Symmetric Pattern Based Word Embeddings for Improved Word Similarity Prediction

Roy Schwartz,[1] **Roi Reichart,**[2] **Ari Rappoport**[1]
[1]Institute of Computer Science, The Hebrew University
[2]Technion, IIT
{roys02|arir}@cs.huji.ac.il roiri@ie.technion.ac.il

Abstract

We present a novel word level vector representation based on *symmetric patterns (SPs)*. For this aim we automatically acquire SPs (e.g., "X *and* Y") from a large corpus of plain text, and generate vectors where each coordinate represents the co-occurrence in SPs of the represented word with another word of the vocabulary. Our representation has three advantages over existing alternatives: First, being based on symmetric word relationships, it is highly suitable for word similarity prediction. Particularly, on the SimLex999 word similarity dataset, our model achieves a Spearman's ρ score of 0.517, compared to 0.462 of the state-of-the-art word2vec model. Interestingly, our model performs exceptionally well on verbs, outperforming state-of-the-art baselines by 20.2–41.5%. Second, pattern features can be adapted to the needs of a target NLP application. For example, we show that we can easily control whether the embeddings derived from SPs deem antonym pairs (e.g. (*big,small*)) as similar or dissimilar, an important distinction for tasks such as word classification and sentiment analysis. Finally, we show that a simple combination of the word similarity scores generated by our method and by word2vec results in a superior predictive power over that of each individual model, scoring as high as 0.563 in Spearman's ρ on SimLex999. This emphasizes the differences between the signals captured by each of the models.

1 Introduction

In the last decade, vector space modeling *(VSM)* for word representation (a.k.a word embedding), has become a key tool in NLP. Most approaches to word representation follow the distributional hypothesis (Harris, 1954), which states that words that co-occur in similar contexts are likely to have similar meanings.

VSMs differ in the way they exploit word co-occurrence statistics. Earlier works (see (Turney et al., 2010)) encode this information directly in the features of the word vector representation. More Recently, Neural Networks have become prominent in word representation learning (Bengio et al., 2003; Collobert and Weston, 2008; Collobert et al., 2011; Mikolov et al., 2013a; Pennington et al., 2014, inter alia). Most of these models aim to learn word vectors that maximize a language model objective, thus capturing the tendencies of the represented words to co-occur in the training corpus. VSM approaches have resulted in highly useful word embeddings, obtaining high quality results on various semantic tasks (Baroni et al., 2014).

Interestingly, the impressive results of these models are achieved despite the shallow linguistic information most of them consider, which is limited to the tendency of words to co-occur together in a pre-specified context window. Particularly, very little information is encoded about the syntactic and semantic relations between the participating words, and, instead, a bag-of-words approach is taken.[1]

This bag-of-words approach, however, comes with a cost. As recently shown by Hill et al. (2014), despite the impressive results VSMs that take this approach obtain on modeling word *association*, they are much less successful in modeling word *similarity*. Indeed, when evaluating these VSMs with datasets such as wordsim353 (Finkelstein et al., 2001), where the word pair scores re-

[1]A few recent VSMs go beyond the bag-of-words assumption and consider deeper linguistic information in word representation. We address this line of work in Section 2.

flect association rather than similarity (and therefore the (*cup,coffee*) pair is scored higher than the (*car,train*) pair), the Spearman correlation between their scores and the human scores often crosses the 0.7 level. However, when evaluating with datasets such as SimLex999 (Hill et al., 2014), where the pair scores reflect similarity, the correlation of these models with human judgment is below 0.5 (Section 6).

In order to address the challenge in modeling word similarity, we propose an alternative, pattern-based, approach to word representation. In previous work patterns were used to represent a variety of semantic relations, including hyponymy (Hearst, 1992), meronymy (Berland and Charniak, 1999) and antonymy (Lin et al., 2003). Here, in order to capture similarity between words, we use *Symmetric patterns (SPs)*, such as "X *and* Y" and "X *as well as* Y", where each of the words in the pair can take either the *X* or the *Y* position. Symmetric patterns have shown useful for representing similarity between words in various NLP tasks including lexical acquisition (Widdows and Dorow, 2002), word clustering (Davidov and Rappoport, 2006) and classification of words to semantic categories (Schwartz et al., 2014). However, to the best of our knowledge, they have not been applied to vector space word representation.

Our representation is constructed in the following way (Section 3). For each word w, we construct a vector v of size V, where V is the size of the lexicon. Each element in v represents the co-occurrence in SPs of w with another word in the lexicon, which results in a sparse word representation. Unlike most previous works that applied SPs to NLP tasks, we do not use a hard coded set of patterns. Instead, we extract a set of SPs from plain text using an unsupervised algorithm (Davidov and Rappoport, 2006). This substantially reduces the human supervision our model requires and makes it applicable for practically every language for which a large corpus of text is available.

