``Distributional differences between antonyms and synonyms

Era Choshen (era.choshen@mail.huji.ac.il)

Supervisor: Dr. Omri Abend

1 Introduction

Distinguishing between antonyms and synonyms is important for NLP[1] applications, mainly for textual entailment. The ability to discern between antonymy and synonymy while analyzing textual entailment has applications in larger NLP systems, such as question answering, document summarization and of course machine translation.

The field of distributional semantics currently favors the coupling of the distributional hypothesis with vector space modeling, using the combination of the two as its framework of choice. The distributional hypothesis, which is based on the work Firth and Harris in the 1950's, can be succinctly described by the idea that words that occur in similar contexts tend to have similar meanings, while vector space modeling (VSM) is the idea of representing words in a document by embedding them in a multi-dimensional vector space. The terms context and meaning according to the distributional hypothesis are open to interpretation, leading to different implementations, each with its own advantages and disadvantages.

The common approach in NLP is to represent words as high-dimensional vectors, with each coordinate containing distributional information. Word similarity is then interpreted as some measure of distance between the vectors representing the words. Unfortunately, most VSM models' word representations tend to capture association rather than similarity. In 2014 Hill et al. demonstrated this by showing that most VSM models perform well in terms of Spearman's ρ on human judgement of word association, while in contrast the same models do not perform as well on data sets containing human judgments of similarity. In this paper, my aim is to attempt to verify the ability to differentiate between antonyms and synonyms independently of specific word structures and large language resources.[2]

I also hope that this paper can be a stepping stone in showing that there does exist a distributional difference between synonyms and antonyms, which can be represented in VSM models, given the right encoding and processing.

[1] Natural language processing

[2] Motivation for this appears in a subsequent section.

2 Related work

Work on VSMs for predicting lexical similarity dates back to the 1970's. Until recently the common approach was to represent each word *w* as a vector, with each coordinate representing the co-occurrence of the word *w* in the context of another word *v*. These co-occurrence vectors are then post-processed using dimensionality reduction techniques or LSA[3]. Similarity measures such as cosine similarity or PPMI[4] are then calculated using these vectors. Until lately, these models encoded the co-occurrence statistics directly into the vectors with little consideration as to the semantic relationship between the words – the *bag of words* approach.

More work has been done to embellish this approach by encoding the distributional information into an objective within a language model, using neural networks to learn these word representations. A good example of this approach is the popular word2vec model[5] (Mikolov et al, 2013). This model employs a two-layer neural network that takes a corpus as input and outputs a vector space in which each unique word is a vector. One architectural implementation of the word2vec model is the continuous-bag-of-words (CBOW) architecture. CBOW predicts a word given its past and future context, yielding the following objective function:

$$max \sum_{t=1}^{T} \log p(w_t | w_{t-c}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+c})$$

Another good example of these types of models is the GloVe model. The main intuition behind this model is that the ratio of word co-occurrence probabilities encodes meaning. The training objective of GloVe is to learn a word vector embedding such that the dot product of two vectors approximates the aforementioned ratio, thus encoding the relationship between different words into the embedding.

All of these models perform well in general, especially in analogy tasks (e.g Paris – France + Italy = Rome), however Hill et al., the authors of the SimLex999 dataset, show by using their dataset that most of these models tend to capture association rather than similarity (Hill et al, 2014).

This downside to VSM has been noted and several attempts to address it exist in the current literature. Several models have been proposed using patterns, WordNet and thesauri (Lin et al., 2003; Turney, 2008; Wang et al., 2010; Mohammad et al., 2013; Schulte im Walde and Koper, 2013; Roth and Schulte im Walde, 2014, Schwartz et al., 2015). These models tend to either employ certain patterns indicative of word relations (e.g from X to Y) in order to induce certain word embeddings, or to use a seed set of antonyms and infer via indirect connections whether two words belong in contrasting relational sets. Other works have tried to assign dissimilar

