LEXICAL ENTAILMENT

BY

JOHN WIETING

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Advisor:

Professor Dan Roth

Abstract

Lexical entailment is a requirement for success in the domains of Recognizing Textual Entailment (RTE) as well as related tasks like Question-Answering and Information Extraction. Previous approaches tend to fall into two camps - those that make use of distributional models and those that make use of knowledge bases such as WordNet. Interestingly, these methods make very different kinds of mistakes and so in this thesis, we construct a new entailment measure by combining these two paradigms in such a way that exploits their differences. We also experiment with including local context and modify an existing approach to achieve the best unsupervised performance so far on the Lexical Substitution task. Overall, we achieve a significant gain in performance on three different evaluations and our approach is also faster than the other distributional approaches we compare to as we are able to avoid fruitless comparisons. Furthermore, we introduce a new approach to evaluate lexical entailment that avoids some of the issues of wordlists - the current conventional way of evaluating lexical entailment, and that can be additionally used to evaluate lexical entailment in context - a novel task introduced in this paper. We also include experiments that show our new lexical entailment model improves performance on the RTE task, the main goal of this work.

To my mother and father who have always supported me.

Acknowledgments

Thanks to everyone in the Cognitive Computation Group for their advice and support when doing this research. Most notably to Mark Sammons who was a great source of feedback and of course my adviser, Dan Roth.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recognizing Textual Entailment (RTE) is an important task in Natural Language Processing as it is used in many applications like Information Extraction and Question Answering. The RTE problem is to automatically decide if one piece of text entails another. That is if the first text is true, to determine if that means the second is also true. The starting point for many RTE systems is to align the tokens between the hypothesis and the text that have the same relationship. In general, there are many relationships between the tokens that need to be understood and included in the alignment and often separate tools are needed to identify these. For instance, it is important to have an entailment measure for named entities, dates, and quantities. It also important to have an entailment measure between words. Additionally, there are other relationships between words that must be known as well, such as derivationally related words and relational antonyms, but this is not the focus of this work.

The focus of this thesis is on lexical entailment. This can be viewed as RTE between tokens themselves. In other words, we would like to know when one word implies another like *dog* implies *animal*. This was formally defined in [Zhitomirsky-Geffet, 2009] and again in [Turney and Mohammad, 2013]. Lexical entailment was also studied in [Kotlerman et al., 2010], [Baroni et al., 2012]. Just as in lexical similarity, there are two main approaches. The first utilizes a knowledge base, most often this is WordNet [Miller, 1995]. The second is a vector space model where each word is represented as a vector. These vectors are often based on statistics from a large corpus.

Interestingly, these two different approaches compliment each other as they tend to make different kinds of mistakes. Vector space models give high scores to words that can often be found in the same contexts. They are thus capable of identifying synonyms and hypernyms. However, they are also noisy as they cannot often keep from giving high scores to other lexical relationships like antonyms and co-hyponyms because these often share the same contexts as well. WordNet based measures can make mistakes primarily for three reasons. The first is that the relationship may not exist in WordNet. WordNet has an extremely large vocabulary, but it is missing some useful relationships between words. The second is that WordNet includes many senses for each word. Some of these senses are very rare. For instance, *person* is a hypernym with a distance of 2 synsets of *computer*, and *turtle* and *sweater* are synonyms. The third issue is that the network in WordNet is not uniform - sometimes a hypernym that will likely be useful is many synsets away in the hierarchy. These issues present challenges

to lexical entailment measures using WordNet as they may identify relationships that are technically correct but not needed in the current context or miss some relationships altogether. One approach for addressing the former problem is to use word sense disambiguation. However, it has been difficult to obtain enough performance on this task using the WordNet senses for this to be viable and even then, one would still have to work with the other limitations of WordNet.

Given that distributional models measure contextual overlap and knowledge bases give correct information about the relationships between words, it would seem that these could be combined in a way to create more useful lexical entailment measures. In this paper, we create an improved lexical entailment measure by using information collected from a large corpus to score knowledge base information. We use synonyms, antonyms and hypernyms from WordNet, and lexical relationships from the paraphrase database [Ganitkevitch et al., 2013] as a source of knowledge and then create a new entailment measure. We also are able to extend this approach to incorporate the local contexts of a word pair. This model greatly reduces mistakes made due to mismatched word sense with little additional computational cost.

We evaluate our new method and compare to other approaches on the RTE task. We also evaluate on the wordlist dataset introduced in [Kotlerman et al., 2010]. Finally, we introduce a new evaluation made up of examples of lexical entailment taken from the MSR RTE2 alignment corpus [Brockett, 2007]. The conventional evaluation has been to evaluate on applications or on wordlists. Evaluating on applications can make error analysis difficult, and the problem with wordlists is that it is difficult to generate negative examples that are sufficiently close to the margin. Additionally, there is a chance that the examples are not natural in the sense that we are likely to encounter them in a real world setting. Thus we used the alignment corpus to create an evaluation that would not have these issues. Moreover, this evaluation can also be used to evaluate lexical entailment in context - a new task we propose that is most similar to the sense matching [Dagan et al., 2006] and lexical substitution [McCarthy and Navigli, 2007] tasks.

Overall, our method significantly outperforms other entailment and similarity measures in the literature on all evaluations. It also has the additional benefit of being significantly faster than purely vector space entailment approaches, allowing us to include local context without much additional cost, as our method is judicious on what word comparisons to make.

Chapter 2

Literature Review

The literature on lexical entailment is fairly sparse and recent. However research tangentially related on lexical similarity has been a topic of interest for decades. I will first discuss lexical entailment and how it relates to similar research in word similarity and relatedness. I will then discuss the two main paradigms to solving this important problem. The first is that of distributional methods which rely on using a large corpus to create a vector representation of a word, and the second are knowledge bases methods which can be either constructed manually by humans or constructed using automatic techniques.

2.1 Lexical Entailment

2.1.1 Definitions

Lexical entailment has been defined in two different ways in the literature. The earliest definition, in [Zhitomirsky-Geffet, 2009], defines lexical entailment based on substitutability.

Definition Given words w and v, we say that w **substitutability entails** v if two conditions hold. The first is that the meaning of a possible sense of w implies a possible sense of v. The second is that w can substitute for v in some naturally occurring sentence, such that the meaning of the modified sentence would entail the meaning of the original one.

[Turney and Mohammad, 2013] critique this definition by building upon a comment in [Zhitomirsky-Geffet, 2009] where the authors state that lexical entailment is more complicated than a superset of known lexical relationships like synonymy, hypernomy, and meronymy. Turney suggests using finer grained semantic relations and so defines lexical entailment as a collection of semantic relations in Bejar's taxonomy [Bejar et al., 1991], which consists of 10 coarse categories that are further divided giving 79 total classifications. Turney and a colleague then went through these relations and decided if they were indicative of entailment or not based on their relational definition of entailment.

Definition Word w is said to **relationally entail** word v if there exists a pre-defined semantic relation between w and v. Secondly, it should follow from the meaning of

that relation that w entails v. Lastly, if two or more semantic relations are possibly between w and v and they do not agree, then it is assumed that w does not entail v.

