However, the AVI term (which is of course not included) has a value of 1.0, reflecting the relatively disordered clustering relative to the given alignment. On the right, we see the optimal solution that includes the AVI term in the clustering objective. This has an AVI of 0, indicating that knowing the clustering of any word is completely informative about the words it is aligned to. By including this term, a slightly worse monolingual solution is

[3] $C \equiv D$ iff $\forall i |\{\mathcal{D}(y) | \forall (x,y) \in \mathcal{A}, \mathcal{C}(x) = i\}| = 1$

chosen, but the clustering corresponds to the reasonable intuition that words with the same meaning (i.e., the two variants of *to play*) should be clustered together.

Inference

Finding the optimal clustering under both the monolingual and bilingual objectives is a computationally hard combinatorial optimization problem (Och, 1995). We use a greedy hill-climbing word exchange algorithm (Martin *et al.*, 1995) to find a minimum value for our objective. We terminate the optimization procedure when the number of words exchanged at the end of one complete iteration through both the languages is less than 0.1% of the sum of vocabulary of the two languages and at least five complete iterations have been completed. In practice, the number of exchanged words drops of exponentially, so this threshold is typically reached in not many iterations. For every language the word clusters are initialized in a round robin order according to the token frequency.

6.1.3 Evaluation

Evaluation of clustering is not a trivial problem. One branch of work seeks to recast the problem as the of part-of-speech (POS) induction and attempts to match linguistic intuitions. However, hard clusters are particularly useful for downstream tasks (Turian *et al.*, 2010). We therefore chose to focus our evaluation on the a downstream task. For our evaluation, we use our word clusters as an input to a named entity recognizer which uses these clusters as a source of features. Our evaluation task is the German corpus with NER annotation that was created for the shared task at CoNLL-2003.[4] The training set contains approximately 220,000 tokens and the development set and test set contains 55,000 tokens each. We use Stanford's Named Entity Recognition system[5] which uses a linear-chain conditional random field to predict the most likely sequence of NE labels (Finkel and Manning, 2009).

Corpora for Clustering. We used parallel corpora for {Arabic, English, French, Korean & Turkish}-German pairs from WIT-3 corpus (Cettolo *et al.*, 2012),[6] which is a collection of translated transcriptions of TED talks. Each language pair contained around 1.5 million German words. The corpus was word aligned in two directions using an unsupervised word aligner (Dyer *et al.*, 2013), then the intersected alignment points were taken.

Monolingual Clustering. For every language pair, we train German word clusters on the monolingual German data from the parallel data. Note that the parallel corpora are of different sizes and hence the monolingual German data from every parallel corpus is different. We

[4] `http://www.cnts.ua.ac.be/conll2003/ner/`

[5] `http://nlp.stanford.edu/ner/index.shtml`

[6] `https://wit3.fbk.eu/mt.php?release=2012-03`

Language Pair	Dev				Test	
	— (only bi)	$\beta = 0$ (only mono)	$\beta = 0.1$ (unrefined)	$\beta = 0.1$ (refined)	$\beta = 0$ (only mono)	$\beta = 0.1$ (refined)
No clusters	68.27				72.32	
English–German	68.95	70.04	**70.33**	**70.64**†	72.30	**72.98**†
French–German	69.16	69.74	69.69	**69.89**	72.66	**72.83**
Arabic–German	69.01	69.65	69.40	**69.67**	72.90	72.37
Turkish–German	69.29	69.46	**69.64**	**70.05**†	72.41	**72.54**
Korean–German	68.95	69.70	**69.78**	**69.95**	72.71	72.54
Average	69.07	69.71	**69.76**	**70.04**†	72.59	**72.65**

Table 6.1: NER performance using different word clustering models. Bold indicates an improvement over the monolingual ($\beta = 0$) baseline; † indicates a significant improvement (McNemar's test, $p < 0.01$).

treat the F_1 score obtained using monolingual word clusters ($\beta = 0$) as the baseline. Table 6.1 shows the F_1 score of NER when trained on these monolingual German word clusters.[7]

Bilingual Clustering. While we have formulated a joint objective that enables using both monolingual and bilingual evidence, it is possible to create word clusters using the bilingual signal only (AVI($\mathcal{C}, \mathcal{D}$)) by removing the first term in Eq.6.4. Table 6.1 shows the performance of NER when the word clusters are obtained using only the bilingual information for different language pairs. As can be seen, these clusters are helpful for all the language pairs. For Turkish the F_1 score improves by 1.0 point over when there are no distributional clusters which clearly shows that the word alignment information improves the clustering quality. We now need to supplement the bilingual information with monolingual information to see if the improvement sustains. The parameter β controls the strength of the bilingual objective and initial experiments showed that the value of $\beta = 0.1$ is fairly robust across other language pairs and hence we fix it to that for all the experiments. We run our bilingual clustering model across all language pairs and note the F_1 scores. Table 6.1 (unrefined) shows that except for Arabic–German & French–German, all other language pairs deliver a better F_1 score than only using monolingual German data. In case of Arabic–German there is a drop in score by 0.25 points. Although we have observed improvement in F_1 score over the monolingual case, the gains do not reach significance according to McNemar's test (Dietterich, 1998).

Thus we propose to further refine the quality of word alignment links as follows: Let x be a word in language Σ and y be a word in language Ω and let there exists an alignment link between x and y. Recall that $\mathcal{A}(x, y)$ is the count of the alignment links between x and y observed in the parallel data, and $\mathcal{A}(x)$ and $\mathcal{A}(y)$ are the respective marginal counts. Then we define an edge association weight $e(x, y) = \frac{2 \times \mathcal{A}(x,y)}{\mathcal{A}(x)+\mathcal{A}(y)}$ This quantity is an association of

[7] Faruqui and Padó (2010) show that for the size of our generalization data in German-NER, $K = 100$ should give us the optimum value.

the strength of the relationship between x and y, and we use it to remove all alignment links whose $e(x, y)$ is below a given threshold before running the bilingual clustering model. We vary e from 0.1 to 0.7 and observe the new F_1 scores on the development data. Table 6.1 (refined) shows the results obtained by our refined model. The values shown in bold are the highest improvements over the monolingual model.

Evaluation on Test Set. We now verify our results on the test set (Table 6.1). We take the best bilingual word clustering model obtained for every language pair on the dev set ($e = 0.1$ for English, French. $e = 0.5$ for Arabic, Turkish. $e = 0.7$ for Korean) and train NER classifiers using these. English again has a statistically significant improvement over the baseline. French and Turkish show the next best improvements. Overall, these results show how cross-lingual word alignments can help improve the quality of distributional word clusters as evaluated on an extrinsic task.

We presented a novel information theoretic model for bilingual word clustering which seeks a clustering with high average mutual information between clusters of adjacent words, and also high mutual information across observed word alignment links. We have shown that improvement in clustering can be obtained across a range of language pairs, evaluated in terms of their value as features in an extrinsic NER task. Our model can be extended for clustering any number of given languages together in a joint framework, and incorporate both monolingual and parallel data.

6.2 Word Vectors

In the previous section we saw how cross-lingual word alignments, or translational context of words can be useful in obtaining better quality distributional word clusters. In this section, we go one step further and argue for incorporating translational context when constructing word vector representations. Simply put: knowing how words translate is a valuable source of lexico-semantic information and should lead to better vector-space word representations. How can we exploit information like this when constructing vector-space models? We propose a technique that first constructs independent vector-space models in two languages and then projects them onto a common vector space such that translation pairs (as determined by automatic word alignments) are maximally correlated.

6.2.1 Bilingual Correlation with CCA

To gain information from the translation of a given word in other languages the most basic thing to do would be to just append the given word representation with the word representations of its translation in the other language. This has three drawbacks: first, it increases the

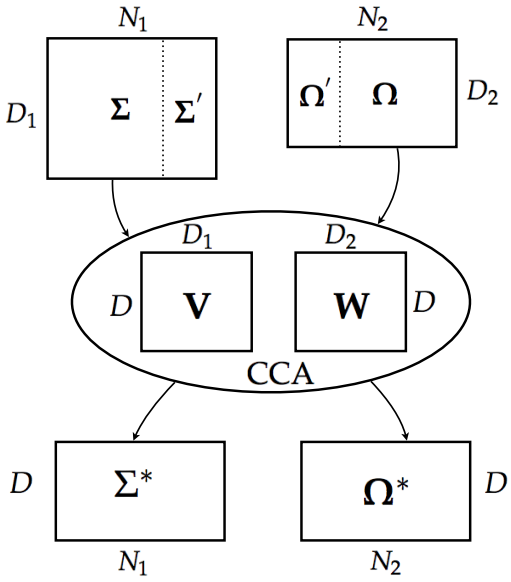

Figure 6.4: Cross-lingual word vector projection using canonical coorelation analysis (CCA).

number of dimensions in the vector; second, it can pull irrelevant information from the other language that doesn't generalize across languages and finally the given word might be out of vocabulary of the parallel corpus or dictionary. To counter these problems we use canonical correlation analysis, which is a way of measuring the linear relationship between two multidimensional variables. CCA finds two projection vectors, one for each variable, that are optimal with respect to correlations in the projected space. The dimensionality of these new projected vectors is equal to or less than the smaller dimensionality of the two variables.