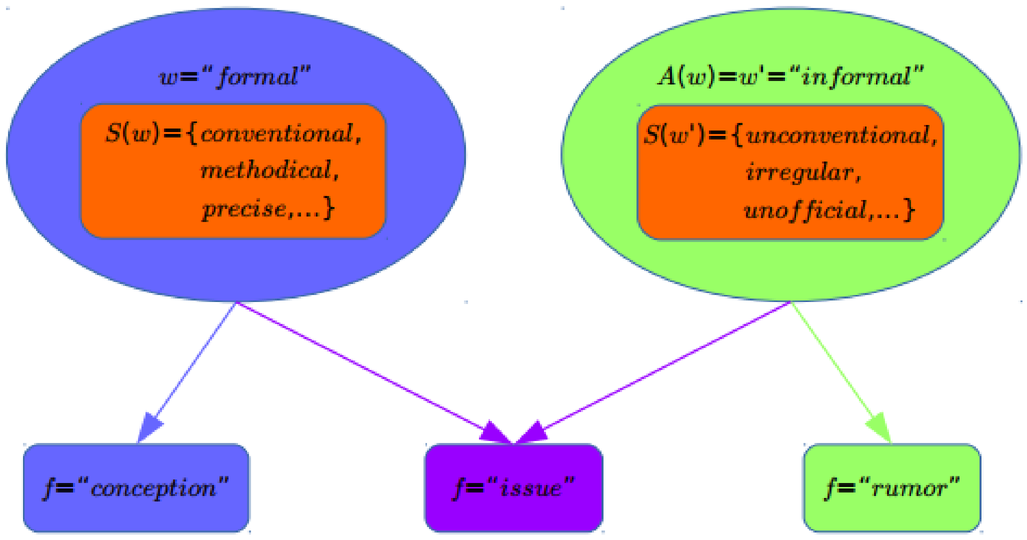
[3] Latent semantic analysis

[4] Positive pointwise mutual information

[5] This is actually a group of models

vectors to antonyms, relying once again on thesauri and WordNet (Yih et al., 2012; Chang et al., 2013; Ono et al., 2015).

Another work dealing with lexical contrast was published by several researchers at Stuttgart University. I present this work separately because the intuition behind their model is similar to my theoretical approach. The idea behind their model is that "the strongest features of a word also tend to represent strong features of its synonyms, but weaker features of its antonyms" (Nguyen et al, 2016). Following is a figure from their paper which provides a schematic visualization of the logic guiding their model:



w="formal"
$S(w)$={conventional, methodical, precise,...}
$A(w)=w'$="informal"
$S(w')$={unconventional, irregular, unofficial,...}
f="conception"
f="issue"
f="rumor"

Figure 1. Illustration of the $weight^{SA}$ scores for the adjective target formal. ". (Nguyen et al., 2016)

Nguyen et al. presented a novel approach to vector representation (the $weight^{SA}$ scheme shown above), incorporating this representation later on into the skip-gram model. Although different from my approach, the underlying logic shares many similarities. This idea is in essence the theoretical backing to my idea that distributional differences can be found between antonym and synonym pairs.

3 Motivation for structure and data independent similarity prediction

In this section, I will outline the motivation for a similarity judgement model that is independent of large sources and specific structures within the language of choice, in this case English.

Many of the works in the current literature that deal with antonym-synonym similarity rely on specific structures within the language or huge databases such as WordNet or other thesauri in order to mitigate the induced similarity between antonyms and synonyms in the different VSM models.[6] While these works are of

[6] A brief outline of these various approaches appears in the next section.

great importance in advancing the field, they create a certain dependency on specific attributes and previous works in the language. For this reason I decided that the framework within which I would like to work in this paper should be devoid of dependencies on specific structures within the English language as well as large databases of information linking words within the language, such as thesauri.

3.1 Dependencies on specific language structures

Recent works, in large part based on Lin et al.'s work, use patterns which were found to be useful in identifying word relations such as antonymy (Lin et al., 2003). The problem with these methods is that they are very language specific. The fact that a certain pattern is indicative of a word relation does not necessarily imply that a similar pattern will be easy to find, or relevant, in a different language. An additional problem is that two words, if they are not very common, will not necessarily appear alongside each other in these patterns, even if the two are antonyms (or synonyms). In my opinion, there is more inferential potential in an embedding or similarity calculation that is independent of such patterns.