2.1.2 Criticism

Turney's definition is more general than that in [Zhitomirsky-Geffet, 2009]. However it has some major drawbacks. For one, it is a much more difficult definition to apply (based on lower inter-annotater agreement) and is entirely dependent on the set of semantic relations that are being considered. Furthermore, it can be argued that the definition is not as useful. One part of the 2012 Sem-Eval task was to order word pairs by how strongly they exemplify one of these 79 relations given several gold pairs. The best performing system [Mikolov et al., 2013c] though had very low Spearman's ρ with the gold scores (27.5), suggesting that classifying these relations with a high enough degree of precision is currently not something that can be done. Turney was simply concerned with identifying whether two words satisfy his relational entailment definition. However, in order to be useful in an entailment system, I argue that we must know exactly how the words are related, especially in an alignment model which was one of the main motivations of these investigations into lexical alignment. Just knowing that they satisfy lexical entailment, according to Turney's definition, is not useful as there is a large difference from an inference point-of-view between $rain \models wet$ and $tulip \models flower$. While an alignment between $tulip$ and $flower$ would aid RTE inference, the relationship between $rain$ and wet would require a more sophisticated inference than what is currently used by alignment models to make the correct entailment decision.

We take the view that lexical entailment should be closely tied to the inference used in RTE systems. Thus other lexical relationships are also of interest that are necessary for the inference used in alignment-based RTE systems that fall outside of these two definitions. These include words that are derivationally related like *gain* and *growth* as well as relational antonyms like *buy* and *sell* and of course multi-word expressions like *take part* and *on the flip side*. For this thesis though, we focus just on the substitutable definition of entailment as it is the most commonly encountered in the RTE datasets and is something that our current alignment algorithm [Chang et al., 2010] is built upon.

2.1.3 Relationship to Similarity and Relatedness

It is important to place lexical entailment in the context of the other major lexical relationships - similarity and relatedness. Lexical entailment can be seen as a directional relationship that is a subset of the more broad category of lexical similarity. It is also important to recognize the similarities and differences between these lexical relationships as they each require their own approaches.

[Agirre et al., 2009] first made the distinction between similarity and relatedness. Words were deemed to be related if they had a meronym-holonym or meronym-holonym relationship or a high enough human similarity score (were seen as topically related)

without having a hypernym-hyponym, hyponym-hypernym, meronym-holonym, holonym-meronym, antonym, or synonym relationship. There were related if they had a synonym, antonym, hyponym-hypernym, hypernym-hyponym relationship.

Lexical entailment, therefore according to the substitution definition, can be seen to roughly correspond to the subset of lexical similarity consisting of synonyms and hyponym-hypernyms. This is an important observation as it tells us that ideas from lexical similarity could be useful when creating lexical entailment approaches. An example of this would be that it is better to use a target word's syntactic dependencies as features in our word representations over just the unigrams in a window around that target word as the former favors similarity and the latter relatedness.

2.2 Distributional Methods

The motivation for distributional methods lies in the distributional hypothesis which states that words with the same meaning tend to occur in the same contexts [Harris, 1954]. In practice, words are represented as vectors where the components of the vectors are features indicating some context type.

2.2.1 Statistics Based Distributional Methods

The most popular and oldest techniques for creating word representations stem from computing statistics from a large corpus. In general we require a weighting function f and some definition of context c. Examples of weighting functions include the identity, TF-IDF, Lin98a [Lin, 1998b], point-wise mutual information, TTest, and frequency. Some are listed below.

$$PMI = \log(\frac{p(w, r, w')}{p(w, *, *)p(*, r, w')}) \tag{2.1}$$

$$Lin98a = \log(\frac{|w, r, w'| \cdot |*, r, *|}{|*, r, w'| \cdot |w, r, *|}) \tag{2.2}$$

$$TTEST = \frac{p(w, r, w') - p(*, r, w')p(w, *, *)}{\sqrt{p(*, r, w')p(w, *, *)}} \tag{2.3}$$

Then to compute the similarity between words, we also require a similarity function between vectors. Possible approaches here include the L1 or L2 norm, cosine (probably the most commonly used), Lin [Lin, 1998b], Jacard, Dice measure, KL-Divergence, and many others. The cosine function is shown in (2.4) and Lin in (2.5).

$$cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{|\mathbf{u}| \cdot |\mathbf{v}|} \tag{2.4}$$

$$Lin(\mathbf{w_1}, \mathbf{w_2}) = \frac{\sum_{(r,w) \in \mathbf{w_1} \cap \mathbf{w_2}} Lin98(w_1, r, w) + Lin98(w_2, r, w)}{\sum_{(r,w) \in \mathbf{w_1}} Lin98(w_1, r, w) + \sum_{(r,w) \in \mathbf{w_2}} Lin98(w_2, r, w)} \tag{2.5}$$

The choice of weighting function and similarity function can have a significant effect on performance. [Curran, 2003] evaluates these many different measures and weighting functions by obtaining the top 200 most similar words for each measure and then comparing the rankings of these with the union of entries in multiple thesauri. He finds DICE and Jaccard to be the top performing similarity measures and TTest, with point-wise mutual information close behind, to be the best performing weighting functions.

Note that there are other approaches where the weights for all words in the vocabulary are placed in a $V \times D$ matrix where W is the size of the vocabulary and D is the size of the feature vectors. Then the dimensionality of this matrix is reduced by mapping it into a $V \times d$ space where $d << D$. Singular Value Decomposition (SVD) is often used to do this transformation [Landauer and Dutnais, 1997].

Window Based Word Contexts

Window based methods have several design choices in addition to which weighting function f to use. One of these is the width of the window (how many words does the window extend over). In general, the smaller the window, the more one is capturing lexical similarity, and as it widens, we are capturing more of lexical relatedness (i.e. words that are more topically related). Other decisions include the symmetry of the window, which means where in the window will the target word be placed, and whether the window is fixed regardless of boundaries due to paragraphs or page breaks. The effect of these decisions was investigated by [Curran, 2003]. One last decision, with this approach is on whether to filter out high frequency but uninformative words or very rarely used words. This is often seen to be a good idea as it lowers the computational cost and usually has no negative effect.

Furthermore, Window Based methods do not simply need to just define a context c as a word. There are other choices as well such as including features like bigrams and the direction (to right or left of the target).

Grammatical Relation Based Word Contexts

Instead of using every word in a context window, we can use only those words that occur in a grammatical relation or dependency of the target word as [Hindle, 1990] suggests. Now the feature space consists of a word and a relation. As full dependency parsing is expensive, [Curran, 2003] investigates using a chunker and shallow parser with the goal of more quickly extracting a smaller set of relations like subject, direct object, and prepositional object of certain prepositional phrases. He compared to using dependency parses and found not surprisingly, that the performance of these approaches on his lexical similarity evaluations scales with the sophistication of linguistic processing involved. Thus, those making use of dependency parse have the highest performance. However he notes, that the difference diminishes as the size of the corpus increases.

2.2.2 Neural Network Word Embeddings

[Bengio et al., 2003] created the first neural network language model where word vector representations are jointly learned with the language model. His was a feedforward architecture that was trained in a supervised fashion on raw text where the goal was to maximize the log-likelihood of the text. Thus, it required no training data. This inspired more neural network language models like in [Collobert and Weston, 2008] and [Mnih and Hinton, 2008].

