Improving Semantic Similarity of Words by Retrofitting Word Vectors in Sense Level

Rui Zhang[a], Peter Schneider-Kamp[b] and Arthur Zimek[c]
Mathematics & Computer Science, University of Southern Denmark, Campusvej 55, Odense, Denmark

Keywords: Natural Language Processing, Post-processing Model, Retrofitting, Word Representations, Knowledge-based Sense Representations, Negative Sampling, Semantic Similarity.

Abstract: This paper presents an approach for retrofitting pre-trained word representations into sense level representations to improve semantic distinction of words. We use semantic relations as positive and negative examples to refine the results of a pre-trained model instead of integrating them into the objective functions used during training. We experimentally evaluate our approach on two word similarity tasks by retrofitting six datasets generated from three widely used techniques for word representation using two different strategies. Our approach significantly and consistently outperforms three state-of-the-art retrofitting approaches.

1 INTRODUCTION

Distributed word representations based on word vectors learned from distributional information about words in large corpora have become a central technique in Natural Language Processing (NLP). On basis of the distributional hypothesis (Harris, 1954), methods convert words into vectors by linguistic contexts as "predictive" models (Mikolov et al., 2013a,b; Bojanowski et al., 2017; Grave et al., 2017) or by co-occurring words as "count-based" models (Pennington et al., 2014). Both of them depend on "co-occurrence" information on words in a large unlabeled corpus. In general, we can observe that the larger data they use, the better such methods tend to perform on NLP tasks.

These approaches for constructing vector spaces predominantly focus on contextual relationships or word morphology. They disregard the constraints obtainable from lexicons which provide semantic information by identifying synonym, antonym, hypernymy, hyponymy, and paraphrase relations. This impedes their performance on word similarity tasks and applications where word similarity plays a significant role such as, e.g., text simplification.

Existing approaches for exploiting external semantic knowledge to improve word vectors can be grouped into two categories (Vulic and Glavas, 2018): (1) *joint specialization* models integrate semantic constraints on word similarity by modifying the objective of the original word vector training in joint neural language models (Yu and Dredze, 2014; Mikolov et al., 2018) or by incorporating relation-specific constraints like the co-occurrence matrix (Chang et al., 2013) or word ordinal ranking (Liu et al., 2015a) into models; (2) *post-processing* models retrofit or refine the pre-trained distributional word vectors in order to fit the semantic constraints (Faruqui et al., 2015; Shiue and Ma, 2017; Lee et al., 2018; Vulic and Glavas, 2018).

Compared with joint specialization models, post-processing models are more flexible because they can be applied to all kinds of distributional spaces. Furthermore, post-processing approaches do not need to re-train models on the large corpora typically used, which is more convenient both for research purposes and in applications.

Recent work on post-processing approaches, mainly based on the graph-based learning technique (Faruqui et al., 2015; Yu et al., 2017; Lee et al., 2018), has had a great influence on the field of retrofitting word vectors. Yet, these studies specifically show the significant improvements on benchmarks of evaluation datasets such as MEN (Bruni et al., 2014) or WordSim-353 (Finkelstein et al., 2002) which conflate relatedness or association with similarity rather than on datasets that exclusively focus on word similarity.

[a] https://orcid.org/0000-0001-9126-9790
[b] https://orcid.org/0000-0003-4000-5570
[c] https://orcid.org/0000-0001-7713-4208

Zhang, R., Schneider-Kamp, P. and Zimek, A.
Improving Semantic Similarity of Words by Retrofitting Word Vectors in Sense Level.
DOI: 10.5220/0008953001080119
In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence (ICAART 2020) - Volume 2*, pages 108-119
ISBN: 978-989-758-395-7; ISSN: 2184-433X

In this paper, we propose a new approach that obtains the new word vectors by retrofitting pre-trained vectors by exploiting synonyms and antonyms obtained from *thesaurus.com*[1] for the 100,000 most frequently-used English words from Wiktionary.[2] For evaluation purposes, we use the same standard benchmarks as used by Vulic and Glavas (2018), i.e., SimLex-999 (Hill et al., 2015) and SimVerb-3500 (Gerz et al., 2016), and also Stanford's Contextual Word Similarities (Huang et al., 2012) used by Lee et al. (2018).