3.2 Dependencies on thesauri and other large databases

Another approach, popularized by Saif Mohammad, relies on an antonym seed set. This approach begins with a large set of antonyms gathered from some source (for example WordNet, or some thesaurus), and uses another source (another thesaurus for example) in order to check whether any synonyms of the words are in contrasting categories and therefore antonyms. Although this approach has promise, it also requires gathering antonym seeds as well as having a rather thorough WordNet-like source. Unfortunately a WordNet-like source is not always available; this is problematic as one of the goals of antonym detection is to enable machine translation, even to resource poor languages. Perhaps it would therefore be wiser not to rely so heavily on resources such as WordNet in order to calculate word similarity. Another potential problem is a certain circularity, for example in Nguyen et al. work from 2016 they define the weights of a target word w and a feature f:

$$weight^{SA}(w,f) = \frac{1}{\#(w,u)} \sum_{u \in W(f) \cap S(w)} sim(w,u) - \frac{1}{\#(w',v)} \sum_{w' \in A(w)} \sum_{v \in W(f) \cap S(w')} sim(w',v)$$

It is important to notice that this requires knowledge of $A(w)$, meaning the set of antonyms of w. While this makes sense for a "training set" of antonyms, knowledge of this set for every target word renders the problem irrelevant, seeing as the problem is reduced to simply checking whether the word we check against is in the set. A similar problem was addressed in Yih et al.'s paper in 2013 using a nearest-neighbor approach, which allows out-of-database embedding. However, the problem there was slightly different and less reliant on the weighting scheme as in Nguyen et al.'s article.

4 Theoretical framework and approach

I chose to work under the assumption that satellite features have the potential to be indicative for certain synonym sets. Thus for example, if we consider the following contrasting sets $A = \{ocean, sea, ogin\}, B = \{land\}$, it seems natural that the feature "blue" would be much more prominent for set A than set B. This assumption can also be seen in the GloVe model, in the model overview. A similar assumption is the underlying intuition behind Nguyen et al.'s model mentioned earlier.

I hypothesized that antonyms and synonyms could be discerned based on their distributions, for example, based on the number of highly co-occurring features they shared. Either way I did not want to assume what the right calculation should be, but rather I chose to assume that there exists some function $f: \mathbb{R}^n \times \mathbb{R}^N \to \{0,1\}$, that receives two vector embeddings as input and outputs whether the represented words are synonyms or antonyms.

I chose to use a feed-forward neural network that would perform the binary classification task, with one option being "antonym" and the other option being "synonym." I presumed that if I could design a neural network that would perform this classification task well, I could then in fact combine it with other existing models that perform well in general (such as GloVe or word2vec) in order to improve their performance on a data set that focuses on similarity, such as SimLex999.

5 Experiment Infrastructure

All the datasets, neural networks and code used to create and evaluate them has been made public in order to replicate the findings if necessary:

https://github.com/ec2604/Antonym-Detection.

5.1 Datasets

5.1.1 Evaluation Dataset

In order to evaluate the results I used the SimLex999 dataset (Hill et al., 2014), which consists of 999 pairs of words. As mentioned earlier in this paper, each pair in this dataset has a similarity ranking which is an average of 50 human judgements.

5.1.2 Training Datasets

At first I created my own synonym-antonym word pair list using the NLTK python package to parse the wacky wiki corpus[7]. I refined this list very carefully, making sure every pair met my personal standard for antonymy or synonymy. Unfortunately this list consisted of only ~7000 pairs of words. While training the neural networks I designed to solve the antonymy classification problem, I slowly realized that 7000 examples for a problem of this type is relatively small.

Due to the possibly problematic size of my manually compiled dataset I decided to try using word pairs that had been previously used in published works. I