Our SP-based word representation is flexible. Particularly, by exploiting the semantics of the pattern based features, our representation can be adapted to fit the specific needs of target NLP applications. In Section 4 we exemplify this property through the ability of our model to control whether its word representations will deem antonyms similar or dissimilar. *Antonyms* are words that have opposite semantic meanings (e.g., (*small,big*)), yet, due to their tendency to co-occur in the same context, they are often assigned similar vectors by co-occurrence based representation models (Section 6). Controlling the model judgment of antonym pairs is highly useful for NLP tasks: in some tasks, like word classification, antonym pairs such as (*small,big*) belong to the same class (size adjectives), while in other tasks, like sentiment analysis, identifying the difference between them is crucial. As discussed in Section 4, we believe that this flexibility holds for various other pattern types and for other lexical semantic relations (e.g. hypernymy, the *is-a* relation, which holds in word pairs such as (*dog,animal*)).

We experiment (Section 6) with the SimLex999 dataset (Hill et al., 2014), consisting of 999 pairs of words annotated by human subjects for similarity. When comparing the correlation between the similarity scores derived from our learned representation and the human scores, our representation receives a Spearman correlation coefficient score (ρ) of 0.517, outperforming six strong baselines, including the state-of-the-art *word2vec* (Mikolov et al., 2013a) embeddings, by 5.5–16.7%. Our model performs particularly well on the verb portion of SimLex999 (222 verb pairs), achieving a Spearman score of 0.578 compared to scores of 0.163–0.376 of the baseline models, an astonishing improvement of 20.2–41.5%. Our analysis reveals that the antonym adjustment capability of our model is vital for its success.

We further demonstrate that the word pair scores produced by our model can be combined with those of word2vec to get an improved predictive power for word similarity. The combined scores result in a Spearman's ρ correlation of 0.563, a further 4.6% improvement compared to our model, and a total of 10.1–21.3% improvement over the baseline models. This suggests that the models provide complementary information about word semantics.

2 Related Work

Vector Space Models for Lexical Semantics. Research on vector spaces for word representation dates back to the early 1970's (Salton, 1971). In traditional methods, a vector for each word w is generated, with each coordinate representing the co-occurrence of w and another context item of interest – most often a word but possibly also a sentence, a document or other items. The feature representation generated by this basic construction is sometimes post-processed using techniques such as Positive Pointwise Mutual Information (PPMI) normalization and dimensionality reduction. For recent surveys, see (Turney et al., 2010; Clark, 2012; Erk, 2012).

Most VSM works share two important characteristics. First, they encode co-occurrence statistics from an input corpus directly into the word vector features. Second, they consider very little information on the syntactic and semantic relations between the represented word and its context items. Instead, a bag-of-words approach is taken.

Recently, there is a surge of work focusing on Neural Network (NN) algorithms for word representations learning (Bengio et al., 2003; Collobert and Weston, 2008; Mnih and Hinton, 2009; Collobert et al., 2011; Dhillon et al., 2011; Mikolov et al., 2013a; Mnih and Kavukcuoglu, 2013; Lebret and Collobert, 2014; Pennington et al., 2014). Like the more traditional models, these works also take the bag-of-words approach, encoding only shallow co-occurrence information between linguistic items. However, they encode this information into their objective, often a language model, rather than directly into the features.

Consider, for example, the successful word2vec model (Mikolov et al., 2013a). Its continuous-bag-of-words architecture is designed to predict a word given its past and future context. The resulted objective function is:

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

where T is the number of words in the corpus, and c is a pre-determined window size. Another word2vec architecture, skip-gram, aims to predict the past and future context given a word. Its objective is:

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

In both cases the objective function relates to the co-occurrence of words within a context window.

A small number of works went beyond the bag-of-words assumption, considering deeper relationships between linguistic items. The Strudel system (Baroni et al., 2010) represents a word using the clusters of lexico-syntactic patterns in which it occurs. Murphy et al. (2012) represented words through their co-occurrence with other words in syntactic dependency relations, and then used the Non-Negative Sparse Embedding (NNSE) method to reduce the dimension of the resulted representation. Levy and Goldberg (2014) extended the skip-gram word2vec model with negative sampling (Mikolov et al., 2013b) by basing the word co-occurrence window on the dependency parse tree of the sentence. Bollegala et al. (2015) replaced bag-of-words contexts with various patterns (lexical, POS and dependency).

We introduce a symmetric pattern based approach to word representation which is particularly suitable for capturing word similarity. In experiments we show the superiority of our model over six models of the above three families: (a) bag-of-words models that encode co-occurrence statistics directly in features; (b) NN models that implement the bag-of-words approach in their objective; and (c) models that go beyond the bag-of-words assumption.