We find that our model consistently provides significant improvements in word similarity compared to state-of-the-art retrofitting models. Specifically, we obtain a Spearman correlation of 0.765 on SimLex-999, improving on the score of 0.76 achieved by the best published model (Recski et al., 2016). Our approach uses negative sampling (Mikolov et al., 2013b) for simplifying the process of neural networks with complicated hidden layers (Vulic and Glavas, 2018). The contributions of this paper are twofold: (i) we generate the vectors of a word with different definitions using the original pre-trained word vector, based on the synonym and antonym sets in the different word senses, and (ii) we enhance the performance of word vectors on the task of semantic similarity of words by only using superficial synonyms and antonyms knowledge.

Figure 1 depicts the structure of our approach. We input the synonyms of a target word with the j-th sense into our model which are the positive samples giving a positive influence on this target word. The antonyms of this target word are added to adjust the probability of this target word. Such adjustments can also affect these synonyms such that we could update the vectors of these synonyms first. Finally under the influence of the updated synonyms and the antonyms, the vector of the target word is retrofitted with the j-th sense.

This paper is structured as follows. An overview on recent related work is provided in Section 2. In Section 3, we introduce the principle of our model, define the objective, and describe the steps of optimization as well as how to update word embeddings into sense embeddings. In Section 4 we describe our experimental setup with datasets, evaluation procedures, and the evaluated tasks. In Section 5 we discuss the results of different configurations on the benchmarks. Finally, we present conclusions and future challenges in Section 6.

[1]https://www.thesaurus.com/

[2]https://gist.github.com/h3xx/1976236

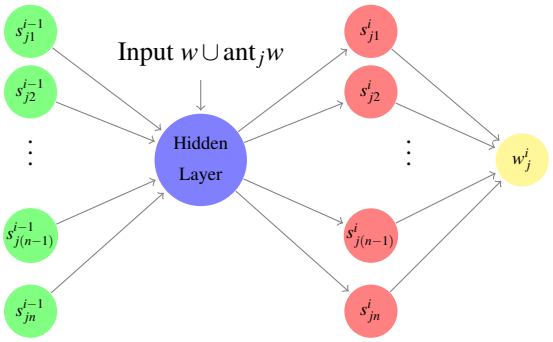


Figure 1: Graphical sketch of the proposed model. s^i_{jn} refers to the n-th synonym of the target word w with the j-th sense in the i-th step and $ant(w_j)$ indicates the antonym set of the target word w with the j-th sense.

2 RELATED WORK

The representation of words which provide continuous low-dimensional vector representations of words plays a pivotal role and has been widely studied in NLP domain. Vector Space Models (VSM) is the basis of many prominent methodologies for word representation learning. The earliest VSMs considered a vector space in document level which uses the vocabulary directly as the features (Salton et al., 1975). Subsequently many kinds of weight computation metrics of individual dimensions such as word frequencies or normalized frequencies (Salton and McGill, 1986) have been proposed. This research has succeeded in various NLP tasks.

However, one crucial problem with the huge corpus is the high dimensionality of the produced vectors. A common solution is dimensionality reduction making use of the Singular Value Decomposition (SVD). Learning low-dimensional word representations directly from text corpora is another strategy that has been achieved by leveraging neural networks. These models are commonly known as word embeddings. Some prominent word embedding architectures have been constructed depending on contextual relationships or word morphology. Beyond that, more complex approaches have been proposed attempting to cure some deficiencies (e.g., conflation of meaning) by exploiting sub-word units (Wieting et al., 2016), probability distributions (Athiwaratkun and Wilson, 2017), specialized similarity measures (Soares et al., 2019), knowledge resources (Camacho-Collados and Pilehvar, 2018), etc.

The process of producing word embeddings is not able to capture different meanings of the same word. And for downstream tasks, the meaning conflation can have a negative influence on accurate semantic modeling, e.g., "*mouse-screen*" and "*mouse-*

cow". There is no relation between "*screen*" and "*cow*", but they can be connected by two different senses of "*mouse*", i.e., computer device and animal.