[7]http://wacky.sslmit.unibo.it/doku.php?id=download

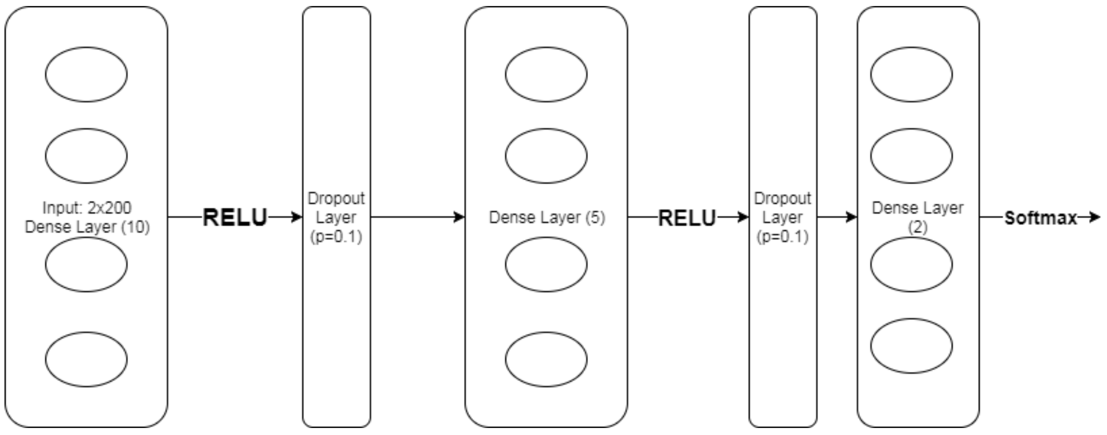
used the synonym-antonym word pairs used by Ono et al. (2015)[8], which were in turn mined from two thesauri WordNet (Miller, 1995) and Roget (Kipfer, 2009) and were provided by Zhang et al. (2014). This new word pair set resulted in a dataset consisting of 723,240 word pairs when using the GloVE model (Pennington et al., 2014). Although a seemingly vast improvement, this dataset caused me to question the validity of the pairs used to create it, an issue that I will elaborate on lately, and which also led me to stop using it.

I created several training datasets. At first I wanted to evaluate the raw co-occurrence vectors, so I created them manually by parsing the 1 GB wacky wiki corpus. As I began designing the neural networks meant to perform the classification task, I realized that due to the high dimensionality of these vectors, using them without additional preprocessing would be futile.

Following my realization concerning the dimensionality problem of raw co-occurrence vectors, I decided to use the pre-trained GloVe[9], word2vec (CBOW)[10] and word2vec (skipgram – fastText)[11] models. I took the word pairs I assembled beforehand and stacked each model's embeddings together to be used as inputs for the neural networks I had designed.

5.2 Neural network for differentiating between word embeddings

I chose to use a simple feed-forward network architecture to deal with the antonym-synonym classification problem:



The output of the network is [x y], where [1 0] signifies an antonymy relation and [0 1] signifies a synonymy relation.

[8] https://github.com/tticoin/AntonymDetection/tree/master/data/dict
[9] Based on Wikipedia 2014 + Gigaword 5. http://nlp.stanford.edu/data/glove.6B.zip
[10] Based on the Google News dataset. https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?usp=sharing
[11] https://s3-us-west-1.amazonaws.com/fasttext-vectors/wiki.en.zip

5.3 Evaluation

In order to evaluate the model I looked at two primary results. Firstly, I looked at the neural network's performance on the validation set. Secondly, I tried to compare the model's performance to results in the current literature. Most VSM literature evaluates the performance of a model by evaluating the Spearman ρ between the similarity scores given by the model to the Simlex999 dataset and comparing it to the Spearman's ρ of other models.

6 Experiments

In the following sections I will present similar evaluations for three different models. The first part of every model evaluation will show the performance of the neural network in terms of model accuracy (training and test). The second part will compare the VSM's Spearman's ρ on the Simlex999 dataset to other models, and then show the results of an OLS model combination between the pre-existing model and my neural network classification. I chose to quantify the quality of each model combination in accordance with similar literature – the Spearman's ρusing 10-fold cross-validation.