Similarity vs. Association Most recent VSM research does not distinguish between association and similarity in a principled way, although notable exceptions exist. Turney (2012) constructed two VSMs with the explicit goal of capturing either similarity or association. A classifier that uses the output of these models was able to predict whether two concepts are associated, similar or both. Agirre et al. (2009) partitioned the wordsim353 dataset into two subsets, one focused on similarity and the other on association. They demonstrated the importance of the association/similarity distinction by showing that some VSMs perform relatively well on one subset while others perform comparatively better on the other.

Recently, Hill et al. (2014) presented the SimLex999 dataset consisting of 999 word pairs judged by humans for similarity only. The participating words belong to a variety of POS tags and concreteness levels, arguably providing a more realistic sample of the English lexicon. Using their dataset the authors show the tendency of VSMs that take the bag-of-words approach to capture association much better than similarity. This observation motivates our work.

Symmetric Patterns. Patterns (*symmetric* or not) were found useful in a variety of NLP tasks, including identification of word relations such as hyponymy (Hearst, 1992), meronymy (Berland and Charniak, 1999) and antonymy (Lin et al., 2003). Patterns have also been applied to tackle sentence level tasks such as identification of sarcasm (Tsur et al., 2010), sentiment analysis (Davidov et al., 2010) and authorship attribution (Schwartz et al., 2013).

Symmetric patterns (SPs) were employed in various NLP tasks to capture different aspects of word similarity. Widdows and Dorow (2002) used SPs for the task of lexical acquisition. Dorow et al. (2005) and Davidov and Rappoport (2006) used them to perform unsupervised clustering of words. Kozareva et al. (2008) used SPs to classify proper names (e.g., fish names, singer names). Feng et al. (2013) used SPs to build a connotation lexicon, and Schwartz et al. (2014) used SPs to perform minimally supervised classification of words into semantic categories.

While some of these works used a hand crafted set of SPs (Widdows and Dorow, 2002; Dorow et al., 2005; Kozareva et al., 2008; Feng et al., 2013), Davidov and Rappoport (2006) introduced a fully unsupervised algorithm for the extraction of SPs. Here we apply their algorithm in order to reduce the required human supervision and demonstrate the language independence of our approach.

Antonyms. A useful property of our model is its ability to control the representation of antonym pairs. Outside the VSM literature several works identified antonyms using word co-occurrence statistics, manually and automatically induced patterns, the WordNet lexicon 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). Recently, Yih et al. (2012), Chang et al. (2013) and Ono et al. (2015) proposed word representation methods that assign dissimilar vectors to antonyms. Unlike our unsupervised model, which uses plain text only, these works used the WordNet lexicon and a thesaurus.

3 Model

In this section we describe our approach for generating pattern-based word embeddings. We start by describing symmetric patterns (SPs), continue to show how SPs can be acquired automatically from text, and, finally, explain how these SPs are used for word embedding construction.

3.1 Symmetric Patterns

Lexico-syntactic patterns are sequences of words and wildcards (Hearst, 1992). Examples of patterns include "X *such as* Y", "X *or* Y" and "X *is a* Y". When patterns are instantiated in text, wildcards are replaced by words. For example, the pattern "**X** *is a* **Y**", with the **X** and **Y** wildcards, can be instantiated in phrases like "**Guffy** *is a* **dog**".

Candidate	Examples of Instances
"X *of* Y"	"point *of* view", "years *of* age"
"X *the* Y"	"around *the* world", "over *the* past"
"X *to* Y"	"nothing *to* do", "like *to* see"
"X *and* Y"	"men *and* women", "oil *and* gas"
"X *in* Y"	"keep *in* mind", "put *in* place"
"X *of the* Y"	"rest *of the* world", "end *of the* war"

Table 1:
The six most frequent pattern candidates that contain exactly two wildcards and 1-3 words in our corpus.

Symmetric patterns are a special type of patterns that contain exactly two wildcards and that tend to be instantiated by wildcard pairs such that each member of the pair can take the **X** or the **Y** position. For example, the symmetry of the pattern "**X** *or* **Y**" is exemplified by the semantically plausible expressions "cats *or* dogs" and "dogs *or* cats".

Previous works have shown that words that co-occur in SPs are semantically similar (Section 2). In this work we use symmetric patterns to represent words. Our hypothesis is that such representation would reflect word similarity (i.e., that similar vectors would represent similar words). Our experiments show that this is indeed the case.

Symmetric Patterns Extraction. Most works that used SPs manually constructed a set of such patterns. The most prominent patterns in these works are "X *and* Y" and "X *or* Y" (Widdows and Dorow, 2002; Feng et al., 2013). In this work we follow (Davidov and Rappoport, 2006) and apply an unsupervised algorithm for the automatic extraction of SPs from plain text.