Let $\mathbf{\Sigma} \in \mathbb{R}^{N_1 \times D_1}$ and $\mathbf{\Omega} \in \mathbb{R}^{N_2 \times D_2}$ be vector space embeddings of two different vocabularies where rows represent words. Since the two vocabularies are of different sizes (N_1 and N_2) and there might not exist translation for every word of $\mathbf{\Sigma}$ in $\mathbf{\Omega}$, let $\mathbf{\Sigma}^{'} \subseteq \mathbf{\Sigma}$ where every word in $\mathbf{\Sigma}^{'}$ is translated to one other word in $\mathbf{\Omega}^{'} \subseteq \mathbf{\Omega}$ and $\mathbf{\Sigma} \in \mathbb{R}^{D_1 \times N}$ and $\mathbf{\Omega} \in \mathbb{R}^{D_2 \times N}$. Let $\mathbf{x}$ and $\mathbf{y}$ be two corresponding vectors from $\mathbf{\Sigma}^{'}$ and $\mathbf{\Omega}^{'}$, and $\mathbf{V}$ and $\mathbf{W}$ be two projection matrices. Then, the projected vectors are: $\mathbf{x}^{'} = \mathbf{V}\mathbf{x}$ & $\mathbf{y}^{'} = \mathbf{V}\mathbf{y}$. and the correlation between the projected vectors can be written as: $\rho(\mathbf{x}^{'}, \mathbf{y}^{'}) = \frac{E[\mathbf{x}^{'}\mathbf{y}^{'}]}{\sqrt{E[\mathbf{x}^{'2}]E[\mathbf{y}^{'2}]}}$. CCA maximizes ρ for the given set of vectors $\mathbf{\Sigma}^{'}$ and $\mathbf{\Omega}^{'}$ and outputs two projection matrices $\mathbf{V}$ and $\mathbf{W}$:

$$\mathbf{V}, \mathbf{W} = \text{CCA}(\mathbf{x}, \mathbf{y}) = \arg\max_{\mathbf{V},\mathbf{W}} \rho(\mathbf{V}\mathbf{x}, \mathbf{W}\mathbf{y})$$

Using these two projection matrices we can project the entire vocabulary of the two languages

	Dim	Lang	WS-353	MEN	SimLex	SEM-REL	SYN-REL	Senti
SVD	80	Mono	34.8	50.4	18.6	13.5	24.4	77.2
	64	Multi	55.2	**67.1**	29.5	23.4	33.2	**77.4**
RNN	80	Mono	23.6	23.8	26.2	4.7	18.2	68.9
	64	Multi	27.9	27.4	21.6	4.1	12.2	68.2
SG	80	Mono	**63.9**	64.6	31.3	**47.8**	**47.8**	75.4
	64	Multi	63.1	67.0	**32.5**	46.5	44.2	76.6

Table 6.2: Spearman's correlation (left 3 columns) and accuracy (right 3 columns) on different tasks. Bold indicates best result across all vector types. Mono: monolingual and Multi: multilingual.

$\mathbf{\Sigma}$ and $\mathbf{\Omega}$. Summarizing:

$$\mathbf{V}, \mathbf{W} = \text{CCA}(\mathbf{\Sigma}^{'}, \mathbf{\Omega}^{'})$$

$$\Sigma^{*} = \mathbf{V\Sigma} \quad \mathbf{\Omega}^{*} = \mathbf{W\Omega} \tag{6.5}$$

where, $\mathbf{V} \in \mathbb{R}^{D \times D_1}$, $\mathbf{W} \in \mathbb{R}^{D \times D_2}$ contain the projection vectors and $D = \min\{\text{rank}(\mathbf{V}), \text{rank}(\mathbf{W})\}$. Thus, the resulting vectors cannot be longer than the original vectors. Since $\mathbf{V}$ and $\mathbf{W}$ can be used to project the whole vocabulary, CCA also solves the problem of not having translations of a particular word in the dictionary. The schema of performing CCA on the monolingual word representations of two languages is shown in Figure 6.4.

Further Dimensionality Reduction: Since CCA gives us correlations and corresponding projection vectors across D dimensions which can be large, we perform experiments by taking projections of the original word vectors across only the top K correlated dimensions. This is trivial to implement as the projection vectors $\mathbf{V}$, $\mathbf{W}$ in Equation 6.2.1 are already sorted in descending order of correlation. Therefore in,

$$\mathbf{\Sigma}^{*}_{K} = \mathbf{V}_K\mathbf{\Sigma} \quad \mathbf{\Omega}^{*}_{K} = \mathbf{W}_K\mathbf{\Omega} \tag{6.6}$$

$\mathbf{\Sigma}^{*}_{K}$ and $\mathbf{\Omega}^{*}_{K}$ are now word vector projections along the top K correlated dimensions, where, $\mathbf{V}_k$ and $\mathbf{W}_k$ are the column truncated matrices.

6.2.2 Evaluation

We evaluate the quality of our word vector representations on different word similarity, word analogy and sentiment analysis. We used three different word vector models: SVD, RNN and Skip-Gram (SG) described in appendix A.

Vector Training Data. For English, German and Spanish we used the WMT-2011[8] monolingual news corpora and for French we combined the WMT-2011 and 2012[9] monolingual news corpora so that we have around 300 million tokens for each language to train the word vectors.

For CCA, a one-to-one correspondence between the two sets of vectors is required. Obviously, the vocabulary of two languages are of different sizes and hence to obtain one-to-one mapping, for every English word we choose a word from the other language to which it has been aligned the maximum number of times[10] in a parallel corpus. We got these word alignment counts using fast-align (Dyer *et al.*, 2010) from the parallel news commentary corpora (WMT 2006-10) combined with the Europarl corpus for English–{German, French, Spanish}.

Evaluation Methodology. We construct word vectors of length 80 for English, German, French and Spanish. We project the English word vectors using CCA by pairing them with German, French and Spanish vectors. For every language pair we take the top k correlated dimensions (cf. equation 6.6), where $k \in \{10\%, 20\%, \ldots, 100\%\}$ and tune the performance on WS-353 task (cf. chapter 3). We then select the k that gives us the best average performance across language pairs, which is $k = 80\%$, and evaluate the corresponding vectors on all other benchmarks. This prevents us from over-fitting k for every individual task.

Results. We compare the best results obtained by using different types of monolingual word representations across all language pairs. We train word vectors of length 80 because it was computationally intractable to train the neural embeddings (RNN) for higher dimensions. For multilingual vectors, we obtain $k = 80\%$. Table 6.2 shows the correlation ratio and the accuracies for the respective evaluation tasks. For the SVD vectors the performance improves upon inclusion of multilingual context for all tasks, and the vectors give the best performance on MEN and Senti tasks. For both RNN and SG vectors we see a drop in performance on the word analogy tasks. Although both SVD and SG vectors improve in performance on the sentiment analysis task, RNN vectors' performance goes down, suggesting that semantic projection in the same space is probably not appropriate for syntactic RNN vectors. Overall, including multilingual information helps improve performance of vector space models on different tasks, even more so when the tasks are semantic in nature. Note that we do not evaluate word vectors using Qvec (§3.3) as the resultant bilingual word vectors are no longer the same dimensionality as the monolingual word vectors, and in the current version of Qvec, vectors with only the same dimensionality can be compared.

[8] `http://www.statmt.org/wmt11/`

[9] `http://www.statmt.org/wmt12/`

[10] We also tried weighted average of vectors across all aligned words and did not observe any significant difference in results.