Summary of the results:

Model	***Baseline Spearman's ρ***	***Model combination – Spearman's ρ***
GloVe	0.3402	0.3753, 0.3391[12]
Word2Vec (CBOW)	0.4419	0.443
Word2Vec (fastText)	0.382	0.386

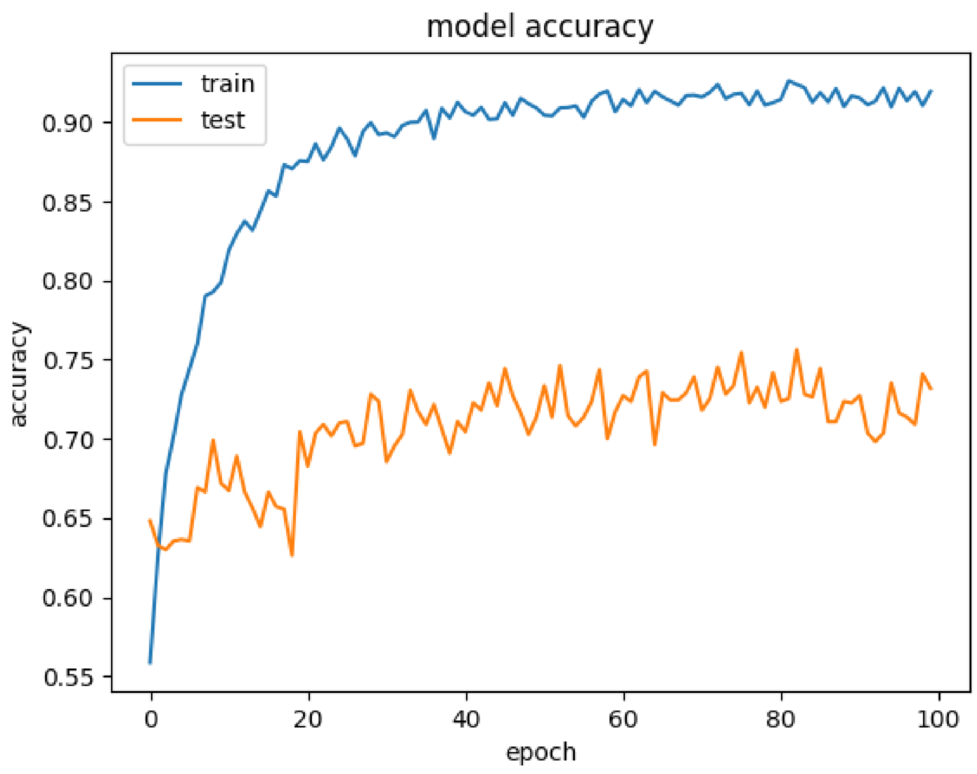
6.1 GloVe Model Results

6.1.1 Neural network results

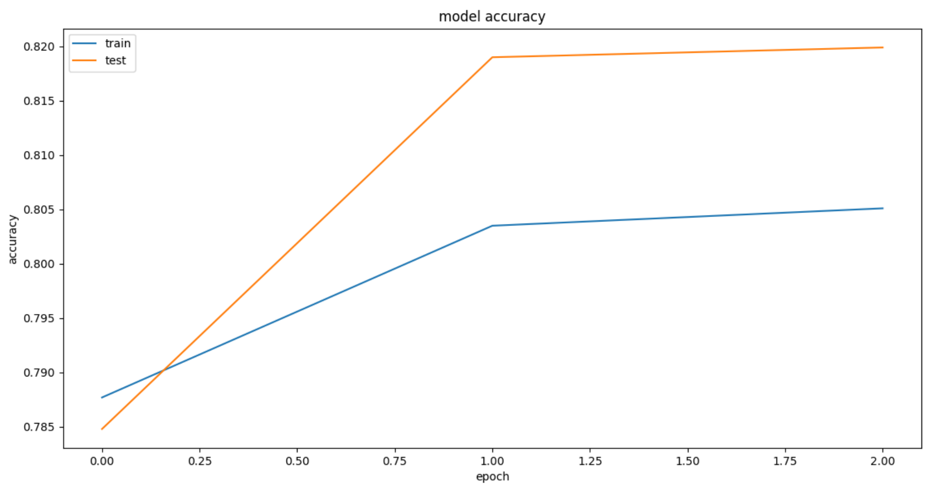
6.1.1.1 Neural network results for manually compiled word-pairs

The accuracy metrics for the validation set was in the low 70's:

[12] First score used the manually compiled word pairs, the second uses Zhang's word pairs.