In [Mikolov et al., 2010], the authors investigated using a recurrent neural network architecture for language modeling. The representations created from this model were shown to surpass other neural network word embeddings in [Mikolov et al., 2013b] on various word similarity evaluations.

Most recently, [Mikolov et al., 2013b] and [Mikolov et al., 2013a] discuss a Skip-gram model for learning word representations. To train, the input to the network is a single word and the output is the nearby context words from some occurence of the input word in a large corpus. This is different from other models where the input is usually the context and the goal is to predict the word in the middle or end of that input context. The training is efficient as it does not require dense matrix multiplications and so the authors were able to train it on 100 billion words in a single day on a single machine. The authors show these word representations outperform all otheres on the word analogy task. [1].

2.2.3 Distributional Lexical Entailment

BalAPincs

This approach, described in [Kotlerman et al., 2010], aims to reward those situations when the first context vector argument is a subset of the second. In other words, the features of the first context vectors should be included in the second. This idea naturally comes from the distributional inclusion hypothesis [Geffet, 2005], which states that if word *a* occurs in a subset of the context of word *b* then *a* often entails *b*. The formula for calculating balAPinc is below. F_i denotes the context vector where all nonzero entries have been removed. In practice, this feature vector includes only the top 1000 or so features to prevent low occurring features from influencing the score and to also increase the speed of the algorithm. The features are weighted by Lin98. In our work we also found it useful to put a limit on the number of times a term occurs before including it in the feature set. This is because, like PMI, Lin98 is sensitive to rare terms having a disproportionate weight. We set this value to 150.

$$\text{balAPinc}(u, v) = \sqrt{\text{APinc}(u, v) \cdot \text{Lin98a}(u, v)} \tag{2.6}$$

$$\text{APinc}(u, v) = \frac{\sum_{r=1}^{|F_u|} |P(r, F_u, F_v) \cdot rel(f_{ur}, F_v)|}{|F_u|} \tag{2.7}$$

[1] These 300 component word representations are available at https://code.google.com/p/word2vec/

$$\mathrm{P}(r, F_u, F_v) = \frac{|\mathrm{inc}(r, F_u, F_v)|}{r} \tag{2.8}$$

$$\mathrm{rel}(f, F_w) = \begin{cases} 1 - \frac{\mathrm{rank}(f, F_w)}{|F_w|+1} & \text{if } f \in F_w \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

$$\mathrm{inc}(r, F_u, F_v) = \{f | \mathrm{rank}(f, F_u) \leq r,\, f \in (F_u \cap F_v)\} \tag{2.10}$$

Supervised Approaches

Recently there has also been two additional distributional lexical entailment approaches that use a supervised approach. The main draw back with these is that they are a function of their training data.

The first of these, ConVecs [Baroni et al., 2012] (short for *concatenated vectors*) operates under the hypothesis that the entailment of words *a* and *b* is a learnable function of the concatenation of their context vectors. The authors use SVD to reduce the dimensionality of the context vectors and then use a SVM with a polynomial kernel to learn this function. In order to obtain a score from this model, probability estimates can be used and the probability for the positive class can be used as the score.

In [Turney and Mohammad, 2013], he introduces a measure called SimDiffs. This operates under the hypothesis that the tendency of word w to entail word v is correlated with some learnable function of the difference in their similarities to a set of reference words. In SimDiffs, two matrices are created, a domain matrix **D** and a function matrix **F**. The domain matrix was created by using nouns that occur near a given word as the context, while the function matrix uses the verbs. **D** is designed to measure the domain similarity between two words (similarity of topic), while **F** is designed to measure the functional similarity (how the words are used). The matrices were decomposed using SVD and PPMI (PMI that is thresholded below at 0) was used to weight the context features. The features sets then for $w \models v$ are the following where R is a set of reference words:

$$S_1 = \{sim_d(w, r) - sim_d(v, r) | r \in R\}$$

$$S_2 = \{sim_f(w, r) - sim_f(v, r) | r \in R\}$$

$$S_3 = \{sim_d(w, r) - sim_f(v, r) | r \in R\}$$

$$S_4 = \{sim_f(w, r) - sim_d(v, r) | r \in R\}$$

The set of reference words he used for his experiments are the 2086 words in Ogden's Basic English. Cosine similarity was used as the similarity function, and the model was trained using SVM with radial basis kernel.

Criticism of Supervised Approaches

One issue with supervised approaches is that they are a function of their training data which poses a difficulty as it hard to discern negative examples that are close to the margin. Moreover, the experiments in [Turney and Mohammad, 2013] comparing balAPincs, Convecs, and SimDiffs give mixed results on the datasets motivated by the substitutable definition of entailment.

2.3 Knowledge-Based Metrics

Knowledge based approaches make use of a knowledge base that contains information about single words or phrases as well as relations between them. Among the knowledge bases used, WordNet [Fellbaum, 1998] has been probably the most utilized and has proven to be a useful resource. Others include dictionaries, thesauri, and automatically constructed resources like the Paraphrase Database [Ganitkevitch et al., 2013].

2.3.1 WordNet Similarity

A number of measures have been developed to quantify the lexical similarity or entailment of two given words [Budanitsky and Hirst, 2006]; in the context of WordNet, for example, these can often be thought of as related so some distance between the two target words along a path in the Wordnet graph. Method vary on how the path is chosen (e.g., which relations are taken into account) and how it is scaled [Wu and Palmer, 1994, Resnik, 1995, Lin, 1998a, Jiang and Conrath, 1997, Leacock and Chodorow, 1998, Hirst and St-Onge, 1998, Banerjee and Pedersen, 2002, Do et al., 2009].

The above metrics are a subset of the following variables when comparing words w_1 and w_2: the length of the shortest path between w_1 and w_2 in WordNet, length of the shortest path between one of w_1 and w_2 and the lowest common subsumer (LCS) - which is the closest node in the network that both words descend from - of w_1 and w_2, the length of the shortest path from the LCS to the global root, the maximum depth of the hierarchy, probability of w_1 or w_2 as measured by a corpus, and the probability of the LCS of w_1 or w_2. Additionally, HSO also takes into account the number of changes in path direction of the path connecting two words in the network. Of these seven algorithms only HSO and Lesk are able to assign scores for words having different parts of speech.

2.3.2 Paraphrase Database

The Paraphrase Database (PPDB) [Ganitkevitch et al., 2013] was constructed by extracting lexical, phrasal, and syntactic paraphrases from a large bilingual parallel corpora. They used the idea that two English strings that translate to the same foreign string can be assumed to have the same meaning. They can pivot over the foreign string and map the English paraphrases to each other. They then were able to calculate

the marginal probability of these phrase alignments by marginalizing over all shared foreign-language translations.

One can obtain an entailment score from the paraphrase database by using the following formula where $P(e|f, LHS)$ is the score in the PPDB that encodes the probability that f is a paraphrase of e:

$$\frac{1}{P(e|f, LHS)} \tag{2.11}$$

2.3.3 Other Recent Approaches

Recently there has been some new approaches which aim to enhance knowledge bases. One of these is PILSA [Yih et al., 2012]. The aim of this work was to develop a vector space representation in which synonyms and antonyms were on opposite ends of a sphere lying in a continuous vector space. The advantage over a standard thesaurus is that new synonyms and antonyms can be discovered and using information from a large corpus, new words can be embedded into the subspace. The authors accomplished this through a modification of LSA. The documents in the model are thesaurus entries for a given word including synonyms and antonyms. Each row in the co-occurrence matrix is a vector representation of a document whose components are the TF-IDF of the words in its entry. To create the desired property of separating antonyms and synonyms, the TF-IDF scores for the antonyms are negated. They then enhance the projection matrix, created by LSA, through discriminative training using a neural network architecture called an S2-Net.