This algorithm starts by defining an SP template to be a sequence of 3-5 tokens, consisting of exactly two wildcards, and 1-3 words. It then traverses a corpus, looking for frequent pattern candidates that match this template. Table 1 shows the six most frequent pattern candidates, along with common instances of these patterns.

The algorithm continues by traversing the pattern candidates and selecting a pattern p if a large portion of the pairs of words w_i, w_j that co-occur in p co-occur both in the ($X = w_i$,$Y = w_j$) form and in the ($X = w_j$,$Y = w_i$) form. Consider, for example, the pattern candidate "X *and* Y", and the pair of words "cat","dog". Both pattern instances "cat *and* dog" and "dog *and* cat" are likely to be seen in a large corpus. If this property holds for a large portion[2] of the pairs of words that co-occur in this pattern, it is selected as symmetric. On the other hand, the pattern candidate "X *of* Y" is in fact asymmetric: pairs of words such as "point", "view" tend to come only in the (X = "point",Y = "view") form and not the other way around. The reader is referred to (Davidov and Rappoport, 2006) for a more formal description of this algorithm. The resulting pattern set we use in this paper is "X *and* Y", "X *or* Y", "X *and the* Y", "*from* X *to* Y", "X *or the* Y", "X *as well as* Y", "X *or a* Y","X *rather than* Y", "X *nor* Y", "X *and one* Y", "*either* X *or* Y".

3.2 SP-based Word Embeddings

In order to generate word embeddings, our model requires a large corpus C, and a set of SPs P. The model first computes a symmetric matrix M of size $V \times V$ (where V is the size of the lexicon). In this matrix, $M_{i,j}$ is the co-occurrence count of both w_i,w_j and w_j,w_i in all patterns $p \in P$. For example, if w_i,w_j co-occur 1 time in p_1 and 3 times in p_5, while w_j,w_i co-occur 7 times in p_9, then $M_{i,j} = M_{j,i} = 1 + 3 + 7 = 11$. We then compute the Positive Pointwise Mutual Information (PPMI) of M, denoted by M^*.[3] The vector representation of the word w_i (denoted by v_i) is the i^{th} row in M^*.

Smoothing. In order to decrease the sparsity of our representation, we apply a simple smoothing technique. For each word w_i, W_i^n denotes the top n vectors with the smallest cosine-distance from v_i. We define the word embedding of w_i to be

$$v_i' = v_i + \alpha \cdot \sum_{v \in W_i^n} v$$

where α is a smoothing factor.[4] This process reduces the sparsity of our vector representation. For example, when $n = 0$ (i.e., no smoothing), the average number of non-zero values per vector is only 0.3K (where the vector size is $\sim$250K). When $n = 250$, this number reaches $\sim$14K.

[2]We use 15% of the pairs of words as a threshold.

[3]PPMI was shown useful for various co-occurrence models (Baroni et al., 2014).

[4]We tune n and α using a development set (Section 5). Typical values for n and α are 250 and 7, respectively.

4 Antonym Representation

In this section we show how our model allows us to adjust the representation of pairs of antonyms to the needs of a subsequent NLP task. This property will later be demonstrated to have a substantial impact on performance.

Antonyms are pairs of words with an opposite meaning (e.g., (*tall,short*)). As the members of an antonym pair tend to occur in the same context, their word embeddings are often similar. For example, in the skip-gram model (Mikolov et al., 2013a), the score of the (*accept,reject*) pair is 0.73, and the score of (*long,short*) is 0.71. Our SP-based word embeddings also exhibit a similar behavior.

The question of whether antonyms are similar or not is not a trivial one. On the one hand, some NLP tasks might benefit from representing antonyms as similar. For example, in word classification tasks, words such as "big" and "small" potentially belong to the same class (size adjectives), and thus representing them as similar is desired. On the other hand, antonyms are very dissimilar by definition. This distinction is crucial in tasks such as search, where a query such as "tall buildings" might be poorly processed if the representations of "tall" and "short" are similar.

In light of this, we construct our word embeddings to be *controllable* of antonyms. That is, our model contains an antonym parameter that can be turned on in order to generate word embeddings that represent antonyms as dissimilar, and turned off to represent them as similar.

To implement this mechanism, we follow (Lin et al., 2003), who showed that two patterns are particularly indicative of antonymy – "*from* X *to* Y" and "*either* X *or* Y" (e.g., "*from* **bottom** *to* **top**", "*either* **high** *or* **low**"). As it turns out, these two patterns are also symmetric, and are discovered by our automatic algorithm. Henceforth, we refer to these two patterns as *antonym patterns*.