6.1.1.2 Neural network results for Zhang et al.'s synonym-antonym word pairs



One notices two surprising things here – first off, the validation set accuracy surpasses that of the training set. Moreover, the validation set accuracy is surprisingly high (82%). These two things could of course be due to chance and the sheer number of examples, however a more detailed analysis reveals that the word-pairs themselves could be skewed. If one takes a close look at the word-pair sets, some of the pairs would hardly be categorized as synonyms or antonyms by a native speaker (e.g provocation-motivation, utterance-assertion, ask-ignore, topping-cheap). Although these combinations create a very large number of examples, they erode the quality of the dataset in my opinion – creating a false sense of accuracy (this would be interesting to check in the original article as well).

Due to the questionability of the word-pairs, which I mentioned earlier in the paper, I decided to not use the datasets generated by these word pairs. All further experiments made use of the word-pairs I manually compiled using the NLTK WordNet package.

6.1.2 GloVe model combination – Spearman's ρ

As a first step I ran the GloVE model on the Simlex999 dataset. The result in terms of Spearman's ρ was 0.3402, a result very similar to that of Schwartz et al.'s (2015) replication (0.35). After applying the OLS method to create a linear combination of the neural network model's prediction alongside that of the GloVe model, I received the following results:

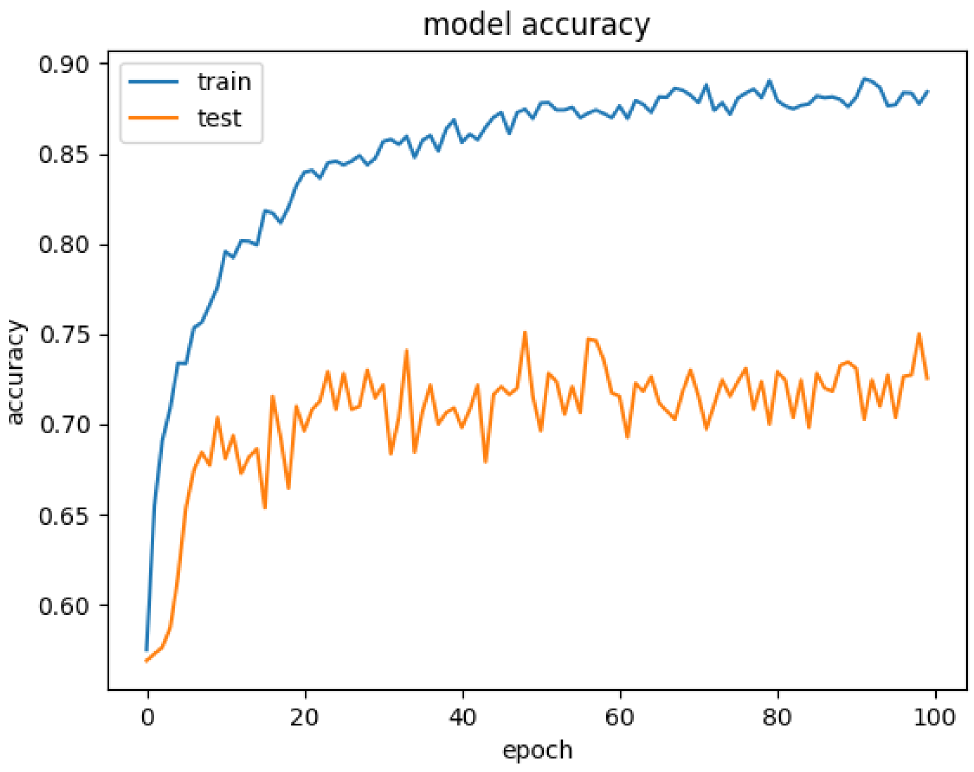
Average 10-fold cross-validated Spearman's ρ for the manually compiled word pairs: 0.3753 (10.3% increase!)

Average 10-fold cross-validated Spearman's ρ for the Zhang et al.'s word pairs: 0.3391 (A decrease)

6.2 Word2Vec (CBOW) Model Results

6.2.1 Neural Network results

The accuracy for the validation set was in the mid 70's:



Clearly the training set accuracy is far superior to that of the validation set. This is probably due to the fact that the data set created by the manual collection of word pairs was relatively small, and to the fact that the number of epochs was relatively large, (100). Nevertheless, this accuracy rate for the validation set seems relatively good, although it is hard to compare to other works in the literature.