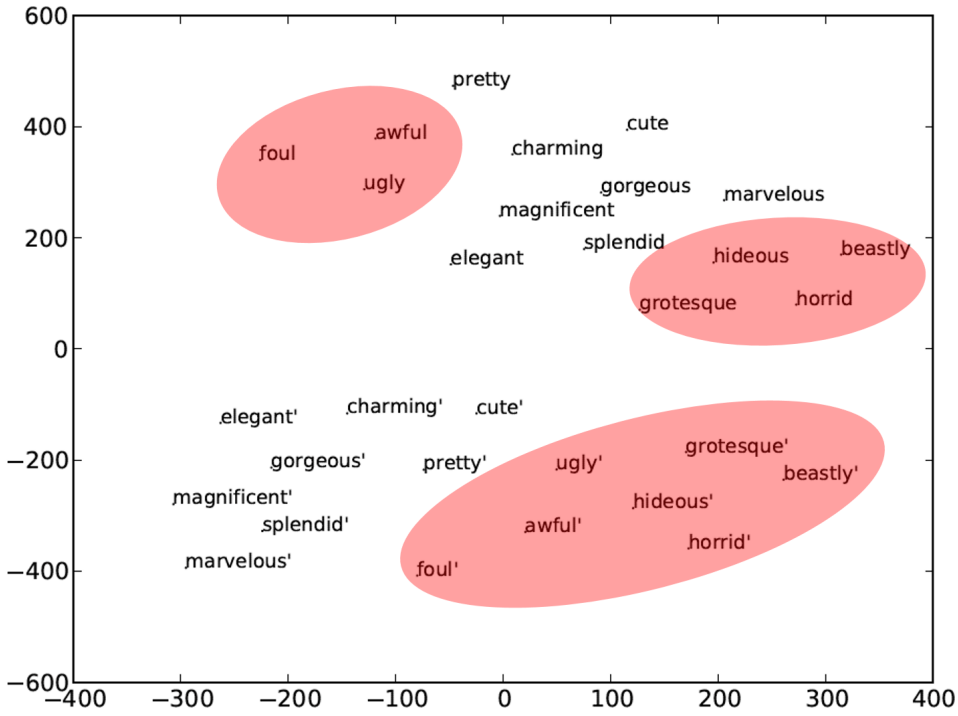

Figure 6.5: Monolingual (top) and multilingual (bottom; marked with apostrophe) word projections of the antonyms (shown in red) and synonyms of "beautiful".

6.2.3 Analysis

To understand how multilingual evidence leads to better results in semantic evaluation tasks, we plot the word representations of several synonyms and antonyms of the word "beautiful" by projecting both the original and improved vectors onto $\mathbb{R}^2$ using the t-SNE tool (van der Maaten and Hinton, 2008). The untransformed LSA vectors are in the upper part of Fig. 6.5, and the CCA-projected vectors are in the lower part. By comparing the two regions, we see that in the untransformed representations, the antonyms are in two clusters separated by the synonyms, whereas in the transformed representation, both the antonyms and synonyms are in their own cluster.

In order to demonstrate that the gains in performance by using multilingual correlation sustains for different number of dimensions, we compared the performance of the monolingual and (German-English) multilingual vectors. It can be see in figure 6.6 that the performance improvement for multilingual vectors remains almost the same for different vector lengths strengthening the reliability of our approach.

To further understand why multilingual context is highly effective for SVD vectors and to a large extent for RNN vectors as well, we plot (Figure 6.7) the correlation ratio obtained by varying the length of word representations by using equation 6.6 for the three different

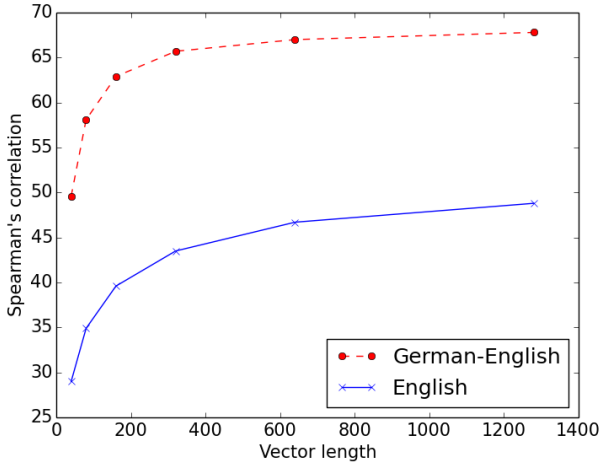

Figure 6.6: Performance of monolingual and multilingual vectors on WS-353 for different vector lengths.

vector types on WS-353 word similarity task. SVD vectors improve performance upon the increase of the number of dimensions and tend to saturate towards the end. For all the three language pairs the SVD vectors show uniform pattern of performance which gives us the liberty to use any language pair at hand. This is not true for the RNN vectors whose curves are significantly different for every language pair. SG vectors show a uniform pattern across different language pairs and the performance with multilingual context converges to the monolingual performance when the vector length becomes equal to the monolingual case ($k = 80$). The fact that both SG and SVD vectors have similar behavior across language pairs can be treated as evidence that semantics or information at a conceptual level (since both of them basically model word cooccurrence counts) transfers well across languages (Dyvik, 2004) although syntax has been projected across languages as well (Hwa *et al.*, 2005; Yarowsky and Ngai, 2001). The pattern of results in the case of RNN vectors are indicative of the fact that these vectors encode syntactic information which might not generalize well as compared to semantic information.

6.3 Conclusion

We have presented two novel models to incorporate translational context information in word cluster and word vectors. The bilingual word clustering model is an information theoretic model which seeks a clustering with high average mutual information between clusters of adjacent words, and also high mutual information across observed word alignment links. The bilingual word vector model uses canonical correlation analysis to project word vectors from two different languages into a shared space where they are maximally correlated. We have

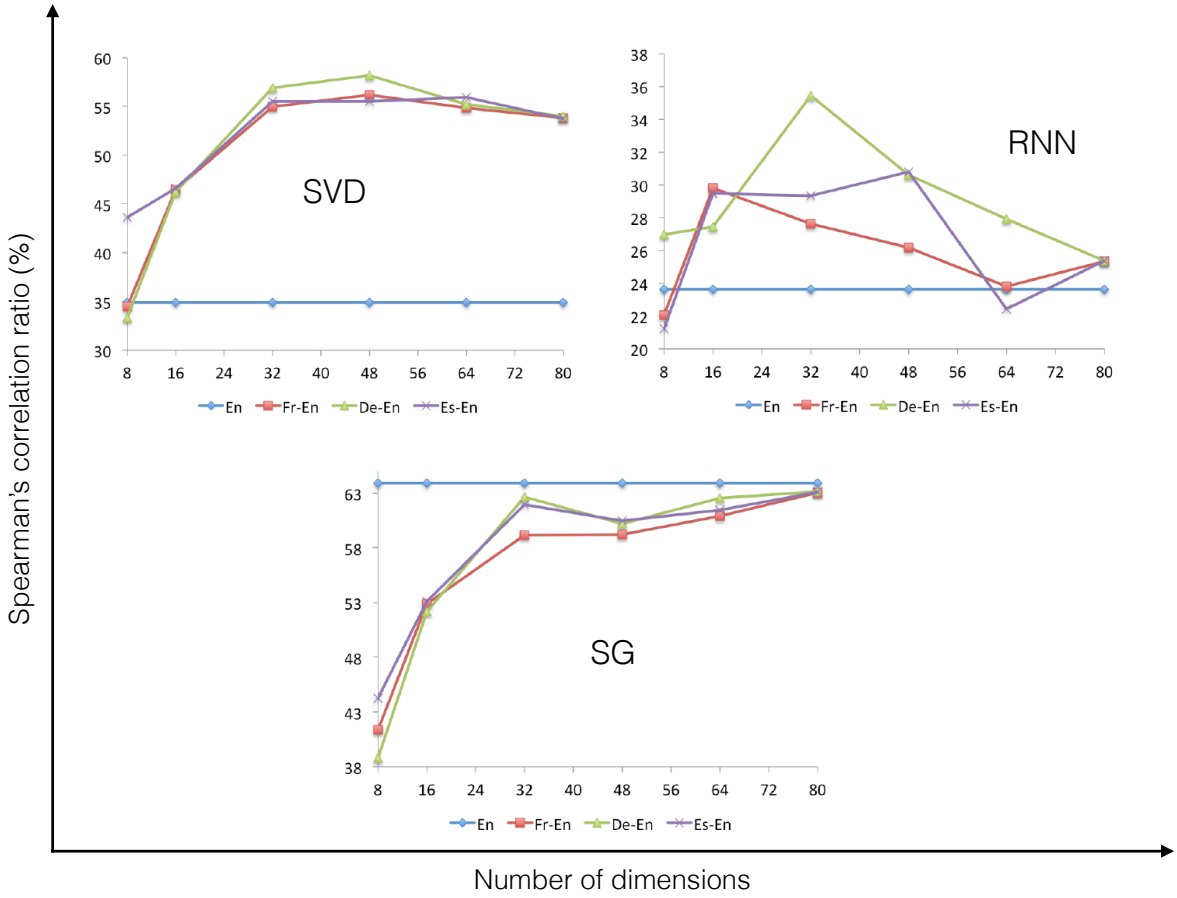

Figure 6.7: Performance as a function of vector length on WS-353. The monolingual vectors always have a fixed length of 80, they are just shown in the plots for comparison.

shown that word meaning representations can be improved using translation context of a word across languages, thus extending the distributional hypothesis. Firth might paraphrase this as *"You shall know a word by how it translates"*.