Another approach that enhances a knowledge base, is an extension of PILSA that makes use of tensor decomposition. In [Chang et al.,], the authors combine multiple relations between words, like hypernymy, synonymy and antonymy, by constructing a 3-way tensor. Then instead of using LSA to decompose, as in the case of a matrix, they decompose it using tensor decomposition. This allows them to include information from multiple sources (Encarta dictionary and WordNet) and they can generalize to cover relations not mentioned in these resources. The degree to which a word satisfies a given relation can be obtained with just simple algebraic manipulations.

2.3.4 Knowledge Based Lexical Entailment

WNSim

WNSim was introduced in [Do et al., 2009] and is a WordNet similarity metric designed to perform well on the RTE task.

$$\text{WNSim}(w1, w2) = \begin{cases} \theta^{l_1+l_2} \text{ if } l_1 + l_2 \leq k \\ \theta^{k} \text{ if } l_1 + l_2 \leq \\ \quad \alpha \cdot \text{depth of } lcs(w1, w2) \\ 0 \text{ if } l_1 + l_2 \leq k \end{cases}$$

Where the parameter values used were $\theta = 0.3$, $k = 3$, and $\alpha = 0.667$, learned through a grid search using Lexical Level Matching on some RTE data. l_i refers to the distance from the synset of w_i to the lcs. Note that this is a symmetric similarity measure and does not give an entailment direction. In WNSim, stems (if they exist) are chosen for each word from each part-of-speech. Then to compute the least-common-subsumer, the synonymy-antonymy, hypernymy-hyponymy, are considered.

Chapter 3

Evaluation of Lexical Entailment

3.1 WordLists

The usual evaluation for lexical entailment, done in [Baroni et al., 2012], [Kotlerman et al., 2010], [Turney and Mohammad, 2013], is to evaluate on lists of word pairs. A threshold is learned on a held out portion of the data and then evaluation is done on the remaining word pairs - some of which lexically entail each other and some do not. A similar approach is also used in word similarity evaluation where wordlists like those introduced in [Agirre et al., 2009], [Finkelstein et al., 2001], [Rubenstein and Goodenough, 1965] are used and the task is to have the model produce scores for the words that are correlated with human judgments. There are at least three issues with this approach. The first is that it can be difficult to construct negative examples that are near the margin. The second is that these word lists may not be very representative of the word pairs that occur in natural text. Lastly, we cannot include the context into the evaluation if we are just given a list of words. Additionally, for the cases where human judgment is used, there is no guarantee that good correlation with human judgments will manifest itself as good performance on NLP applications.

3.2 Recognizing Textual Entailment Alignments

Our main evaluation method, which also helped motivate the ideas in this paper, is the dataset we constructed from the aligned RTE2 data from [Brockett, 2007]. This type of evaluation was designed to avoid those three issues that affect wordlist evaluations. To create the dataset, we processed the aligned corpus by first removing all alignments between named entities, numbers, words that are exact matches or differ only up to stemming, and stop words. This left us with 1041 alignments. These remaining alignments were then categorized into several different categories by two annotators. These categories included lexical entailment, where we used the substitutable definition of lexical entailment in [Zhitomirsky-Geffet, 2009], word pairs that would lexically entail if they were the correct part of speech like *grow* and *gains* and derivationally related words, relational antonyms like *buy* and *sell*, and finally multi-word expressions. The task was then, given the aligned word in the hypothesis and a relation, to identify its match in the text. The evaluation used 20-fold cross validation, where a threshold was tuned on some held out set, and then use this threshold to calculate precision, recall,

and F1 on a test set. This latter use case reflects how lexical entailment and word similarity measures are often used in real systems. For this paper we only used 289 lexical entailment examples for evaluation and leave the other relations for future work. For lexical entailment, the kappa score was 74.9 indicating substantial agreement.

This dataset, we argue provides a better evaluation for lexical entailment than any previous resource and resolves the problems with wordlist evaluation. Negative examples are the surrounding words in the sentences and so mistakes on these will correlate well with mistakes on real NLP tasks. The aligned word pairs are also representative of those that can be found in natural text. Lastly, we can evaluate lexical entailment algorithms that do include context. If a system does well on our evaluation, it is making the correct alignments which is often a crucial first step in RTE, thus success on our task can be seen to be directly related to success on the goal task of RTE - this is harder to justify for wordlist evaluations.

As previously mentioned, our dataset can be used to assess the performance of lexical entailment in context, a novel task. There is related work in this area. For instance, [Dagan et al., 2006] introduces a dataset that gives a pair of two synonymous words and a set of sentences containing one of the words in this pair. The goal is to determine whether the context in each of the sentences is expressing the sense illustrated by the word pair. There is also the lexical substitution task [McCarthy and Navigli, 2007] where the task is to find an alternative word for a target word in context. The gold standard in the lexical substitution task are words proposed by human judges. In our tasks, we are given two target words and two contexts, and the task is to determine if the words lexically entail in these contexts. This more closely matches the real use case in RTE as well in related tasks like Question-Answering and Information Extraction.

3.3 Recognizing Textual Entailment LLM

LLM is an evaluation in the context of an application. This was initially used to evaluate lexical entailment in [Do et al., 2009].

This evaluation can be seen as taking, for each word in the hypothesis, the maximum score between this word and all words in the text. This is repeated for each non-stop word in the hypothesis and the average of all these values is returned for the score. It can be seen as scoring an alignment between the hypothesis and the text (3.1), where s_i is the set of tokens in sentence i.

$$LLM(s_1, s_2) = \frac{\sum_{v \in s_2} \max_{u \in s_1} \text{sim}(u, v)}{\|s_2\|} \tag{3.1}$$

In [Mihalcea et al., 2006], the author proposes a formula similar to the above in order to compare different lexical measures on the MSR paraphrase corpus. The formula is shown in (3.2) where $maxSim(w, s)$ is the maximum similarity score between word w and all words in text s. The advantage of this formulation over (3.1) is that it takes word specificity into account by incorporating the idf of the words. Thus words that are more rare have more weight in the final score - there is no need for a stop-word

list as these naturally have very low idf.

$$sim(s_1, s_2) = \frac{1}{2}\frac{\sum_{w \in s_1} maxSim(w, s_2) \cdot idf(w)}{\sum_{w \in s_1} idf(w)} + \frac{1}{2}\frac{\sum_{w \in s_2} maxSim(w, s_1) \cdot idf(w)}{\sum_{w \in s_2} idf(w)} \tag{3.2}$$

We modified this formula for use in RTE by just matching the hypothesis tokens with those in the text. This gives the formula (3.3) used in our LLM experiments. This formulation gives better results on our RTE experiments than that in (3.1). The idf values were calculated from the documents in the NYT segment of Annotated GigaWord [Napoles et al., 2012].

$$newLLM(s_1, s_2) = \frac{\sum_{w \in s_2} maxSim(w, s_1) \cdot idf(w)}{\sum_{w \in T_2} idf(w)} \tag{3.3}$$

All of the examples in the RTE dataset are then assigned a score and an optimal threshold is tuned on the development set. This threshold is used in evaluation on the test set.