Partitioning the meanings of words into multiple senses is not an easy task. Computational approaches relying on text corpora or semantic knowledge (such as a dictionary or thesaurus), generally can be categorized into unsupervised techniques and knowledge-based techniques.

2.1 Unsupervised Techniques

Unlabeled monolingual corpora can be exploited for word sense disambiguation by clustering-based approaches or joint models. Clustering the context in which an ambiguous word occurs can discern senses automatically. Context-group discrimination (Schütze, 1998) is an approach to compute the centroid vector of the context of an ambiguous word, and then cluster these context centroid vectors into a predetermined number of clusters. Context clusters for an ambiguous word are interpreted as representations for different senses of this word. Models applying this strategy are also called two-stage models (Erk and Padó, 2008; Van de Cruys et al., 2011).

Joint models (Li and Jurafsky, 2015; Qiu et al., 2016) are proposed as various extensions of traditional word embedding models. The primary difference in contrast to clustering-based approaches is that joint models merge the clustering and the sense representation step. In this way, the joint model can dynamically select the potential sense for an ambiguous word during training. Topical Word Embeddings (TWE) (Liu et al., 2015b) are proposed for inducing the sense representations of a word based only on its local context, which reduces computational complexity. Joint models have serious limitations, though, which require the disambiguation of the context of a word as well as predetermining a fixed number of senses per word.

Another recent branch of unsupervised techniques is to generate contextualized word embeddings. Here, context-sensitive latent variables for each word are inferred from a fuzzy word clustering and then integrated to the sequence tagger as additional features (Li and McCallum, 2005).

2.2 Knowledge-based Techniques

Knowledge-based techniques about semantic representations fall into three categories: (1) improving word representations, (2) using sense representations, and (3) using concept and entity representations. Studies related to all three categories make use of similar knowledge resources. In particular, WordNet (Baker et al., 1998) as an example of expert-made resources and Wikipedia as an example of collaboratively-constructed resources are widely applied (Camacho-Collados and Pilehvar, 2018). Some similar collaborative works powered by Wikipedia like Freebase (Bollacker et al., 2008) and DBpedia (Bizer et al., 2009) also provide large structured data in the form of the knowledge base. Further examples are BabelNet (Navigli and Ponzetto, 2012), a combination of expert-made resources and collaboratively-constructed resources, and ParaPhrase DataBase (PPDB) (Ganitkevitch et al., 2013) gathering over 150 million paraphrases and providing a graph structure. In the following, we shortly recap some of the relevant related work for each of the three categories.

2.2.1 Improving Word Representations

The earlier attempts to improve word embedding using lexical resources modified the objective functions of a neural network model for learning a word representation (Yu and Dredze, 2014; Kiela et al., 2015). Typically, they integrate the external semantic constraints into the learning process directly, resulting in joint specialization models. Some recent approaches try to improve the pre-trained word vectors through post-processing (Faruqui et al., 2015; Lengerich et al., 2018), which is a more versatile approach than the joint models. The popular term "retrofitting" is used when implanting external lexicon knowledge into random pre-trained word vectors. Given any pre-trained word vectors, generated by any tools or techniques, the main idea of graph-based retrofitting is to minimize the distance between synonyms and maximize the distance between antonyms. Building upon the retrofitting idea, explicit retrofitting constructs a neural network by modeling pairwise relations (Goikoetxea et al., 2015; Vulic and Glavas, 2018), which also specialize vectors of words unseen in external lexical resources.

2.2.2 Using Sense Representations

The second category consists of sense vector representation techniques. These generate vectors by "deconflating" a word (with conflated meanings) into several individuals with different senses. Methods based on linear models draw support by the synonym and antonym sets of the target word with different senses to retrofit the original word vectors (Pilehvar and Collier, 2016; Lee et al., 2018). Chen et al. (2015) exploit a convolutional neural network architecture for initializing sense embedding. Neelakantan et al.

(2015) rely on the Skip-gram model for learning sense embeddings. The Lesk algorithm (Vasilescu et al., 2004) has been adapted for learning word sense embeddings (Yang and Mao, 2016).