Based on this observation, we present a variant of our model, which is designed to assign dissimilar vector representations to antonyms. We define two new matrices: M^{SP} and M^{AP}, which are computed similarly to M^* (see Section 3.2), only with different SP sets. M^{SP} is computed using the original set of SPs, excluding the two antonym patterns, while M^{AP} is computed using the two antonym patterns only.

Then, we define an antonym-sensitive, co-occurrence matrix M^{+AN} to be

$$M^{+AN} = M^{SP} - \beta \cdot M^{AP}$$

where β is a weighting parameter.[5] Similarly to M^*, the antonym-sensitive word representation of the i^{th} word is the i^{th} row in M^{+AN}.

Discussion. The case of antonyms presented in this paper is an example of one relation that a pattern based representation model can control. This property can be potentially extended to additional word relations, as long as they can be identified using patterns. Consider, for example, the hypernymy relation (is-a, as in the (*apple,fruit*) pair). This relation can be accurately identified using patterns such as "X *such as* Y" and "X *like* Y" (Hearst, 1992). Consequently, it is likely that a pattern-based model can be adapted to control its predictions with respect to this relation using a method similar to the one we use to control antonym representation. We consider this a strong motivation for a deeper investigation of pattern-based VSMs in future work.

We next turn to empirically evaluate the performance of our model in estimating word similarity.

5 Experimental Setup

5.1 Datasets

Evaluation Dataset. We experiment with the SimLex999 dataset (Hill et al., 2014),[6] consisting of 999 pairs of words. Each pair in this dataset was annotated by roughly 50 human subjects, who were asked to score the similarity between the pair members. SimLex999 has several appealing properties, including its size, part-of-speech diversity, and diversity in the level of concreteness of the participating words.

We follow a 10-fold cross-validation experimental protocol. In each fold, we randomly sample 25% of the SimLex999 word pairs ($\sim$250 pairs) and use them as a development set for parameter tuning. We use the remaining 75% of the pairs ($\sim$750 pairs) as a test set. We report the average of the results we got in the 10 folds.

Training Corpus. We use an 8G words corpus, constructed using the word2vec script.[7] Through this script we also apply a pre-processing step which employs the word2phrase tool (Mikolov et al., 2013c) to merge common word pairs and triples to expression tokens. Our corpus consists of four datasets: (a) The 2012 and 2013 crawled news articles from the ACL 2014 workshop on statistical machine translation (Bojar et al., 2014);[8] (b) The One Billion Word Benchmark of Chelba et al. (2013);[9] (c) The UMBC corpus (Han et al., 2013);[10] and (d) The September 2014 dump of the English Wikipedia.[11]

5.2 Baselines

We compare our model against six baselines: one that encodes bag-of-words co-occurrence statistics into its features (model 1 below), three NN models that encode the same type of information into their objective function (models 2-4), and two models that go beyond the bag-of-words assumption (models 5-6). Unless stated otherwise, all models are trained on our training corpus.

1. BOW. A simple model where each coordinate corresponds to the co-occurrence count of the represented word with another word in the training corpus. The resulted features are re-weighted according to PPMI. The model's window size parameter is tuned on the development set.[12]

2-3. word2vec. The state-of-the-art *word2vec* toolkit (Mikolov et al., 2013a)[13] offers two word embedding architectures: continuous-bag-of-words (**CBOW**) and **skip-gram**. We follow the recommendations of the word2vec script for setting the parameters of both models, and tune the window size on the development set.[14]

4. GloVe. GloVe (Pennington et al., 2014)[15] is a global log-bilinear regression model for word embedding generation, which trains only on the nonzero elements in a co-occurrence matrix. We use the parameters suggested by the authors, and tune the window size on the development set.[16]

[5]We tune β using a development set (Section 5). Typical values are 7 and 10.

[6]`www.cl.cam.ac.uk/~fh295/simlex.html`

[7]`code.google.com/p/word2vec/source/browse/trunk/demo-train-big-model-v1.sh`

[8]`http://www.statmt.org/wmt14/training-monolingual-news-crawl/`

[9]`http://www.statmt.org/lm-benchmark/1-billion-word-language-modeling-benchmark-r13output.tar.gz`

[10]`http://ebiquity.umbc.edu/redirect/to/resource/id/351/UMBC-webbase-corpus`

[11]`dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2`

[12]The value 2 is almost constantly selected.

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

[14]Window size 2 is generally selected for both models.

[15]`nlp.stanford.edu/projects/glove/`

[16]Window size 2 is generally selected.

5. NNSE. The NNSE model (Murphy et al., 2012). As no full implementation of this model is available online, we use the off-the-shelf embeddings available at the authors' website,[17] taking the *full document and dependency* model with 2500 dimensions. Embeddings were computed using a dataset about twice as big as our corpus.