Chapter 7

Conclusion

7.1 Summary of Contributions

The distributional hypothesis states that the meaning of words is evidenced by the words it co-occurs with in natural language text. Current methods of obtaining word meaning representations like latent semantic analysis, Brown clustering etc. use information present in the distributional context of a word to obtain word representations in the form of word clusters or word vectors. Although distributional context is an important source of information for understanding word meaning, in this thesis, I show that other sources like word lexicons, provide complementary information about word meaning. I presented methods that can construct and improve models of word meaning representation by including information beyond the distributional context of the word. Contributions of this work include the following:

- I presented a method for intrinsic evaluation of the quality of word vector representations (chapter 3), which computes how well the dimensions of distributional word vectors quantify the linguistic content of words. An intrinsic evaluation method for word vectors will help in faster prototyping and comparison of different models of word vector representation.

- I showed that information contained in linguistic word lexicons can be used to construct word vector representations which are competitive to their distributional counterparts. Then, I present a model that can use such information extracted from semantic lexicons to improve the quality of distributional word vectors (chapter 4).

- I showed that the morphology of a word provides evidence about its meaning and present a model (chapter 5) for generating large morpho-syntactic word lexicons from small manually constructed seed lexicons using information extracted from the distributional context (in the form of word clusters), and the morphological structure of the word.

- I presented models (chapter 6) that can capture a word's context across languages using automatically extracted word alignments, to improve the quality of word cluster and word vector representations trained using monolingual distributional context.

7.2 Future Directions

The work presented in this thesis outlines three axes along which word representations are augmented. There are several directions that deserve further pursuit.

7.2.1 Richer knowledge

In chapter 3, we are currently using only supersenses of words as features representative of the linguistic content of words in QVEC. Supersenses only provide semantic information about the word. The linguistic matrix can be expanded to include various other sources of information like syntactic treebanks (ex. the Penn Treebank), FrameNet, emotion and connotation lexicons etc. which we used to construct the non-distributional word vectors (§4.1). As the size of the linguistic matrix increases, the runtime of QVEC will increase, and an interesting research problem is to find the optimal set of linguistic features that should be used in evaluation.

In chapter 4, we use retrofitting to make vector representation of words which have synonyms meanings lie closer to each other in the vector space. Retrofitting was only used for words related by synonymy, hyponymy, and hypernymy relation, where all of these relations were treated similarly by the model. Since these relations convey different information they should be treated differently. This can be done by tuning the edge weights for each relation for a given particular task. Retrofitting can also be modified to be used with words with antonymous meaning. In this direction, Mrkšić *et al.* (2016) have presented a method called *counter-fitting* to include words with antonymous meaning in the graph.

In chapter 5, we construct a graph of words, over which we propagate morpho-syntactic information to construct large morpho-syntactic lexicons from seed lexicons. The edges in the graph are labeled with features that are shared between the words. We use easily obtainable features like word clusters, {2, 3}-gram character prefix and suffix and morphological transformation between words. Instead of using raw {2, 3}-gram character suffixes and prefixes, we can use the correct suffix and prefix of the word by using a morphological segmentation tool, like Morfessor (Creutz and Lagus, 2005). We can also explore using the sum, difference, elementwise product, inner product or dot product of the word vectors of the two words connected by an edge as set of features shared between the words.

7.2.2 Richer representation models

Many words in the language have more than one sense of meaning, generally called word senses. For example, the word *bank* can refer to both a financial institution or the land alongside a river or lake, depending on the context it is used in. We have exclusively focused on word representations that can capture only one word sense at a time (or an average notion of word meaning across all senses). However, word vector models can be augmented to capture multiple senses (Reisinger and Mooney, 2010; Neelakantan *et al.*, 2014). Jauhar *et al.* (2015) present an extension of retrofitting that is capable of capturing multiple word senses.

The word representations explored in this thesis strictly capture meaning of an individual lexical unit, and are not inherently compositional. For example, adding the vectors of *New* and *York* may not produce a vector which represents the city *New York*. The methods presented in this thesis augment existing method of training word representations, and can be extended to multi-word expressions (MWE), by treating each MWE as an individual word (Baroni *et al.*, 2012). For example, if we know the vectors *New York*, *Chicago* and from a lexicon we know that both of them are cities, we can apply retrofitting to bring them closer together. There has been advances in training word representations that can compose together to form representations of the composed unit (Kiela and Clark, 2013; Vecchi *et al.*, 2013; Paperno *et al.*, 2014).

7.2.3 Bilingual to multilingual

In chapter 6, we show that the quality of word clusters and word vector representations can be improved by using information extracted from the translational context of a word in another language. We have currently only experimented with only using a pair of languages at a time, however, our models are easily extensible to support using translational context of a word across multiple languages together. For word clustering, this would imply computing the aligned variation of information between the clustering schemes of all pairs of languages, instead of just two. Specifically, in our word clustering model, if the monolingual clustering of one language is represented by $\mathcal{X}_i$ for $i = 1 \cdots N$, with corresponding vocabularies being $\mathbf{v}_i$, given N distinct languages, our new optimization objective is a generalization of Eq. 6.4: $\arg\min_{\forall i, \mathcal{X}_i} \sum_i H(\mathcal{X}_i; \mathbf{v}_i) + \beta \sum_{i \neq j} \text{AVI}(\mathcal{X}_i, \mathcal{X}_j)$.

For enriching word vectors with multilingual information, generalized canonical correlation analysis (CCA) can be used (Vıa *et al.*, 2005; Luo *et al.*, 2015). Generalized CCA projects the original data from different views into a common space so that the multiple correlation coefficient (a generalization of the standard Pearson correlation coefficient) between the random variables (views) is maximized. In other words, we compute the best low-rank approximation to the higher-order covariance tensor between all views, instead of the second-order covariance matrix between just two views (Ammar *et al.*, 2016).

7.2.4 Linear to non-linear models

In chapter 6, we use CCA to project the word vectors from two languages into a space where they are maximally correlated. CCA is a linear function. Lu *et al.* (2015) extended our method to use deep CCA, which first feeds the word vectors in a deep neural network and then applies CCA on the transformed vectors to obtain improved results. It is claimed that introducing non-linearities in the structure of the model helps learn better relations in the vector space of the two languages. On these lines, it would be interesting to see how kernel CCA (Lai and Fyfe, 2000), which also introduces non-linearity performs in comparison to CCA and deep CCA.

7.2.5 Multi-class classification

In chapter 5, we treat every morpho-syntactic attribute to be independent of all other attributes. This, is suboptimal as we know that morpho-syntactic attributes are correlated. For example, if a word is a verb, then it must also possess a tense. Instead of using the binary relevance method to train a binary classifier for every attribute independently, we can instead model such correlations as constraints during the learning step to obtain better solutions (Ghamrawi and McCallum, 2005; Tsoumakas and Katakis, 2006; Read *et al.*, 2011).

We can also treat the morpho-syntactic bundle of every word that it displays in a given context as a classification unit. For example, the word *games* can display two morpho-syntactic bundles {POS:Verb, NUM:Sing} or {POS:Noun, NUM:Plur}. We can train the classifier to predict both of these as individual units instead of trying to predict {POS:Verb, NUM:Sing, POS:Noun, NUM:Plur}. These bundles can also be looked upon as different senses that a word displays in the language.