2.2.3 Using Concept and Entity Representations

The main idea in this branch is to construct a strong relation between related entities. Given a knowledge base as a set of triples $\{(e_1, e_2, r)\}$, where e_1 and e_2 are entities and r is the relation between them, the goal is to approach the entities by the relation r ($\vec{e_1} + \vec{r} \approx \vec{e_2}$) for all triples in the space. Typical approaches integrating concepts and entities from external knowledge bases rely exclusively on knowledge graphs to build an embedding space for entities and relations (Bordes et al., 2013). In addition, some hybrid models have been proposed to exploit text corpora and knowledge bases as well (Camacho-Collados et al., 2016).

2.3 Summary

The approach we propose is inspired by Word2Vec (Mikolov et al., 2013b,a) which reduces the complexity of the hidden layer so that the approach is both simple and practical. Considering the importance of semantic relation and sense representation, we retrofit the unitary pre-trained word vectors from the corpus into sense level representations by external semantic knowledge.

3 NEGATIVE-SAMPLING RETROFITTING

An important aspect in Natural Language Processing is the Statistical Language Model which can be used to calculate the probability of a sentence

$$\Pr(w_1 w_2 \dots w_N).$$

We assume $w_1^N = (w_1 w_2 \dots w_N)$. According to Bayes' theorem, this probability could be decomposed into conditional probabilities

$$\Pr(w_1), \Pr(w_2|w_1), \dots, \Pr(w_N|w_1^{N-1})$$

which are the parameters of a Language Model. In order to find the optimal model parameters, we will do optimization on an objective function $\Pr(w|\mathrm{POS}(w))$ generated by a maximum likelihood estimate method, where $\mathrm{POS}(w)$ is the positive sample set of the target word w. That is, under the positive condition, the probability of w should be greater than under the negative condition. Thus, we explore a method that uses a set of linguistic constraints from an external lexical resource,

$$LC = \{(w_j, \mathrm{syn}(w_j), \mathrm{ant}(w_j)) | w \in \mathcal{V}\},$$

each consisting of a word w from the associated vocabulary $\mathcal{V}$ with the j-th sense and its synonyms ($\mathrm{syn}(w_j)$) as positive samples and its antonyms ($\mathrm{ant}(w_j)$) as negative samples to retrofit the vector of each target word. More specifically, the synonyms and antonyms of the corresponding senses helps us to refine the vectors into a sense level. We employ Negative Sampling (Mikolov et al., 2013b), a simplified version of Noise Contrastive Estimation (NCE) (Gutmann and Hyvärinen, 2010) which aims at improving the quality of results and decreasing training time.

Our approach consists of two major components: (1) updating the synonym set of the target word in the j-th sense, and (2) generating the corresponding new sense embedding of this word.

3.1 Objective Functions

Let $\boldsymbol{X} = \{\boldsymbol{x}_w | w \in \mathcal{V}\}$, $\boldsymbol{x}_w \in \mathbb{R}^d$ be the pre-trained d-dimensional distributed vector space and let $\boldsymbol{Y} = \{\boldsymbol{y}_w | w \in \mathcal{V}\}$, $\boldsymbol{y}_w = \{\boldsymbol{y}_{w_j}\}_{j=1}^{|sense_w|}$, $\boldsymbol{y}_{w_j} \in \mathbb{R}^d$ be the corresponding retrofitted sense vector space. Recall that $\mathrm{syn}(w)$ is the positive sample set and $\mathrm{ant}(w)$ is the negative sample set of target word w, respectively. Therefore, for any word u in the pre-trained word vector space, we first define Equation (1) to denote the label of word u. That means if the word u is the target word, the corresponding optimization process for the objective function will be activated.

$$L^w(u) = \begin{cases} 1, & u = w \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

According to the given positive sample set $\mathrm{syn}(w)$, the objective is to maximize the conditional probability of a word under a condition of its synonyms. Certainly, each synonym can affect this condition, either on its own or with other related synonyms. On the basis of the above, we propose two strategies for this goal. One would be to maximize a series of probability functions if we set each synonym as an individual (NS-sv):

$$\begin{aligned} \mathcal{L} &= \prod_{w \in \mathcal{V}} \prod_{\tilde{w} \in \mathrm{syn}(w)} g(\boldsymbol{x}_w, \boldsymbol{\theta}) \\ &= \prod_{w \in \mathcal{V}} \prod_{\tilde{w} \in \mathrm{syn}(w)} \prod_{u \in \{w\} \cup \mathrm{ant}(w)} \Pr(\boldsymbol{x}_u | \boldsymbol{x}_{\tilde{w}}) \end{aligned} \tag{2}$$

where θ is the parameter matrix of the hidden layer, and g is the objective conditional probability function.