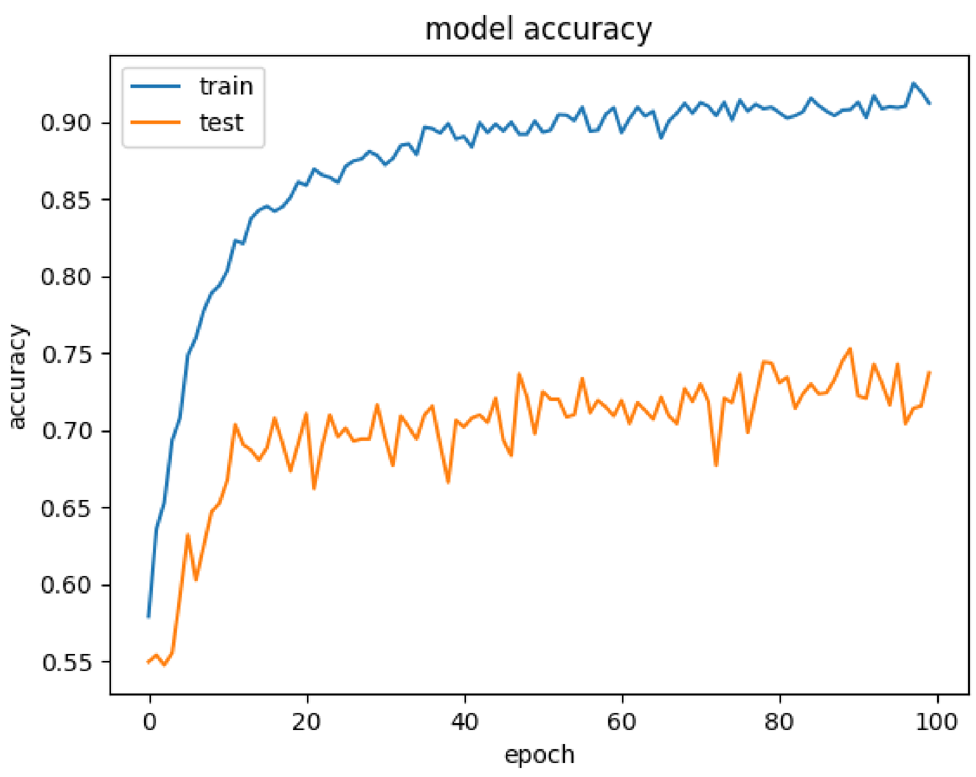
6.2.2 Word2Vec model combination - Spearman's ρ

First I ran the word2vec model by itself on the Simlex999 dataset in order to determine the baseline Spearman's ρ. The result was 0.4419, a result that approximately matches up with the model's results when replicated by Schwartz et al (2015).

After calculating the baseline for the word2vec model, I tried combining it with my neural network classification model. I performed 10-fold cross validation on the Simlex999 dataset and calculated the word2vec cosine similarity as well as the neural network's prediction (after softmax activation) for the training set, and tried to fit them together as a linear combination using the OLS approach. The average Spearman's ρ after 10-fold cross validation was 0.443.

6.3 Skipgram (FastText) Model Results

6.3.1 Neural network results



We can see that the neural network's results are rather similar to the other models' performance.

6.3.2 Skipgram (fastText) model combination - Spearman's ρ

When evaluating the cosine similarity of the fastText model on the Simlex999 dataset, the result in terms of Spearman's ρ was 0.382. Unfortunately, this was not very similar to Schwartz et al.'s (2015) results (0.462) on Simlex999, probably due to the fact that they trained the word2vec skipgram model themselves on an 8G corpus.

When combining the fastText model with the neural network's predictions, the result of 10-fold cross-validated Spearman's ρ was 0.386, a 1% increase.

7 Conclusions

I originally set out to verify whether there exists a semantic distributional difference between pairs of antonyms and synonyms. My original intention was to use raw co-occurrence vectors in order to train a simple feed-forward neural network that would perform binary classification such that the network could discern between antonyms and synonyms. Due to practical considerations (the dimensions of such vectors are enormous and force the network to learn an impossibly large number of parameters), I switched my approach to the use of pre-trained word embedding models. It is important to note that these models all encode co-occurrence statistics in the vectors' features – as such, I feel that my original intention has still been achieved to some extent.