6. Dep. The modified, dependency-based, skip-gram model (Levy and Goldberg, 2014). To generate dependency links, we use the Stanford POS Tagger (Toutanova et al., 2003)[18] and the MALT parser (Nivre et al., 2006).[19] We follow the parameters suggested by the authors.

5.3 Evaluation

For evaluation we follow the standard VSM literature: the score assigned to each pair of words by a model m is the cosine similarity between the vectors induced by m for the participating words. m's quality is evaluated by computing the Spearman correlation coefficient score (ρ) between the ranking derived from m's scores and the one derived from the human scores.

6 Results

Main Result. Table 2 presents our results. Our model outperforms the baselines by a margin of 5.5–16.7% in the Spearman's correlation coefficient (ρ). Note that the capability of our model to control antonym representation has a substantial impact, boosting its performance from $\rho = 0.434$ when the antonym parameter is turned off to $\rho = 0.517$ when it is turned on.

Model Combination. We turn to explore whether our pattern-based model and our best baseline, skip-gram, which implements a bag-of-words approach, can be combined to provide an improved predictive power.

For each pair of words in the test set, we take a linear combination of the cosine similarity score computed using our embeddings and the score computed using the skip-gram (SG) embeddings:

$$f^{+}(w_i, w_j) = \gamma \cdot f_{SP}(w_i, w_j) + (1-\gamma) \cdot f_{SG}(w_i, w_j)$$

In this equation $f_{<m>}(w_i, w_j)$ is the cosine similarity between the vector representations of words w_i and w_j according to model m, and γ is a weighting parameter tuned on the development set (a common value is 0.8).

Model	Spearman's ρ
GloVe	0.35
BOW	0.423
CBOW	0.43
Dep	0.436
NNSE	0.455
skip-gram	0.462
SP$^{(-)}$	0.434
SP$^{(+)}$	**0.517**
Joint (SP$^{(+)}$, skip-gram)	**0.563**
Average Human Score	0.651

Table 2:
Spearman's ρ scores of our SP-based model with the antonym parameter turned on (*SP*$^{(+)}$) or off (*SP*$^{(-)}$) and of the baselines described in Section 5.2. *Joint (SP*$^{(+)}$*, skip-gram)* is an interpolation of the scores produced by *skip-gram* and our *SP*$^{(+)}$ model. *Average Human Score* is the average correlation of a single annotator with the average score of all annotators, taken from (Hill et al., 2014).

As shown in Table 2, this combination forms the top performing model on SimLex999, achieving a Spearman's ρ score of 0.563. This score is 4.6% higher than the score of our model, and a 10.1–21.3% improvement compared to the baselines.

wordsim353 Experiments. The wordsim353 dataset (Finkelstein et al., 2001) is frequently used for evaluating word representations. In order to be compatible with previous work, we experiment with this dataset as well. As our word embeddings are designed to support word similarity rather than relatedness, we focus on the similarity subset of this dataset, according to the division presented in (Agirre et al., 2009).

As noted by (Hill et al., 2014), the word pair scores in both subsets of wordsim353 reflect word association. This is because the two subsets created by (Agirre et al., 2009) keep the original wordsim353 scores, produced by human evaluators that were instructed to score according to association rather than similarity. Consequently, we expect our model to perform worse on this dataset compared to a dataset, such as SimLex999, whose annotators were guided to score word pairs according to similarity.

Contrary to SimLex999, wordsim353 treats antonyms as similar. For example, the similarity score of the (*life,death*) and (*profit,loss*) pairs are 7.88 and 7.63 respectively, on a 0-10 scale. Consequently, we turn the antonym parameter off for this experiment.

Table 3 presents the results. As expected, our

[17]`http://www.cs.cmu.edu/~bmurphy/NNSE/`
[18]`nlp.stanford.edu/software/`
[19]`http://www.maltparser.org/index.html`

Model	Spearman's ρ
GloVe	0.677
Dep	0.712
BOW	0.729
CBOW	0.734
NNSE	0.78
skip-gram	**0.792**
SP$^{(-)}$	0.728
Average Human Score	0.756

Table 3:
Spearman's ρ scores for the similarity portion of wordsim353 (Agirre et al., 2009). SP$^{(-)}$ is our model with the antonym parameter turned off. Other abbreviations are as in Table 2.

Model	Adj.	Nouns	Verbs
GloVe	0.571	0.377	0.163
Dep	0.54	0.449	0.376
BOW	0.548	0.451	0.276
CBOW	0.579	0.48	0.252
NNSE	0.594	0.487	0.318
skip-gram	0.604	**0.501**	0.307
SP$^{(+)}$	**0.663**	0.497	**0.578**

Table 4:
A POS-based analysis of the various models. Numbers are the Spearman's ρ scores of each model on each of the respective portions of SimLex999.

model is not as successful on a dataset that doesn't reflect pure similarity. Yet, it still crosses the $\rho = 0.7$ score, a quite high performance level.