An alternative strategy is to maximize the conditional probability under the integration of the synonyms (NS-sv-sum):

$$\begin{aligned}\mathcal{L} &= \prod_{w\in\mathcal{V}} g(\boldsymbol{x}_w, \boldsymbol{\theta}) \\ &= \prod_{w\in\mathcal{V}} \prod_{u\in\{w\}\cup \mathrm{ant}(w)} \Pr(\boldsymbol{x}_u|\boldsymbol{x}_{\widetilde{w}})\end{aligned} \tag{3}$$

where $\widetilde{w}$ is composed of all synonyms instead of each single word. In our approach, we integrate them by:

$$\boldsymbol{x}_{\widetilde{w}} = \sum_{s\in\mathrm{syn}(w)} \boldsymbol{x}_s \tag{4}$$

3.2 Optimization

No matter which strategy we use to produce the objective, the goal is to maximize the probability of the positive sample and, simultaneously, to minimize the probability of the negative sample. We use a sigmoid function as the activation function:

$$\sigma(x) = \frac{1}{1+\exp(x)} \tag{5}$$

such that the derivative of sigmoid function is:

$$\sigma'(x) = \sigma(x)\left[1-\sigma(x)\right] \tag{6}$$

Thence, $\Pr(\boldsymbol{x}_u|\boldsymbol{x}_{\widetilde{w}})$ could be denoted as:

$$\Pr(\boldsymbol{x}_u|\boldsymbol{x}_{\widetilde{w}}) = \begin{cases} \sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}^u\right), & L^w(u)=1 \\ 1-\sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}^u\right), & L^w(u)=0 \end{cases} \tag{7}$$

Putting the pieces together, $g(w,\boldsymbol{\theta})$ would become:

$$g(\boldsymbol{x}_w, \boldsymbol{\theta}) = \prod_{u\in\{w\}\cup\mathrm{ant}(w)} \sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}^u\right)^{L^w(u)} \cdot \left(1-\sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}^u\right)\right)^{1-L^w(u)} \tag{8}$$

The purpose of training by negative sampling is to maximize the objective with respect to the model parameters by the commonly used log-likelihood function:

$$\begin{aligned}\log g(\boldsymbol{x}_w, \boldsymbol{\theta}) = \sum_{u\in\{w\}\cup\mathrm{ant}(w)} \Big\{ & L^w(u)\cdot\log\left[\sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}^u\right)\right] + \\ & \left[1-L^w(u)\right]\cdot\log\left[1-\sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}^u\right)\right] \Big\}\end{aligned} \tag{9}$$

To explain the procedure in detail, we let F_w^{ij} represent the log-likelihood function of the target word w in the i-th step with the j-th sense:

$$\begin{aligned}F_w^{ij} = & L_{ij}^w(u)\cdot\log\left[\sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}_{ij}^u\right)\right] + \\ & \left[1-L_{ij}^w(u)\right]\cdot\log\left[1-\sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}_{ij}^u\right)\right]\end{aligned} \tag{10}$$

We choose gradient ascent for optimization. The parameters $\boldsymbol{\theta}_{ij}^u$ and $\boldsymbol{x}_{\widetilde{w}}$ are updated by the learning rate α as follows:

$$\begin{aligned}\boldsymbol{\theta}_{ij}^u &= \boldsymbol{\theta}_{ij}^u + \alpha\frac{\partial F_w^{ij}}{\partial \boldsymbol{\theta}_{ij}^u} \\ \boldsymbol{x}_{\widetilde{w}} &= \boldsymbol{x}_{\widetilde{w}} + \alpha\sum_{u\in\{w\}\cup\mathrm{ant}_j(w)} \frac{\partial F_w^{ij}}{\partial \boldsymbol{x}_{\widetilde{w}}}\end{aligned} \tag{11}$$