During the evaluation of the networks, I was able to achieve a satisfying 70-80% accuracy on the validation sets used during the training of the neural networks. Furthermore, when attempting to put the different networks into context with other models, I succeeded in achieving an increase of up to 10% in the Spearman's ρ measure by using a linear combination of the original VSM and my neural network output. It is important to note that I adhered to my desire to create the neural network models independently of word structure and large datasets of linguistic data (such as thesauri). No word structures, such as patterns, were used, and furthermore no thesauri are necessary in order to predict the accuracy of new words – all that is necessary is their original VSM embedding.[13]

I believe that this paper has shown two important things. Firstly, even with a small dataset that enforces a rather strict interpretation of synonymy and antonymy, a neural network can use embeddings based on distributional semantics in order to discern between the two relations. Secondly, the results of such a network can be combined with existing state-of-the-art models in order to improve their results on similarity datasets such as Simlex999. Although the improvement to the models is not exceptional (except for GloVe), it is important to note that the main purpose of this paper was to show that an above-chance classification of lexical contrast is possible using distributional semantics – and as such, most of the effort went into creating the datasets and tuning the neural networks properly.

8 References

Baroni, Marco, et al. "The WaCky wide web: a collection of very large linguistically processed web-crawled corpora." *Language resources and evaluation* 43.3 (2009): 209-226.

Bird, Steven, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

Bojanowski, Piotr, et al. "Enriching word vectors with subword information." *arXiv preprint arXiv:1607.04606* (2016).

Chang, Kai-Wei, Wen-tau Yih, and Christopher Meek. "Multi-Relational Latent Semantic Analysis." *EMNLP*. 2013.

Fellbaum, C. "About WordNet. WordNet. Princeton University." (2010).

Firth, John Rupert. "Modes of Meaning. Papers in Linguistics 1934-51, 190–215." (1957).

Harris, Zellig S. "Distributional structure." *Word* 10.2-3 (1954): 146-162.

Hill, Felix, Roi Reichart, and Anna Korhonen. "Simlex-999: Evaluating semantic models with (genuine) similarity estimation." *Computational Linguistics* (2016).

[13] These VSM's can be trained on new corpora.

Lin, Dekang, et al. "Identifying synonyms among distributionally similar words." *IJCAI*. Vol. 3. 2003.Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

Mohammad, Saif M., et al. "Computing lexical contrast." *Computational Linguistics* 39.3 (2013): 555-590.

Nguyen, Kim Anh, Sabine Schulte im Walde, and Ngoc Thang Vu. "Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction." *arXiv preprint arXiv:1605.07766*(2016).

Ono, Masataka, Makoto Miwa, and Yutaka Sasaki. "Word Embedding-based Antonym Detection using Thesauri and Distributional Information." *HLT-NAACL*. 2015.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014.

Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *Journal of Machine Learning Research* 12.Oct (2011): 2825-2830.

Rehurek, Radim, and Petr Sojka. "Software framework for topic modelling with large corpora." *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. 2010.

Roth, Michael, and Sabine Schulte Im Walde. "Combining Word Patterns and Discourse Markers for Paradigmatic Relation Classification." *ACL (2)*. 2014.

Im Walde, Sabine Schulte, and Maximilian Köper. "Pattern-based distinction of paradigmatic relations for German nouns, verbs, adjectives." *Language Processing and Knowledge in the Web*. Springer, Berlin, Heidelberg, 2013. 184-198.

Turney, Peter D. "A uniform approach to analogies, synonyms, antonyms, and associations." *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2008.

Wang, Wenbo, et al. "Pattern-based synonym and antonym extraction." *Proceedings of the 48th Annual Southeast Regional Conference*. ACM, 2010.

Yih, Wen-tau, Geoffrey Zweig, and John C. Platt. "Polarity inducing latent semantic analysis." *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 2012.

Zhang, Jingwei, et al. "Word Semantic Representations using Bayesian Probabilistic Tensor Factorization." *EMNLP*. 2014.