Part-of-Speech Analysis. We next perform a POS-based evaluation of the participating models, using the three portions of the SimLex999: 666 pairs of nouns, 222 pairs of verbs, and 111 pairs of adjectives. Table 4 indicates that our SP$^{(+)}$ model is exceptionally successful in predicting verb and adjective similarity. On verbs, SP$^{(+)}$ obtains a score of $\rho = 0.578$, a 20.2–41.5% improvement over the baselines. On adjectives, SP$^{(+)}$ performs even better ($\rho = 0.663$), an improvement of 5.9–12.3% over the baselines. On nouns, SP$^{(+)}$ is second only to *skip-gram*, though with very small margin (0.497 vs. 0.501), and is outperforming the other baselines by 1–12%. The lower performance of our model on nouns might partially explain its relatively low performance on wordsim353, which is composed exclusively of nouns.

Analysis of Antonyms. We now turn to a qualitative analysis, in order to understand the impact of our modeling decisions on the scores of antonym word pairs. Table 5 presents examples of antonym pairs taken from the SimLex999 dataset, along with their relative ranking among all pairs in the set, as judged by our model (SP$^{(+)}$ with $\beta = 10$ or SP$^{(-)}$ with $\beta = -1$) and by the best

Pair of Words	*SP*		*skip-gram*
	+AN	-AN	
new - old	1	6	6
narrow - wide	1	7	8
necessary - unnecessary	2	2	9
bottom - top	3	8	10
absence - presence	4	7	9
receive - send	1	9	8
fail - succeed	1	8	6

Table 5:
Examples of antonym pairs and their decile in the similarity ranking of our *SP* model with the antonym parameter turned on (+AN, β=10) or off (-AN, β=-1), and of the skip-gram model, the best baseline. All examples are judged in the lowest decile (1) by SimLex999's annotators.

baseline representation (skip-gram). Each pair of words is assigned a score between 1 and 10 by each model, where a score of M means that the pair is ranked at the M'th decile. The examples in the table are taken from the first (lowest) decile according to SimLex999's human evaluators. The table shows that when the antonym parameter is off, our model generally recognizes antonyms as similar. In contrast, when the parameter is on, ranks of antonyms substantially decrease.

Antonymy as Word Analogy. One of the most notable features of the skip-gram model is that some geometric relations between its vectors translate to semantic relations between the represented words (Mikolov et al., 2013c), e.g.:

$$v_{woman} - v_{man} + v_{king} \approx v_{queen}$$

It is therefore possible that a similar method can be applied to capture antonymy – a useful property that our model was demonstrated to have.

To test this hypothesis, we generated a set of 200 analogy questions of the form "X - Y + Z = ?" where X and Y are antonyms, and Z is a word with an unknown antonym.[20] Example questions include: "*stupid* - *smart* + *life* = ?" (*death*) and "*huge* - *tiny* + *arrive* = ?" (*leave*). We applied the standard word analogy evaluation (Mikolov et al., 2013c) on this dataset with the skip-gram embeddings, and found that results are quite poor: 3.5% accuracy (compared to an average 56% accuracy this model obtains on a standard word analogy dataset (Mikolov et al., 2013a)). Given these results, the question of whether skip-gram is capable of accounting for antonyms remains open.

[20]Two human annotators selected a list of potential antonym pairs from SimLex999 and wordsim353. We took the intersection of their selections (26 antonym pairs) and randomly generated 200 analogy questions, each containing two antonym pairs. The dataset is submitted with the paper.

7 Conclusions

We presented a symmetric pattern based model for word vector representation. On SimLex999, our model is superior to six strong baselines, including the state-of-the-art word2vec skip-gram model by as much as 5.5–16.7% in Spearman's ρ score. We have shown that this gain is largely attributed to the remarkably high performance of our model on verbs, where it outperforms all baselines by 20.2–41.5%. We further demonstrated the adaptability of our model to antonym judgment specifications, and its complementary nature with respect to word2vec.

In future work we intend to extend our pattern-based word representation framework beyond symmetric patterns. As discussed in Section 4, other types of patterns have the potential to further improve the expressive power of word vectors. A particularly interesting challenge is to enhance our pattern-based approach with bag-of-words information, thus enjoying the provable advantages of both frameworks.

Acknowledgments

We would like to thank Elad Eban for his helpful advice. This research was funded (in part) by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI) and the Israel Ministry of Science and Technology Center of Knowledge in Machine Learning and Artificial Intelligence (Grant number 3-9243).

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proc. of HLT-NAACL*.