7.2.6 Task-specific vs. task-independent representations

In this thesis we have exclusively discussed task independent word representations. These word representations were obtained using various resources that can provide information about word meaning, and evaluated on a number of tasks. The training of word vectors was done independently of the evaluation task. Training word representations this way is often sufficient for use in tasks, but it has been shown that tailoring them for a specific task at hand gives improved results (Socher *et al.*, 2013; Bansal *et al.*, 2014; Ling *et al.*, 2015b; Dyer *et al.*, 2015). Although task-specific representations provide better results, obtaining them for every task individually is cumbersome. An easy way to train task-specific word vectors is to initialize them using the methods described in this thesis, and further tune them as part of the training objective of the end task. Many of the results presented in this thesis, for example sentiment analysis, can be improved with task-specific training. The decision between task-

specific and task-independent representations is a good philosophical question, an exploration of which I leave to future work.

Appendices

Appendix A

Word Vector Models

Recurrent Neural Network Word Vectors. The recurrent neural network (RNN) language model maximizes the log-likelihood of the training corpus for the task of language modeling using RNNs. The architecture consists of an input layer, a hidden layer with recurrent connections to itself, an output layer and the corresponding weight matrices (Mikolov *et al.*, 2013b). The input vector $\mathbf{w}_t$ represents input word at time t encoded using 1-of-N encoding and the output layer $\mathbf{y}_t$ produces a probability distribution over words in the vocabulary V. The hidden layer maintains a representation of the sentence history in $\mathbf{s}_t$. The values in the hidden and output layer are computed as follows:

$$\mathbf{s}_t = f(\mathbf{U}\mathbf{w}_t + \mathbf{W}\mathbf{s}_{t-1})$$

$$\mathbf{y}_t = g(\mathbf{V}\mathbf{s}_t)$$

where, f and g are the logistic and softmax functions respectively. $\mathbf{U}$ and $\mathbf{V}$ are weight matrices and the word representations are found in the columns of $\mathbf{U}$. The model is trained using back-propagation. Since this model implicitly encodes the information about the position of words in the sentence, it induces word vectors with syntactic properties. We use the RNNLM toolkit[1] to induce these word representations.

Skip Gram. In the RNN model, most of the complexity is caused by the non-linear hidden layer. This is avoided in the new model proposed in Mikolov *et al.* (2013a) where they remove the non-linear hidden layer and there is a single projection layer for the input word. Precisely, each current word is used as an input to a log-linear classifier with continuous projection layer and words within a certain range before and after the word are predicted. These vectors are called the skip-gram vectors. If a context word c occurs with the target word w, then the

[1] `http://www.fit.vutbr.cz/~imikolov/rnnlm/`

optimization maximizes the following objective:

$$l = \sum_{w \in V_w} \sum_{c \in V_c} count(w, c) \log \sigma(\mathbf{w} \cdot \mathbf{c}) + \sum_{i=1, n \sim q}^{k} \log \sigma(\mathbf{w} \cdot \mathbf{n})$$

where, V_w and V_c are the vocabulary of target and context words respectively. $\sigma(\cdot)$ is the logistic function, k is the number of noise words drawn from the noise distribution q with n being the noise word. $count(w, c)$ counts the co-occurrence of the two words w and c. Google has released vectors trained using this model on 100 billion words of Google news dataset of length 300, which we often use in experiments throughout this thesis. A variant of this model in which the word is predicted given the contextual words is called the continuous bag of words model (CBOW). These vector models have been collectively released in the word2vec toolkit.[2]

CWindow and Structured Skip-Gram (SSG). Ling *et al.* (2015b) propose a syntactic modification to the WORD2VEC models that accounts for word order information, obtaining state-of-the-art performance in syntactic downstream tasks.[3]

CBOW with Attention (Attention). Ling *et al.* (2015a) further improve the WORD2VEC CBOW model by employing an attention model which finds, within the contextual words, the words that are more relevant for each prediction. In other words, it learns to identify and weigh probabilistically the importance of each contextual word. These vectors have been shown to benefit both semantically and syntactically oriented tasks.

Glove. Global vectors for word representations (Pennington *et al.*, 2014) are trained on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations are shown to display interesting linear substructures of the word vector space. These vectors were trained on 6 billion words from Wikipedia and English Gigaword and are of length 300.[4]

Global Context. These vectors are learned using a recursive neural network that incorporates both local and global (document-level) context features (Huang *et al.*, 2012). These vectors were trained on 1 billion words of English Wikipedia and are of length 50.[5]

[2] `https://code.google.com/p/word2vec`
[3] `https://github.com/wlin12/wang2vec`
[4] `http://www-nlp.stanford.edu/projects/glove/`
[5] `http://nlp.stanford.edu/~socherr/ACL2012_wordVectorsTextFile.zip`

Multilingual. Faruqui and Dyer (2014b) learned vectors by first performing SVD on text in different languages, then applying canonical correlation analysis (CCA) on pairs of vectors for words that align in parallel corpora. The monolingual vectors were trained on WMT-2011 news corpus for English, French, German and Spanish. We use the English word vectors projected in the common English–German space. The monolingual English WMT corpus had 360 million words and the trained vectors are of length 512.[6]

Latent Semantic Analysis (LSA). We construct word-word co-occurrence matrix $\mathbf{X}$; every element in the matrix is the pointwise mutual information between the two words (Church and Hanks, 1990). Then, truncated singular value decomposition is applied to factorize $\mathbf{X}$, where we keep the k largest singular values. Low dimensional word vectors of dimension k are obtained from $\mathbf{U_k}$ where $\mathbf{X} \approx \mathbf{U_k}\Sigma{\mathbf{V_k}}^\mathsf{T}$ (Landauer and Dumais, 1997).

GloVe+WN, GloVe+PPDB, LSA+WN, LSA+PPDB. We use retrofitting (Faruqui *et al.*, 2015) as a post-processing step to enrich GloVe and LSA vectors with semantic information from WordNet and Paraphrase database (PPDB) (Ganitkevitch *et al.*, 2013).[7]

[6] `http://www.wordvectors.org/web-eacl14-vectors/de-projected-en-512.txt.gz`
[7] `https://github.com/mfaruqui/retrofitting`

Appendix B

Additional Benchmarks for Qvec

Apart from the extrinsic evaluation dataset described in §3.2, we use additional text classification datasets for computing correlation between performance of word vectors on extrinsic tasks and QVEC.

20 Newsgroup Dataset. We consider four binary categorization tasks from the 20 Newsgroups dataset.[1] Each task involves categorizing a document according to two related categories with training/dev/test split in accordance with Yogatama and Smith (2014): (1) Sport: baseball vs. hockey (958/239/796) (2) Comp: ibm vs. mac (929/239/777) (3) Relig: atheism vs. christian (870/209/717) and (4) Science: medical vs. space. We train an ℓ_2-regularized logistic regression classifier on average of the word vectors of a given sentence as features. Classifier is tuned on the dev set and accuracy is reported on the test set.

Metaphor Prediction. A metaphor is a type of conceptual mapping, where words or phrases are applied to objects and actions in ways that do not permit a literal interpretation. For example, *my car drinks gasoline* is a metaphor, as a car can not drink gasoline literally. We use a state-of-the-art metaphor prediction system which takes word vectors as inputs for training and prediction (Tsvetkov *et al.*, 2014b).[2] The system uses word vectors as features in a random forest classifier to label adjective-noun pairs as literal/metaphoric. We report the system accuracy in 5-fold cross validation.

[1] `http://qwone.com/~jason/20Newsgroups`
[2] `https://github.com/ytsvetko/metaphor`

Bibliography

Agić, Ž., Aranzabe, M. J., Atutxa, A., Bosco, C., Choi, J., de Marneffe, M.-C., Dozat, T., Farkas, R., Foster, J., Ginter, F., Goenaga, I., Gojenola, K., Goldberg, Y., Hajič, J., Johannsen, A. T., Kanerva, J., Kuokkala, J., Laippala, V., Lenci, A., Lindén, K., Ljubešić, N., Lynn, T., Manning, C., Martínez, H. A., McDonald, R., Missilä, A., Montemagni, S., Nivre, J., Nurmi, H., Osenova, P., Petrov, S., Piitulainen, J., Plank, B., Prokopidis, P., Pyysalo, S., Seeker, W., Seraji, M., Silveira, N., Simi, M., Simov, K., Smith, A., Tsarfaty, R., Vincze, V., and Zeman, D. (2015). Universal dependencies 1.1. LINDAT/CLARIN digital library at Institute of Formal and Applied Linguistics, Charles University in Prague.

Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Paşca, M., and Soroa, A. (2009). A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proc. of NAACL*.

Ammar, W., Mulcaire, G., Tsvetkov, Y., Lample, G., Dyer, C., and Smith, N. A. (2016). Massively multilingual word embeddings. *CoRR*, **abs/1602.01925**.

Arisoy, E., Saraçlar, M., Roark, B., and Shafran, I. (2010). Syntactic and sub-lexical features for turkish discriminative language models. In *Proc. of ICASSP*.

Bach, F. R. and Jordan, M. I. (2005). A probabilistic interpretation of canonical correlation analysis. *UC Berkeley*.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The berkeley framenet project. In *Proc. of ACL*.

Baker, L. D. and McCallum, A. K. (1998). Distributional clustering of words for text classification. In *Proc. of SIGIR*.

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proc. of ACL*.

Bansal, M., Gimpel, K., and Livescu, K. (2014). Tailoring continuous word representations for dependency parsing. In *Proc. of ACL*.

Baroni, M. and Lenci, A. (2011). How we blessed distributional semantic evaluation. In *Proc. of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '11, pages 1–10.

Baroni, M., Bernardi, R., Do, N.-Q., and Shan, C.-c. (2012). Entailment above the word level in distributional semantics. In *Proc. of EACL*.

Bauer, L. (2003). *Introducing linguistic morphology*. Edinburgh University Press Edinburgh.

Bekkerman, R., El-Yaniv, R., Tishby, N., and Winter, Y. (2003). Distributional word clusters vs. words for text categorization. *The Journal of Machine Learning Research*, **3**, 1183–1208.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *JMLR*.

Bengio, Y., Delalleau, O., and Le Roux, N. (2006). Label propagation and quadratic criterion. In *Semi-Supervised Learning*. MIT Press.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **35**(8), 1798–1828.

Bian, J., Gao, B., and Liu, T.-Y. (2014). Knowledge-powered deep learning for word embedding. In *Proc. of MLKDD*.

Bohnet, B. and Nivre, J. (2012). A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proc. of EMNLP*.

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Comput. Linguist.*

Bruni, E., Boleda, G., Baroni, M., and Tran, N.-K. (2012). Distributional semantics in technicolor. In *Proc. of ACL*.

Budanitsky, A. and Hirst, G. (2006). Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, **32**(1), 13–47.

Bullinaria, J. A. and Levy, J. P. (2007). Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, **39**(3), 510–526.

Carpenter, B. (2008). Lazy sparse stochastic gradient descent for regularized multinomial logistic regression.

Cettolo, M., Girardi, C., and Federico, M. (2012). Wit^3: Web inventory of transcribed and translated talks. In *Proc. of EAMT*, Trento, Italy.

Chang, K.-W., Yih, W.-t., and Meek, C. (2013). Multi-relational latent semantic analysis. In *Proc. of EMNLP*.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Comput. Linguist.*, **16**(1), 22–29.

Ciaramita, M. and Altun, Y. (2006). Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proc. of EMNLP*.

Cipra, B. A. (2000). The ising model is np-complete. *SIAM News*, **33**(6), 1–3.

Clark, A. (2003). Combining distributional and morphological information for part of speech induction. In *Proc. of EACL*.

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. of EMNLP*.

Creutz, M. and Lagus, K. (2005). *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0*. Helsinki University of Technology.

Das, D. and Petrov, S. (2011). Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proc. of ACL*.

Das, D. and Smith, N. A. (2011). Semi-supervised frame-semantic parsing for unknown predicates. In *Proc. of ACL*.

De Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., and Manning, C. D. (2014). Universal stanford dependencies: A cross-linguistic typology. In *Proceedings of LREC*, pages 4585–4592.

de Melo, G. and Weikum, G. (2009). Towards a universal wordnet by learning from combined evidence. In *Proc. of CIKM*.

De Saussure, F. (1989). *Cours de linguistique générale: Édition critique*, volume 1. Otto Harrassowitz Verlag.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Senior, A., Tucker, P., Yang, K., Le, Q. V., *et al.* (2012). Large scale distributed deep networks. In *Proc. of NIPS*.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*.

Denis, P., Sagot, B., *et al.* (2009). Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art pos tagging with less human effort. In *Proc. of PACLIC*.

Dhillon, I. S., Mallela, S., and Kumar, R. (2002). Enhanced word clustering for hierarchical text classification. In *Proc. of KDD*.

Diab, M. T. (2003). *Word sense disambiguation within a multilingual framework*. Ph.D. thesis, University of Maryland at College Park, College Park, MD, USA. AAI3115805.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*.

Dryer, M. S. (2013). *Prefixing vs. Suffixing in Inflectional Morphology*. Max Planck Institute for Evolutionary Anthropology.

Duchi, J., Hazan, E., and Singer, Y. (2010). Adaptive subgradient methods for online learning and stochastic optimization. Technical Report UCB/EECS-2010-24, UC Berkeley.

Dukes, K. and Habash, N. (2010). Morphological annotation of quranic arabic. In *Proc. of LREC*.

Dyer, C. (2014). Notes on noise contrastive estimation and negative sampling. *arXiv preprint arXiv:1410.8251*.

Dyer, C., Lopez, A., Ganitkevitch, J., Weese, J., Setiawan, H., Ture, F., Eidelman, V., Blunsom, P., and Resnik, P. (2010). cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *In Proc. of ACL System Demonstrations*.

Dyer, C., Chahuneau, V., and Smith, N. A. (2013). A simple, fast, and effective reparameterization of IBM Model 2. In *Proc. of NAACL*.

Dyer, C., Ballesteros, M., Ling, W., Matthews, A., and Smith, N. A. (2015). Transition-based dependency parsing with stack long short-term memory. In *Proc. of ACL*.

Dyvik, H. (2004). Translations as semantic mirrors: from parallel corpus to wordnet. *Language and Computers*, **49**(1), 311–326.

Eisner, J. (2002). Transformational priors over grammars. In *Proc. of ACL*.

Faruqui, M. and Dyer, C. (2013). An information theoretic approach to bilingual word clustering. In *Proc. of ACL*.

Faruqui, M. and Dyer, C. (2014a). Community evaluation and exchange of word vectors at wordvectors.org. In *Proceedings of ACL: System Demonstrations*.

Faruqui, M. and Dyer, C. (2014b). Improving vector space word representations using multilingual correlation. In *Proc. of EACL*.

Faruqui, M. and Dyer, C. (2015). Non-distributional word vector representations. In *Proc. ACL*.

Faruqui, M. and Padó, S. (2010). Training and evaluating a german named entity recognizer with semantic generalization. In *Proc. of KONVENS*.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., and Smith, N. A. (2015). Retrofitting word vectors to semantic lexicons. In *Proc. of NAACL*.

Faruqui, M., McDonald, R., and Soricut, R. (2016). Morpho-syntactic lexicon generation using graph-based semi-supervised learning. *Transactions of the Association for Computational Linguistics*, **4**, 1–16.

Fellbaum, C., editor (1998). *WordNet: an electronic lexical database*. MIT Press.

Feng, S., Kang, J. S., Kuznetsova, P., and Choi, Y. (2013). Connotation lexicon: A dash of sentiment beneath the surface meaning. In *Proc. of ACL*.

Ferng, C.-S. and Lin, H.-T. (2011). Multi-label classification with error-correcting codes. In *Proc. of ACML*.

Fillmore, C., Bach, E., and Harms, R. (1968). *The Case for Case*. Eric Document Service.

Fillmore, C., Johnson, C., and Petruck, M. (2003). Lexicographic relevance: selecting information from corpus evidence. *International Journal of Lexicography*.

Finkel, J. R. and Manning, C. D. (2009). Nested named entity recognition. In *Proc. of EMNLP*.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2001). Placing search in context: the concept revisited. In *Proc. of WWW*.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2002). Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, **20**(1).

Firth, J. (1957). A synopsis of linguistic theory 1930-1955. *Studies in linguistic analysis*.

Frege, G. (1884). Die grundlageneine der arithmetik logisch-mathematische untersuchung über den begriff der zahl. *Wittgenstein Studien 3.2*.

Fürnkranz, J., Hüllermeier, E., Mencía, E. L., and Brinker, K. (2008). Multilabel classification via calibrated label ranking. *Machine learning*, **73**(2), 133–153.

Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). PPDB: The paraphrase database. In *Proc. of NAACL*.