The gradients above are calculated as follows:

$$\begin{aligned}\frac{\partial F_w^{ij}}{\partial \boldsymbol{\theta}_{ij}^u} &= \left[F_w^{ij} - \sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}_{ij}^u\right)\right]\boldsymbol{x}_{\widetilde{w}} \\ \frac{\partial F_w^{ij}}{\partial \boldsymbol{x}_{\widetilde{w}}} &= \left[F_w^{ij} - \sigma\left(\boldsymbol{x}_{\widetilde{w}}^{\mathrm{T}}\boldsymbol{\theta}_{ij}^u\right)\right]\boldsymbol{\theta}_{ij}^u\end{aligned} \tag{12}$$

Consequently, we obtain the updating function for the parameter matrix and the synonym embeddings.

3.3 Generating Sense Vectors

In this part, we collect the updated synonym vectors of the target word in the j-th sense to generate the target word vector with the j-th sense:

$$\boldsymbol{y}_{w_j} := \boldsymbol{x}_w + \sum_{s\in\mathrm{syn}(w_j)} \boldsymbol{x}_s \tag{13}$$

Moreover, to ensure that each synonym could contribute to the new word vector even if it has no pre-trained vector, we give such a synonym an initial vector randomly, and revise it by the target word vector:

$$\boldsymbol{x}_{s_k} := \lambda\boldsymbol{x}_{s_k} + (1-\lambda)\boldsymbol{x}_w \tag{14}$$

where λ is a weight for unknown neighbor vectors to keep balance of the whole vector space.

3.4 Result

As a result of the procedure, we have sense embeddings

$$\boldsymbol{y_i} = \{\boldsymbol{y}_{w_j}\}_{j=1}^{|sense_{w_i}|}$$

of each word w in the vocabulary $\mathcal{V}$. For each word w, we update the embeddings of the synonyms of w with the j-th sense by the pre-trained word vector of w. Likewise, the antonyms of w are utilized for negative sampling. After traversing of all synonyms, we aggregate these updated synonym embeddings with the pre-trained word embedding $\boldsymbol{x}_w$ to produce the new j-th sense embedding of word w.

Experimental experience tells that initializing the parameter matrix to be the same as the pre-trained vector matrix can generate effective sense retrofitted vectors with only a few iterations.

4 EXPERIMENTAL SETUP

We evaluate our approach on two aspects: semantic relatedness and contextual word similarity. We use the average of all sense vectors to represent the missing word. For all models reported in this paper, the same processing method and the same computation method are applied to compare their performance.

4.1 Datasets

We first experiment with three widely used and publicly available pre-trained word vectors for English corpora.

1. **Word2Vec** (Mikolov et al., 2013b,a): Word2Vec is fast and widely used. In practice, we use the python module[3] (Sujono, 2015) (which implements the core of Word2Vec as the gensim implementation (Řehůřek and Sojka, 2010)) to train enwik9[4] by CBOW model and Skip-gram model with Negative Sampling separately in 300 dimensions, where we use context windows of size 5 and 5 negative examples.
2. **GloVe Vectors**[5] (Pennington et al., 2014): The GloVe word vector approach integrates the global co-occurrence matrix of word pairs. We use the pre-trained word vectors directly. 6B.50d, 6B.100d, 6B.200d and 6B.300d are trained on Wikipedia 2014 with English Gigaword in vector length of the range 50 to 300 respectively, and 42B.300d is trained on Common Crawl in 300 dimensions.
3. **FastText Vectors**[6] (Bojanowski et al., 2017): FastText is an extension of the continuous skip-gram model by summing the n-gram vectors. We use their pre-trained word vector files directly. crawl-300d-2M is trained on Common Crawl including 2 million word vectors, and wiki-300d-1M is trained on Wikipedia 2017, UMBC webbase corpus and *statmt.org* dataset including 1 million word vectors.