Marco Baroni, Brian Murphy, Eduard Barbu, and Massimo Poesio. 2010. Strudel: A corpus-based semantic model based on properties and types. *Cognitive Science*.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proc. of ACL*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *JMLR*.

Matthew Berland and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proc. of ACL*.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, and Lucia Specia, editors. 2014. *Proc. of the Ninth Workshop on Statistical Machine Translation*.

Danushka Bollegala, Takanori Maehara, Yuichi Yoshida, and Ken ichi Kawarabayashi. 2015. Learning word representations from relational graphs. In *Proc. of AAAI*.

Kai-wei Chang, Wen-tau Yih, and Christopher Meek. 2013. Multi-Relational Latent Semantic Analysis. In *Proc. of EMNLP*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*.

Stephen Clark. 2012. Vector space models of lexical meaning. *Handbook of Contemporary Semanticssecond edition*, pages 1–42.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proc. of ICML*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537.

Dmitry Davidov and Ari Rappoport. 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proc. of ACL-Coling*.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proc. of Coling*.

Paramveer Dhillon, Dean P Foster, and Lyle H Ungar. 2011. Multi-view learning of word embeddings via cca. In *Proc. of NIPS*.

Beate Dorow, Dominic Widdows, Katarina Ling, Jean-Pierre Eckmann, Danilo Sergi, and Elisha Moses. 2005. Using Curvature and Markov Clustering in Graphs for Lexical Acquisition and Word Sense Discrimination.

Katrin Erk. 2012. Vector Space Models of Word Meaning and Phrase Meaning: A Survey. *Language and Linguistics Compass*, 6(10):635–653.

Song Feng, Jun Seok Kang, Polina Kuznetsova, and Yejin Choi. 2013. Connotation lexicon: A dash of sentiment beneath the surface meaning. In *Proc. of ACL*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proc. of WWW*.

Lushan Han, Abhay L. Kashyap, Tim Finin, James Mayfield, and Jonathan Weese. 2013. Umbc_ebiquity-core: Semantic textual similarity systems. In *Proc. of *SEM*.

Zellig Harris. 1954. Distributional structure. *Word*.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of Coling – Volume 2*.

Felix Hill, Roi Reichart, and Anna Korhonen. 2014. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *arXiv:1408.3456 [cs.CL]*.

Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proc. of ACL-HLT*.

Rémi Lebret and Ronan Collobert. 2014. Word embeddings through hellinger pca. In *Proc. of EACL*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proc. of ACL (Volume 2: Short Papers)*.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *Proc. of IJCAI*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proc. of NAACL-HLT*.

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In *Proc. of NIPS*.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Proc. of NIPS*.

Saif M. Mohammad, Bonnie J. Dorr, Graeme Hirst, and Peter D. Turney. 2013. Computing lexical contrast. *Computational Linguistics*, 39(3):555–590.

Brian Murphy, Partha Pratim Talukdar, and Tom Mitchell. 2012. Learning Effective and Interpretable Semantic Models using Non-Negative Sparse Embedding. In *Proc. of Coling*.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. Maltparser: A data-driven parser-generator for dependency parsing. In *Proc. of LREC*.

Masataka Ono, Makoto Miwa, and Yutaka Sasaki. 2015. Word embedding-based antonym detection using thesauri and distributional information. In *Proc. of NAACL*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*.

Michael Roth and Sabine Schulte im Walde. 2014. Combining Word Patterns and Discourse Markers for Paradigmatic Relation Classification. In *Proc. of ACL*.

Gerard Salton. 1971. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Sabine Schulte im Walde and Maximilian Koper. 2013. Pattern-based distinction of paradigmatic relations for german nouns, verbs, adjectives. *Language Processing and Knowledge in the Web*, pages 184–198.

Roy Schwartz, Oren Tsur, Ari Rappoport, and Moshe Koppel. 2013. Authorship attribution of micro-messages. In *Proc. of EMNLP*.

Roy Schwartz, Roi Reichart, and Ari Rappoport. 2014. Minimally supervised classification to semantic categories using automatically acquired symmetric patterns. In *Proc. of Coling*.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of NAACL*.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. Icwsm–a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *Proc. of ICWSM*.

Peter D. Turney, Patrick Pantel, et al. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence research*.

Peter D. Turney. 2008. A uniform approach to analogies, synonyms, antonyms, and associations. In *Proc. of Coling*.

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

Wenbo Wang, Christopher Thomas, Amit Sheth, and Victor Chan. 2010. Pattern-based synonym and antonym extraction. In *Proc. of ACM*.

Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proc. of Coling*.

Wen-tau Yih, Geoffrey Zweig, and John C. Platt. 2012. Polarity inducing latent semantic analysis. In *Proc. of EMNLP-CoNLL*.