Gardner, M., Huang, K., Papalexakis, E., Fu, X., Talukdar, P., Faloutsos, C., Sidiropoulos, N., and Mitchell, T. (2016). Translation invariant word embeddings. In *Proc. of EMNLP*.

Ghamrawi, N. and McCallum, A. (2005). Collective multi-label classification. In *Proc. of CIKM*.

Godbole, S. and Sarawagi, S. (2004). Discriminative methods for multi-labeled classification. In *Proc. of KDD*.

Goldberg, Y., Tsarfaty, R., Adler, M., and Elhadad, M. (2009). Enhancing unlexicalized parsing performance using a wide coverage lexicon, fuzzy tag-set mapping, and em-hmm-based lexical probabilities. In *Proc. of EACL*.

Golub, G. H. and Van Loan, C. F. (1996). *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. (2015). Qualitatively characterizing neural network optimization problems. In *Proc. of ICLR*.

Goodman, J. (2001). Classes for fast maximum entropy training. In *Proc. of ICASSP*.

Goyal, K. and Hovy, E. (2014). Unsupervised word sense induction using distributional statistics. In *Proc. of COLING*.

Green, S. and DeNero, J. (2012). A class-based agreement model for generating accurately inflected translations. In *Proc. of ACL*.

Guo, J., Che, W., Wang, H., and Liu, T. (2014). Revisiting embedding features for simple semi-supervised learning. In *Proc. of EMNLP*.

Gurevych, I. (2005). Using the structure of a conceptual network in computing semantic relatedness. In *Proc. of IJCNLP*.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proc. of AISTATS*.

Hajič, J. (2000). Morphological tagging: Data vs. dictionaries. In *Proc. of NAACL*.

Hajič, J. and Hladká, B. (1998a). Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. In *Proc. of Coling*.

Hajič, J. and Hladká, B. (1998b). Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. In *Proc. of Coling*.

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

Hassan, S. and Mihalcea, R. (2009). Cross-lingual semantic relatedness using encyclopedic knowledge. In *Proc. of EMNLP*.

Haywood, J. A. and Nahmad, H. M. (1999). *A new Arabic grammar of the written language*. Lund Humphries Publishers.

Heaps, H. S. (1978). *Information retrieval: Computational and theoretical aspects*. Academic Press, Inc.

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

Hinton, G. E. (1984). Distributed representations. *Carnegie Mellon University Technical Report*.

Hsu, D., Kakade, S., Langford, J., and Zhang, T. (2009). Multi-label prediction via compressed sensing. In *Proc. of NIPS*.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proc. of ACL*.

Hwa, R., Resnik, P., Weinberg, A., Cabezas, C., and Kolak, O. (2005). Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, **11**, 11–311.

Ising, E. (1925). Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik A Hadrons and Nuclei*, **31**(1), 253–258.

Jauhar, S. K., Dyer, C., and Hovy, E. (2015). Ontologically grounded multi-sense representation learning for semantic vector space models. In *Proc. NAACL*.

Joubarne, C. and Inkpen, D. (2011). Comparison of semantic similarity for different languages using the google n-gram corpus and second- order co-occurrence measures. In *Proc. of CAAI*.

Kemeny, J. G. (1959). Mathematics without numbers. *j-DAEDALUS*, **88**(4), 577–591.

Kiela, D. and Clark, S. (2013). Detecting compositionality of multi-word expressions using nearest neighbours in vector space models. In *Proc. of EMNLP*.

Kindermann, R. and Snell, J. L. (1980). *Markov Random Fields and Their Applications*. AMS.

Kneser, R. and Ney, H. (1993a). Forming word classes by statistical clustering for statistical language modelling. In *Contributions to Quantitative Linguistics*. Springer.

Kneser, R. and Ney, H. (1993b). Improved clustering techniques for class-based statistical language modelling. In *Proc. of Eurospeech*.

Kokkinakis, D., Toporowska-Gronostaj, M., and Warmenius, K. (2000). Annotating, disambiguating & automatically extending the coverage of the swedish simple lexicon. In *Proc. of LREC*.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proc. of ACL*.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, **8**, 30–37.

Kübler, S., McDonald, R., and Nivre, J. (2009). *Dependency parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Lai, P. L. and Fyfe, C. (2000). Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems*, **10**(05), 365–377.

Landauer, T. K. and Dumais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*.

Lazaridou, A., Vecchi, E. M., and Baroni, M. (2013). Fish transporters and miracle homes: How compositional distributional semantics can help NP parsing. In *Proc. of EMNLP*.

Lemaire, B. and Denhiere, G. (2008). Effects of high-order co-occurrences on word semantic similarities. *arXiv preprint arXiv:0804.0143*.

Levin, B. (1993). *English verb classes and alternations : a preliminary investigation*. University of Chicago Press.

Levy, O. and Goldberg, Y. (2014a). Dependency-based word embeddings. In *Proc. of ACL*.

Levy, O. and Goldberg, Y. (2014b). Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185.

Lewis, M. and Steedman, M. (2013). Combined distributional and logical semantics. *TACL – Volume 1*, pages 179–192.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proc. of COLING*.

Ling, W., Chu-Cheng, L., Tsvetkov, Y., Amir, S., Fermandez, R., Dyer, C., Black, A. W., and Trancoso, I. (2015a). Not all contexts are created equal: Better word representations with variable attention. In *Proc. of EMNLP*.

Ling, W., Dyer, C., Black, A., and Trancoso, I. (2015b). Two/too simple adaptations of `word2vec` for syntax problems. In *Proc. of NAACL*.

Lu, A., Wang, W., Bansal, M., Gimpel, K., and Livescu, K. (2015). Deep multilingual correlation for improved word embeddings. In *NAACL*.

Luo, Y., Tao, D., Wen, Y., Ramamohanarao, K., and Xu, C. (2015). Tensor Canonical Correlation Analysis for Multi-view Dimension Reduction. *ArXiv e-prints*.

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

Màrquez, L. and Giménez, J. (2004). A general pos tagger generator based on support vector machines. *Journal of Machine Learning Research*.

Martin, S., Liermann, J., and Ney, H. (1995). Algorithms for bigram and trigram word clustering. In *Speech Communication*, pages 1253–1256.

Martin, S., Liermann, J., and Ney, H. (1998). Algorithms for bigram and trigram word clustering. *Speech communication*, **24**(1), 19–37.

McDonald, R. T. and Pereira, F. C. (2006). Online learning of approximate dependency parsing algorithms. In *Proc. of EACL*.

McDonald, R. T., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K. B., Petrov, S., Zhang, H., Täckström, O., *et al.* (2013). Universal dependency annotation for multilingual parsing. In *Proc. of ACL*.

Meilă, M. (2003). Comparing Clusterings by the Variation of Information. In *Learning Theory and Kernel Machines*, pages 173–187. Springer.

Mikolov, T., Karafiát, M., Burget, L., Cernocky, J., and Khudanpur, S. (2010). Recurrent neural network based language model. *Proc. of Interspeech*.

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

Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In *Proc. of NAACL*.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*.

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

Miller, G. A., Leacock, C., Tengi, R., and Bunker, R. T. (1993). A semantic concordance. In *Proc. of HLT*, pages 303–308.

Minkov, E., Toutanova, K., and Suzuki, H. (2007). Generating complex morphology for machine translation. In *Proc. of ACL*.

Mnih, A. and Kavukcuoglu, K. (2013). Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*.

Mnih, A. and Teh, Y. W. (2012). A fast and simple algorithm for training neural probabilistic language models. In *Proc. of ICML*.

Mohammad, S. (2011). Colourful language: Measuring word-colour associations. In *Proc. of the Workshop on Cognitive Modeling and Computational Linguistics*.

Mohammad, S. M. and Turney, P. D. (2013). Crowdsourcing a word-emotion association lexicon. *Computational Intelligence*, **29**(3), 436–465.

Moore, R. (2015). An improved tag dictionary for faster part-of-speech tagging. In *Proc. of EMNLP*.

Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Proc. of AISTATS*.

Mrkšić, N., Séaghdha, D. Ó., Thomson, B., Gašić, M., Rojas-Barahona, L., Su, P.-H., Vandyke, D., Wen, T.-H., and Young, S. (2016). Counter-fitting word vectors to linguistic constraints. In *Proc. of NAACL*.

Müller, T. and Schuetze, H. (2015). Robust morphological tagging with word representations. In *Proceedings of NAACL*.

Müller, T., Schmid, H., and Schütze, H. (2013). Efficient higher-order crfs for morphological tagging. In *In Proc. of EMNLP*.