One difficulty with LLM is that each metric has its own range of scores and its own distribution over this range. In order to solve this problem, we evaluate with the binary version of LLM which is meant to make the metrics less reliant on the range and distribution of the scores they give. For each metric, a threshold was learned through grid search again on the development data. If the score of the metric surpassed the threshold, a score of 1 was assigned to the word pair. Otherwise a score of 0 was assigned.

For our experiments, we used the RTE set from SemEval 2014 Task 1. Parameters were tuned on 500 sentence pairs in the development set and tested on the remaining 9500 pairs in the training and test sets. The dataset was chosen for its large number of examples and the relative simplicity of the sentence pairs where lexical methods can be expected to do fairly well. We report F_1 as the dataset is not balanced (Positive examples are labeled *Entailment* while negative examples are labeled *Neutral* or *Contradiction*).

Chapter 4

New Approaches to Lexical Entailment

4.1 Analysis

As mentioned in the second chapter, two families of metrics have been used in the literature for computing lexical entailment and lexical similarity – metrics that are based on knowledge bases, and those that are based on vector space representations of words (also called distributional or corpus-based approaches) [Mihalcea et al., 2006].

While these two families have been used in a range of applications it is interesting to observe that each family has some systematic deficiencies that result in characteristic mistakes each family of methods is making.

Figure 4.2 exemplifies typical mistakes made by a successful WordNet-based measure, WNSim [Do et al., 2009]. It illustrates faulty alignments made for the word *deal*, with *make, release* and *billion*. In this case, a sense of *deal* also contains words like *heap* and *pile*. A hypernym of this synset is the concept of a *large, indefinite quantity*, which is also a hypernym of *billion*, hence the co-hypernym based alignment. A similar analysis explains the alignments made with *make* and *release*. While these relationships are logically sound given the graph structure of WordNet, they result in two types of common mistakes: first, paths that correspond to rare senses are being followed and suggest similarities that are only appropriate in a very narrow set of contexts; second, highly polysemous words like *make* are prone to these mistakes as they often have ties to a large collection of words - most of which would rarely, if ever, be aligned in any "natural" context.

Frye says, that he (a homeopathy expert) and Iris Bell recently studied homeopathic treatment of fibromyalgia. A new analysis - comparing published studies of homeopathic drugs[1] to matched, randomly selected studies of medical drugs - suggests that these apparent homeopathic drug effects are merely placebo effects.

What really irks Frye and other doctors of homeopathy, however, is that homeopathic[2] remedies[3] are not supposed to be used like medical[4] drugs.

Figure 4.1: Characteristic mistakes of a Vector Space based measure: *drugs*[1] should be aligned with *remedies*[3] but, instead, it is aligned *homeopathic*[2] and with *medical*[4], due to a large number of related contexts.

Figure 4.1 exemplifies typical mistakes made by a vector space measure; in this case, one of the most widely cited vector space lexical entailment models, balAP-

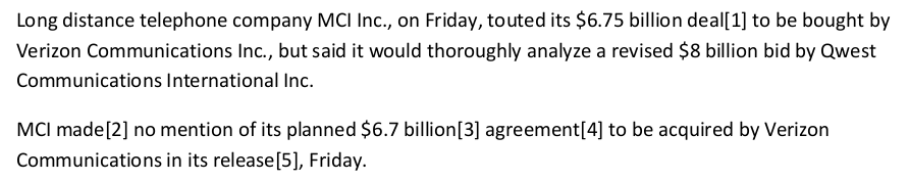

Figure 4.2: Characteristic mistakes of a WordNet-based measure: While *deal*[1] is correctly determined to be entailed by *agreement*[4], it is also entailed by *billion*[3] and *make*[2], a result of following paths in the WordNet graph that correspond to rare senses (leading to billion) and highly polysemous words (make).

incs [Kotlerman et al., 2010]. In this case, the score for the gold alignment, *drugs* and *remedies* falls below the threshold. However, the scores relating *drugs* and *homeopathic* and, similarly, *drugs* and *medical* are higher. While these words are certainly related to *drugs* as they share a common topic, they do not entail *drugs*. This is a typical example, illustrating that vector space methods can often be biased to reflect *relatedness* rather than entailment.

These examples illustrate the disparity in the types of mistakes made by these two different paradigms for lexical entailment. The analysis suggests that these can be combined: knowledge based methods can be used to constrain which relationships are sensible, and a distributional method can be used to score the strength of a relationship and thus guide a better selection of paths in the WordNet graph, for example. We next suggest our algorithm for combining these two different approaches.

4.2 Combining Distributional and Knowledge Base Approaches

The motivation for our method is that we wanted to create a framework where local context could be included if available. We also wanted a very precise lexical entailment metric that minimized the mistakes made by both vector space methods and knowledge resources. The first step of both our methods is to check a knowledge base in order to see if the entailment relation exists. Then if it does, we want to score the strength of this relationship in such a way that the scores are indicative of the likelihood of entailment.

We present two main approaches each with several variations. The first approach KBDist does not make use of local context. The second algorithm, C-KBDist , is an extension of KBDist and incorporates local context.

In KBDist we check if there is a synonym or hypernym relationship in WordNet or if the tokens were aligned in the paraphrase database. We used the XXL lexical paraphrases from the database. If the relationship has been confirmed, we then simply score with a distributional measure. Any knowledge base could be used and our method could be extended to also use dictionaries or thesauri. Similarly any distributional measure could be used to score the relationship.

We choose to use WordNet and the paraphrase database for two reasons. WordNet

provides a vast network of hypernyms that are important for lexical entailment, but it is very sparse when it comes to synonyms. Not to mention that its coverage can be lacking. The paraphrase database, while not without errors, is a great supplement to WordNet as it's strength is a vast collection of what are mostly synonyms. Some of these synonyms are words that are in WordNet, but the relationship is not. An example would be *person* and *people*. It also contains many synonym pairs that are not in WordNet like *smoke-free* and *smokeless*, not to mention it contains alternate spellings of words and abbreviations that are not included in WordNet. Some of the noise was removed from PPDB by first making sure that the words were not antonyms in WordNet.

An extension of these algorithms has been proposed, but so far the results are not nearly as good as the original version due to lots of erroneous information being interjected into the knowledge base. This extension seeks to extend the word relationships in the knowledge base. The idea is that we can improve the coverage of the WordNet and PPDB database by stipulating that if PPDB aligns two words u and v and s is a hyponym of u, then s is also a hyponym of v. An example of this being useful is that *people* and *person* are in different synsets in WordNet and *worker* is a hyponym of person, however if we are trying to align *people* and *workers*, the connection will not be made. However, with this addition to the knowledge base, we will be able to determine *workers* $\models$ *people* as *people* and *person* are aligned in the PPDB. The issue with this approach is that PPDB occasionally aligns hyponyms-hypernyms and so we must stop the search up the hypernym tree stop when it has reached a less specific concept. For instance, it PPDB aligned *people* and *entity*, then we would be erroneously saying any noun could entail *people* as all are hyponyms of *entity*. One way to measure this specificity of a synset is the use of Information Content, which is used in the Resnik WordNet similarity measure [Resnik, 1995]. Thus it seems natural that imposing some constraint that we restrict our hypernym search to those synsets with less Information Content than what we aim to align it to, possibly with some margin, we should be able to alleviate this issue.