Myers, J. L. and Well, A. D. (1995). *Research Design & Statistical Analysis*. Routledge.

Nastase, V. (2008). Unsupervised all-words word sense disambiguation with grammatical dependencies. In *Proc. of IJCNLP*.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2014). Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proc. of EMNLP*.

Ng, A. Y., Jordan, M. I., Weiss, Y., *et al.* (2002). On spectral clustering: Analysis and an algorithm. In *Proc. of NIPS*.

Nida, E. A. (1949). *Morphology: The descriptive analysis of words.* ERIC.

Nießen, S. and Ney, H. (2004). Statistical machine translation with scarce resources using morpho-syntactic information. *Computational linguistics*, **30**(2).

Nivre, J. (2008). Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, **34**(4), 513–553.

Och, F. J. (1995). *Maximum-Likelihood-Schätzung von Wortkategorien mit Verfahren der kombinatorischen Optimierung*. Studienarbeit, University of Erlangen.

Och, F. J. (1999). An efficient method for determining bilingual word classes. In *Proc. of EACL*.

Oflazer, K. and Kuruöz, Ì. (1994). Tagging and morphological disambiguation of turkish text. In *Proc. of ANLP*.

Opper, M. and Winther, O. (2001). From naive mean eld theory to the tap equations. *Advanced Mean Field Methods: Theory and Practice. MIT Press*.

Ordway, E. B. (1913). *Synonyms and Antonyms: An Alphabetical List of Words in Common Use, Grouped with Others of Similar and Opposite Meaning*. Sully and Kleinteich.

Owoputi, O., OConnor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N. A. (2013). Improved part-of-speech tagging for online conversational text with word clusters. In *Proc. of NAACL*.

Padó, S. and Lapata, M. (2007). Dependency-based construction of semantic space models. *Comput. Linguist.*, **33**(2), 161–199.

Pang, B. and Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proc. of ACL*.

Paperno, D., Pham, N. T., and Baroni, M. (2014). A practical and linguistically-motivated approach to compositional distributional semantics. In *Proc. of ACL*.

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

Pereira, F., Tishby, N., and Lee, L. (1993). Distributional clustering of english words. In *Proc. of ACL*.

Pirinen, T. A. (2011). Modularisation of finnish finite-state language descriptionâĂŤtowards wide collaboration in open source development of a morphological analyser. In *Proc. of Nodalida*.

Rapp, R. (2003). Word sense discovery based on sense descriptor dissimilarity. In *Proc. of the MT Summit*.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proc. of EMNLP*.

Read, J., Pfahringer, B., Holmes, G., and Frank, E. (2011). Classifier chains for multi-label classification. *Machine Learning*, **85**(3), 333–359.

Reisinger, J. and Mooney, R. J. (2010). Multi-prototype vector-space models of word meaning. In *Proc. of NAACL*.

Resnik, P. and Yarowsky, D. (1999). Distinguishing systems and distinguishing senses: new evaluation methods for word sense disambiguation. *Nat. Lang. Eng.*

Roget, P. M. (1852). *Roget's Thesaurus of English words and phrases*. Available from Project Gutemberg.

Salakhutdinov, R. and Mnih, A. (2008). Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. of ICML*.

Saul, L. K. and Pereira, F. (1997). Aggregate and mixed-order markov models for statistical language processing. *CoRR*.

Sayed, A. H. and Kailath, T. (2001). A survey of spectral factorization methods. *Numerical linear algebra with applications*, **8**(6-7), 467–496.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proc. of the international conference on new methods in language processing*.

Schuler, K. K. (2005). *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.

Smith, N. A., Smith, D. A., and Tromble, R. W. (2005). Context-based morphological disambiguation with random fields. In *Proc. of EMNLP*.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*.

Soricut, R. and Och, F. (2015). Unsupervised morphology induction using word embeddings. In *Proc. of NAACL*.

Stump, G. T. (2001). *Inflectional morphology: A theory of paradigm structure*, volume 93. Cambridge University Press.

Subramanya, A. and Talukdar, P. P. (2014). Graph-based semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, **8**(4).

Subramanya, A., Petrov, S., and Pereira, F. (2010). Efficient graph-based semi-supervised learning of structured tagging models. In *Proc. of EMNLP*.

Täckström, O., McDonald, R., and Uszkoreit, J. (2012a). Cross-lingual word clusters for direct transfer of linguistic structure. In *Proc. of NAACL*.

Täckström, O., McDonald, R., and Uszkoreit, J. (2012b). Cross-lingual word clusters for direct transfer of linguistic structure. In *NAACL*.

Tipping, M. E. and Bishop, C. M. (1999). Mixtures of probabilistic principal component analyzers. *Neural computation*, **11**(2), 443–482.

Trón, V., Halácsy, P., Rebrus, P., Rung, A., Vajda, P., and Simon, E. (2006). Morphdb. hu: Hungarian lexical database and morphological grammar. In *Proc. of LREC*.

Tsoumakas, G. and Katakis, I. (2006). Multi-label classification: An overview. *Dept. of Informatics, Aristotle University of Thessaloniki, Greece*.

Tsvetkov, Y., Schneider, N., Hovy, D., Bhatia, A., Faruqui, M., and Dyer, C. (2014a). Augmenting english adjective senses with supersenses. In *Proc. of LREC*.

Tsvetkov, Y., Boytsov, L., Gershman, A., Nyberg, E., and Dyer, C. (2014b). Metaphor detection with cross-lingual model transfer. In *Proc. ACL*.

Tsvetkov, Y., Faruqui, M., Ling, W., Lample, G., and Dyer, C. (2015). Evaluation of word vector representations by subspace alignment. In *Proc. of EMNLP*.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proc. of ACL*.

Turney, P. D. (2001). Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *Proc. of ECML*.

Turney, P. D. (2002). Mining the web for synonyms: Pmi-ir versus lsa on toefl. *CoRR*.

Turney, P. D. and Pantel, P. (2010). From frequency to meaning : Vector space models of semantics. *JAIR*, pages 141–188.

Uszkoreit, J. and Brants, T. (2008). Distributed word clustering for large scale class-based language modeling in machine translation. In *Proc. of ACL*.

Utt, J. and Padó, S. (2014). Crosslingual and multilingual construction of syntax-based vector space models. *Transactions of the Association of Computational Linguistics*, **2**, 245–258.

van der Maaten, L. and Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, **9**, 2579–2605.

Vecchi, E. M., Zamparelli, R., and Baroni, M. (2013). Studying the recursive behaviour of adjectival modification with compositional distributional semantics. In *Proc. of EMNLP*.

Vıa, J., Santamarıa, I., and Pérez, J. (2005). Canonical correlation analysis (cca) algorithms for multiple data sets: Application to blind simo equalization. In *Proceedings of EUSIPCO*.

Wang, F., Wang, S., Zhang, C., and Winther, O. (2007). Semi-supervised mean fields. In *Proc. of AISTATS*.

Xu, C., Bai, Y., Bian, J., Gao, B., Wang, G., Liu, X., and Liu, T.-Y. (2014). Rc-net: A general framework for incorporating knowledge into word representations. In *Proc. of CIKM*.

Yarowsky, D. and Ngai, G. (2001). Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proc. of NAACL*.

Yih, W.-t., Zweig, G., and Platt, J. C. (2012). Polarity inducing latent semantic analysis. In *Proc. of EMNLP*.

Yogatama, D. and Smith, N. A. (2014). Linguistic structured sparsity in text categorization. In *Proc. of ACL*.

Yu, M. and Dredze, M. (2014). Improving lexical embeddings with semantic knowledge. In *Proc. of ACL*.

Zhang, M.-L. and Zhou, Z.-H. (2005). A k-nearest neighbor based algorithm for multi-label classification. In *Proc. of IEEE Conf. on Granular Computing*.

Zhang, Y. and Nivre, J. (2011). Transition-based dependency parsing with rich non-local features. In *Proc. of ACL*.

Zhang, Y. and Schneider, J. G. (2011). Multi-label output codes using canonical correlation analysis. In *Proc. of AISTATS*.

Zhila, A., Yih, W.-t., Meek, C., Zweig, G., and Mikolov, T. (2013). Combining heterogeneous models for measuring relational similarity. In *Proc. of NAACL*.

Zhu, X., Lafferty, J., and Rosenfeld, R. (2005). *Semi-supervised learning with graphs*. Carnegie Mellon University.