4.3 Contextualized Vector Representation

C-KBDist attempts to incorporate local context into making lexical entailment decisions. Since our goal is a more precise entailment measure, having a way to score the likelihood of entailment by taking the local context, and hence the sense, into account should give a better estimate of this likelihood for the case of the actual sentences of interest.

One task in the literature, related to this goal of using local context to determine the lexical substitutably of words, is the Lexcial Substitution task from 2007 SemEval [McCarthy and Navigli, 2007]. In this task, the goal is to pick the most likely substitute for a given word in a sentence. It is related to the Word Sense Disambiguation (WSD) task as they both aim to elucidate the meaning of the word. However one major difference, is that this task is completely independent of a sense inventory and so it

can overcome the issues of the granularity of sense distinctions. This is important for lexical entailment as limiting ourselves to a sense inventory means that we can only declare lexical entailment if we identify both words to belong to entailing senses and so we are relying on the inventory being vast and complete, not to mention having a reliable WSD algorithm which is hefty requirement as the best systems do not do much better than beating the most frequent sense baseline [Navigli et al., 2007].

Extending the Lexical Substitution task to aid in lexical entailment is straightforward according to the substitutable definition of lexical entailment. We simply require that for word w to entail v we must find that the terms entail using our knowledge base just as in KBDist . If there is an entailment relationship, then we score it using a modification of the simple yet state-of-the-art unsupervised lexical substitution algorithm introduced in [Thater et al., 2011].

4.3.1 Lexical Substitution

[Thater et al., 2011] introduced a lexical substitution scheme based on *contextualized word representations*. This means that the dimensions of a word representation are re-weighted by the context, in this case direct syntactic dependents. The simplest case is just to retain those dimensions corresponding to just the syntactic neighbors. This creates a very sparse vector with zeros for all but a few components. Performance can be improved by not only maintaining the components corresponding to the syntactic neighbors, but also by those words have the same dependent relationship with the target word weighted by their semantic similarity.

Formally we assume a set W of words and a set R of syntactic relations. The word representations then lie in vector space, which is the span of the set of basis vectors $\{\mathbf{e_{r,w'}}|\mathbf{r} \in \mathbf{R}, \mathbf{w'} \in \mathbf{W}\}$. We then define, the contextualized vector of the target with respect to one syntactic dependent as in (4.1) where r_c refers to the syntactic relation of the target w with a word in its context, w_c.

$$v_{r_c,w_c}(w) = \sum_{w' \in W} sim(w_c, w') lin98(r_c, w, w') \mathbf{e_{r_c,w'}} \tag{4.1}$$

The complete contextualization vector is just the sum of the contextualized vectors for all n syntactic dependents of the target comprising context $\mathbf{c}$.

$$\mathbf{v_c}(w) = \sum_{i=1}^{n} v_{r_{c_i},w_{c_i}}(w) \tag{4.2}$$

Then to score the suitably of a candidate word with the target, the authors just take the dot product of their respective contextualized vectors with respect to the context of the target. If there are no syntactic dependents, the semantic similarity between the target and the candidate words, as measured by the cosine measure, is used to do the ranking.

4.3.2 Modification for Lexical Entailment

One of the main issues with the contextualization approach in [Thater et al., 2011] is that it can be expensive to compute as we are summing over all words in the vocabulary. This also is hard to justify as a random pair of words usually have nonzero semantic similarity due to the nature of the word representations, so the end score can be influenced by words that have very little semantic relationship with the context word. This is especially true for relatively common syntactic relations. Thus we created a thesaurus by using the semantic similarity in [Lin, 1998b] which means we took all words in our vocabulary and calculated their semantic similarity with all of the other words in the vocabulary and ordered the scores. Thus we can investigate how many similar words are necessary to use to achieve optimal performance on the lexical substitution task as well as how much performance we are losing if less words are used. The results are shown in Table 4.5.

The other issue with this approach is that it was designed for ranking a word in context and thus the score is not normalized. Thus it is difficult to use in lexical entailment as it is hard to elucidate a threshold that indicates substitutability since the score is dependent on such factors as number of syntactic dependents a target word has. We solve this issue by using the cosine which is normalized.

Lastly, to check if u entails v we need to check not only if u can substitute for v, but we also should score how well v substitutes for u. If both scores are high, then it is likely that the words have the same meaning in their respective contexts. This then leads us to the formula in (4.3) for scoring the lexical entailment between u and v.

$$C - KBDist(u, v) = cos(\mathbf{v_{c_u}}(u), \mathbf{v_{c_u}}(v)) \cdot cos(\mathbf{v_{c_v}}(u), \mathbf{v_{c_v}}(v)) \tag{4.3}$$

4.3.3 Using a Language Model

We also experiment with using a similar idea but with a language model. For simplicity we used a 5-gram model created using the SRILM package [Stolcke, 2002]. While Recurrent neural networks have become state-of-the-art, they are difficult to train and to train. Training an n-gram model can be done on a much larger corpus in a much shorter time making it more convenient and possibly more accuract than an RNN trained on less data.

We trained the language model on the NYT section of Annotated Gigaword [Napoles et al., 2012]. The context in this case was an approximation to the probability of the target word existing in its location in the sentence divided by its prior probability. To add further components to our vector, we measured this value in a modified context, where we swapped one its neighboring words with one that had high semantic similarity with it using [Lin, 1998b]. We then weighted these probabilities by the similarity between the original word and its replacement. We computed the probability of the word given the context divided by its prior using the (4.4) where both probabilities in the numerator can be obtained by the language model. $P_{l_{target}} = p(w_{target}|w_0, ..., w_{target-1})$ is the probability of the word given the previous words

and $P_{r_{target}} = p(w_{target}|w_{target+1}, ..., w_n)$ is the probability of the word given the words to its right. The latter probability can be computed by training the language model backwards. The prior probability wsa obtained by counting the occurrences of the word in the corpus used to train the language model and normalizing by the total number of words.

$$P(w|\mathbf{c}) = \frac{P_l + P_r}{2 \cdot P(w)} \tag{4.4}$$

Thus by modifying the local words in the context, the P_f and P_r change and so we can create a contextualized vector weighting these probabilities similar to the model above. The advantage this model has over the dependency based model is that it can be used when there are few or uninformative syntactic dependents to the target word.

4.4 Experiments

We used the collapsed Stanford dependencies from Annotated GigaWord [Napoles et al., 2012] to create our word representations. These consisted of over 183 million sentences. We lemmatized version of the words and kept only those dependency triples occurring at least 3 times leaving us with about 70 million unique dependency triples.

For the recurrent neural network model, we used the 1600 component vectors obtained from http://www.fit.vutbr.cz/ imikolov/rnnlm/.

We experimented with a wide variety of similarity and entailment measures using the evaluations outlined in Chapter 3. We also evaluated the average time (in seconds) per lexical comparison on the SemEval RTE data, and show the results of the dependency and language model contextualized vectors on [McCarthy and Navigli, 2007]. The performance measure used for the latter evaluation is Generalized Average Precision (GAP) [Kishida, 2005], which ranges from 0.0 to 1.0 where 1.0 indicates all correct items are ranked before all incorrect ones.

4.4.1 WordLists

Algorithm	Accuracy	Coverage
WNSim [Do et al., 2009]	62.7	92.8
Balapincs [Kotlerman et al., 2010]	57.3	99.4
Cosine (PPMI, Dependency)	62.4	99.4
Lin Distributional [Lin, 1998b]	61.8	99.4
RNN [Mikolov et al., 2013c]	57.6	90.7
Resnik [Resnik, 1995]	54.0	92.8
Lin [Lin, 1998a]	56.2	92.8
Paraphrase [Ganitkevitch et al., 2013]	68.2	100
KBDist (Cosine)	69.2	100

Table 4.1: Comparison of algorithms on wordlist in [Kotlerman et al., 2010]. Accuracy was computed using 20 fold cross-validation.

Table 4.1 shows the results of comparing the lexical measures on the entailment wordlist in [Kotlerman et al., 2010]. WordLists for entailment were also used in [**?**] and [Turney and Mohammad, 2013] but they were not used. The reason being, that the one used in [**?**] is derived from WordNet which would give an advantage to entailment methods using that resource and the wordlist in [Turney and Mohammad, 2013] uses the relational definition of entailment. The original wordlist contained 3772 examples, 1068 labeled *entails* and 2704 labeled *does not entail*. It also contained some multi-word expressions. We balanced the dataset so that it contained 880 pairs of both classes and we removed the multi-word phrases. 20 fold cross validation was used where a threshold was tuned on the training set. Coverage refers to the percent of word pairs that the resource was able to score. Only those that were covered were included in the evaluation.

The experiment shows KBDist and Paraphrase well outperform the other methods with KBDist having slightly better results. We used the simple cosine measure to score the likelihood of entailment as it was the strongest distributional method tested.

4.4.2 RTE Alignments

Algorithm	P	R	F1
WNSim [Do et al., 2009]	71.3	60.9	65.6
Balapincs [Kotlerman et al., 2010]	53.4	60.3	56.5
Cosine (PPMI, Dependency)	76.2	76.0	76.1
Lin Distributional [Lin, 1998b]	73.2	64.8	68.8
RNN [Mikolov et al., 2013c]	49.2	66.5	56.6
Resnik [Resnik, 1995]	50.2	38.4	43.5
Lin [Lin, 1998a]	46.4	37.0	41.2
Paraphrase [Ganitkevitch et al., 2013]	75.6	78.4	77.1
KBDist (Cosine)	78.2	77.7	78.0
C-KBDist (200)	84.5	77.9	81.1

Table 4.2: Comparison of algorithms on RTE alignment dataset. Precision, recall and F1 were computed by 20 fold cross validation.

Table 4.2 shows the results on our substitutable lexical entailment dataset constructed from the RTE2 Alignment corpus [Brockett, 2007]. 20 fold cross validation was used where a threshold was tuned on the held out set. The results show that once again, Paraphrase and KBDist have the best performance, with KBDist a slight edge. Incorporating context using C-KBDist gives a significant improvement in precision and overall F_1. In this experiment, 200 of the most similar words to a given context word were used in creating the contextualized vector.

4.4.3 RTE LLM

Algorithm	P	R	F1
WNSim [Do et al., 2009]	46.8	81.1	59.3
Balapincs [Kotlerman et al., 2010]	NA	NA	NA
Cosine (PPMI, Dependency)	46.2	84.3	59.7
Lin Distributional [Lin, 1998b]	46.1	84.3	59.6
RNN [Mikolov et al., 2013c]	45.9	83.7	59.3
Resnik [Resnik, 1995]	47.2	77.5	58.7
Lin [Lin, 1998a]	49.5	71.5	58.5
Paraphrase [Ganitkevitch et al., 2013]	64.6	53.5	58.6
KBDist (Cosine)	53.1	81.1	64.1
C-KBDist (200)	40.6	80.5	58.7

Table 4.3: Comparison of Algorithms on RTE LLM. Precision, recall and F1 were computed on the 9500 examples in the train and test set. Parameters were turned on the dev set which consisted of 500 examples.

Table 4.3 shows the results of using LLM as described in Chapter 3. We used equation (3.3). The results again show KBDist giving the best results by a significant margin. This is particularly impressive as the RTE task is very difficult and even small improvements are seen as important, not to mention large improvements, as seen in this table. BalAPincs wasn't evaluated due to its very slow speed. This is discussed further in the next section.

4.4.4 Speed

Algorithm	Time per Comparison (ms)	Time per Example (ms)
WNSim [Do et al., 2009]	10	1269
Balapincs [Kotlerman et al., 2010]	2687	194828
Cosine (PPMI, Dependency)	15	1881
Lin Distributional [Lin, 1998b]	6	705
RNN [Mikolov et al., 2013c]	0.2	29
Resnik [Resnik, 1995]	4	547
Lin [Lin, 1998a]	5	602
Paraphrase [Ganitkevitch et al., 2013]	0.002	0.3
KBDist (Cosine)	2	313
C-KBDist (200)	25	3244

Table 4.4: Average time (in milliseconds) per lexical entailment comparison and example on SemEval 2014 RTE data using LLM.

Table 4.4 shows the speed of the various entailment measures. We measured the average time per word comparison as well as the average time for evaluating (3.3) on an example from the dataset. As can be seen, KBDist , is at least several times faster than

any other distributional method. This is because it relies on accessing a knowledge base, which can be done quickly, to avoid the costly cosine evaluation.

BalAPincs is very expensive to compute due to the quadratic complexity, that combined with its weak performance in the evaluations make it not a very useful lexical entailment measure. These experiments show that including a method that incorporates context, like contextualized vectors, is not prohibitively expensive.

4.4.5 Similarity in Context

Algorithm	GAP
Random	29.3
Contextualized Vector (Dependency) (0)	46.7
Contextualized Vector (Dependency) (1)	47.0
Contextualized Vector (Dependency) (10)	47.6
Contextualized Vector (Dependency) (100)	49.6
Contextualized Vector (Dependency) (1000)	51.9
Contextualized Vector (Dependency) (2000)	51.8
Contextualized Vector (LM) (0)	44.4
[Thater et al., 2011]	51.7
[Dinu and Lapata, 2010] (LDA/ NMF)	42.9
[Séaghdha, 2010] (LDA)	49.5

Table 4.5: GAP on test set in [McCarthy and Navigli, 2007].

Table 4.5 compares our modified contextualized vector approach with others in the literature. The models with *Dependency* in parenthesis used dependency features and the models with *LM* in parenthesis used the language model features. The table shows how the performance scales as the number of word replacements increases (the number in parentheses). The results show that there is an optimum for this parameter which was expected as there are only so many words that have a meaningful semantic relationship with a given target word and only substituting these adds an additional useful component to the contextualized vector.

Notice also how those methods that rely on much more sophisticated approaches-like LDA, which also have scaling issues, do not do as well.

4.5 Future Work

One main avenue of future work is the extension to multi-word phrases. Since one contribution of this work showed how the lexical paraphrases from the PPDB are useful in lexical entailment, it seems likely that the multi-word phrases would be as well. WordNet also includes multi-word phrases and so both PPDB and WordNet can provide the knowledge-base backbone for this extension just as they did in this work.

Another avenue of future work is to improve the performance when using local context. One issue with the current model is that there is an assumption about the functional form that the components of the contextualized vector should have. Namely the product of Lin98 and the semantic similarity of the replacement words, or in the

case of the language model, the product of the probability of the target divided by its prior and the semantic similarity of the replacement words. This is almost surely a non-optimal functional form as there should be a step function in regards to the semantic similarity since even two random words have a significant score compared to even two synonyms, weakening the effect of strong replacement words. To alleviate this and also at the same time provide a way to combine the two models discussed in this thesis and allow for more flexibility to include additional information such as the specificity of the word, a nonlinear function over these contextualized features should be learned. A simple neural network could be used for this purpose and the model could be trained to optimize pairwise rankings on the training data where we can use a hinge-loss on the dot product of the contextualized vectors. The model would have a small parameter space and so lack of training data shouldn't be a problem and this approach should give a significant gain in performance.

Lastly, with these improvements in lexical entailment and eventually phrasal entailment, it is important to consider new RTE models that can make use of these new tools to further push performance in the RTE task.

References

[Agirre et al., 2009] 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 *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 19–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Banerjee and Pedersen, 2002] Banerjee, S. and Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145, Mexico City, USA.

[Baroni et al., 2012] Baroni, M., Bernardi, R., Do, N.-Q., and Shan, C.-c. (2012). Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 23–32, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Bejar et al., 1991] Bejar, I. I., Chaffin, R., and Embretson, S. (1991). A taxonomy of semantic relations. In *Cognitive and psychometric analysis of analogical problem solving*, pages 55–91. Springer.

[Bengio et al., 2003] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *JOURNAL OF MACHINE LEARNING RESEARCH*, 3:1137–1155.

[Brockett, 2007] Brockett, C. (2007). Aligning the rte 2006 corpus. *Microsoft Research*.

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

[Chang et al.,] Chang, K.-W., Yih, W.-t., and Meek, C. Multi-relational latent semantic analysis. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*.

[Chang et al., 2010] Chang, M.-W., Goldwasser, D., Roth, D., and Srikumar, V. (2010). Discriminative learning over constrained latent representations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 429–437. Association for Computational Linguistics.

[Collobert and Weston, 2008] Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning.

[Curran, 2003] Curran, J. (2003). *From Distributional to Semantic Similarity*. PhD thesis, University of Edinburgh.

[Dagan et al., 2006] Dagan, I., Glickman, O., Gliozzo, A., Marmorshtein, E., and Strapparava, C. (2006). Direct word sense matching for lexical substitution. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 449–456, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Dinu and Lapata, 2010] Dinu, G. and Lapata, M. (2010). Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1162–1172. Association for Computational Linguistics.

[Do et al., 2009] Do, Q., Roth, D., Sammons, M., Tu, Y., and Vydiswaran, V. (2009). Robust, light-weight approaches to compute lexical similarity. Technical report.

[Fellbaum, 1998] Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

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

[Ganitkevitch et al., 2013] Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

[Geffet, 2005] Geffet, M. (2005). The distributional inclusion hypotheses and lexical entailment. In *In Proceedings of ACL-2005. Ann Arbor*, pages 107–114.

[Harris, 1954] Harris, Z. S. (1954). Distributional structure. *Word*.

[Hindle, 1990] Hindle, D. (1990). Noun classification from predicate-argument structures. In *Proceedings of the 28th annual meeting on Association for Computational Linguistics*, pages 268–275. Association for Computational Linguistics.

[Hirst and St-Onge, 1998] Hirst, G. and St-Onge, D. (1998). Lexical chains as representations of context for the detection and correction of malapropisms. In Fellbaum, C., editor, *WordNet: An electronic lexical database*, pages 305–332. MIT Press.

[Jiang and Conrath, 1997] Jiang, J. and Conrath, D. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International conference on Research in Computational Linguistics*, pages 19–33.

[Kishida, 2005] Kishida, K. (2005). *Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments*. National Institute of Informatics Tokyo, Japan.

[Kotlerman et al., 2010] Kotlerman, L., Dagan, I., Szpektor, I., and Zhitomirsky-geffet, M. (2010). Directional distributional similarity for lexical inference. *Nat. Lang. Eng.*, 16(4):359–389.

[Landauer and Dutnais, 1997] Landauer, T. K. and Dutnais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240.

[Leacock and Chodorow, 1998] Leacock, C. and Chodorow, M. (1998). Combining local context and wordnet similarity for word sense identification. In Fellbaum, C., editor, *WordNet: An Electronic Lexical Database*, pages 265–283. The MIT Press.

[Lin, 1998a] Lin, D. (1998a). Automatic retrieval and clustering of similar words. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 768–774.

[Lin, 1998b] Lin, D. (1998b). An information-theoretic definition of similarity. In *Proceedings of the International Conference on Machine Learning (ICML)*.

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

[Mihalcea et al., 2006] Mihalcea, R., Corley, C., and Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, Boston, USA.

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

[Mikolov et al., 2010] Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In Kobayashi, T., Hirose, K., and Nakamura, S., editors, *INTERSPEECH*, pages 1045–1048. ISCA.

[Mikolov et al., 2013b] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

[Mikolov et al., 2013c] Mikolov, T., Yih, W.-t., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

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

[Mnih and Hinton, 2008] Mnih, A. and Hinton, G. (2008). A scalable hierarchical distributed language model. In *In NIPS*.

[Napoles et al., 2012] Napoles, C., Gormley, M., and Van Durme, B. (2012). Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 95–100. Association for Computational Linguistics.

[Navigli et al., 2007] Navigli, R., Litkowski, K. C., and Hargraves, O. (2007). Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 30–35. Association for Computational Linguistics.

[Resnik, 1995] Resnik, P. (1995). Disambiguating noun groupings with respect to wordnet senses. In *Proceedings of the Third Annual Workshop on Very Large Corpora*.

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

[Séaghdha, 2010] Séaghdha, D. O. (2010). Latent variable models of selectional preference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 435–444. Association for Computational Linguistics.

[Stolcke, 2002] Stolcke, A. (2002). Srilm - an extensible language modeling toolkit. pages 901–904.

[Thater et al., 2011] Thater, S., Fürstenau, H., and Pinkal, M. (2011). Word meaning in context: A simple and effective vector model. In *IJCNLP*, pages 1134–1143.

[Turney and Mohammad, 2013] Turney, P. D. and Mohammad, S. M. (2013). Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, pages 1–40.

[Wu and Palmer, 1994] Wu, Z. and Palmer, M. (1994). Verb semantics and lexical selection. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 133–138.

[Yih et al., 2012] Yih, W.-t., Zweig, G., and Platt, J. C. (2012). Polarity inducing latent semantic analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1212–1222, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Zhitomirsky-Geffet, 2009] Zhitomirsky-Geffet (2009). Bootstrapping distributional feature vector quality. *Comput. Linguist.*, 35(3):435–461.