As we mentioned above, we generated the synonym sets and antonym sets of the 100,000 most-frequently used words from Wiktionary via the API of *thesaurus.com*[7]. Each word is a leader of its synonym and antonym sets corresponding to its definitions. After inputting them into our program, we are going to mark the leading word as $w\#j$, denoting the word w with the j-th sense. It is worth mentioning that we choose *thesaurus.com* as our knowledge base rather than WordNet which considers both semantic relations and lexicon relations. That is because the primary but superficial semantic knowledge is a cost effective way during training in practice.

[3]https://github.com/deborausujono/word2vecpy
[4]http://mattmahoney.net/dc/textdata.html
[5]https://nlp.stanford.edu/projects/glove/
[6]https://fasttext.cc/docs/en/english-vectors.html
[7]https://github.com/Manwholikespie/thesaurus

4.2 Evaluation Measure

For semantic relatedness task, we evaluate the quality of the retrofitted embedding spaces on two word similarity benchmarks: **SimLex-999** (Hill et al., 2015), which comprises 666 noun pairs, 222 verb pairs and 111 adjective pairs; and **SimVerb-3500** (Gerz et al., 2016), which consists of 3500 verb pairs covering all normed verb types of 827 distinct verbs.

For contextual word similarity task, we conduct experiments with the **Stanford's Contextual Word Similarities** (SCWS) (Huang et al., 2012) which includes 2003 word pairs together with human-rated scores. Higher scores indicate higher semantic similarity.

Our experiments are all based on intrinsic evaluation for the quality and coherence of vector space. We use **Spearman's ρ rank correlation coefficient** (Well and Myers, 2003) between the cosine similarity scores calculated by the retrofitted vectors and the human-provided ratings for assessment.

On the other hand, according to Faruqui et al. (2016) and Rastogi et al. (2015), it is necessary to perform statistical significance tests to the difference between the Spearman's Correlations even for comparisons on small evaluation sets. Rastogi et al. (2015) introduce $\sigma^{\rho}_{p_0}$ as the *Minimum Required Difference for Significance (MRDS)* which satisfies the the following:

$$(\rho_{AB} < \rho) \wedge \left(|\rho_{\hat{BT}} - \rho_{\hat{AT}}| < \sigma^{\rho}_{p_0}\right) \Rightarrow p\text{-value} > p_0 \quad (15)$$

where A and B are the lists of ratings over the same items, produced by the competitive models and T denotes the gold ratings T. ρ_{AT}, ρ_{BT}, and ρ_{AB} denote the Spearman's correlations between $A:T$, $B:T$, and $A:B$, respectively. Then let $\rho_{\hat{AT}}$, $\rho_{\hat{BT}}$, and $\rho_{\hat{AB}}$ be their empirical estimates. This proposition indicates that differences in correlations, if below the MRDS threshold, are not statistically significant. Rastogi et al. (2015) also provide the MRDS values for **SimLex-999** word similarity dataset and here we provide the threshold[8] for **SimVerb-3500** and **Stanford's Contextual Word Similarities** in Table 1.

[8]https://github.com/se4u/mvlsa provides the way to assign a minimum threshold to a testset.

Table 1: The Minimum Required Difference for Significance (MRDS) values for SimLex-999 (SL), SimVerb-3500 (SV) and Stanford's Contextual Word Similarities (SCWS).

Dataset	**Size**	$\sigma^{0.5}_{0.01}$	$\sigma^{0.7}_{0.01}$	$\sigma^{0.9}_{0.01}$	$\sigma^{0.5}_{0.05}$	$\sigma^{0.7}_{0.05}$	$\sigma^{0.9}_{0.05}$
SL	999	0.073	0.057	0.032	0.052	0.040	0.023
SV	3500	0.039	0.030	0.017	0.027	0.021	0.012
SCWS	2003	0.051	0.04	0.023	0.036	0.028	0.016

Table 2: Spearman's correlation for three word distributed representations (300 dimensions), Word2Vec (w2v), GloVe (glove) and FastText (FT) on SimLex-999, and the performance comparison with two strategies of retrofitting models, NS-sv and NS-sv-sum (MaxSim / AveSim) using $\sigma^{0.9}_{0.05}$ as the threshold.

	w2v.cb	w2v.sg	glove.42B	glove.6B	FT.wiki	FT.crawl
baseline	0.230	0.293	0.374	0.371	0.450	0.503
NS-sv	**0.642/0.650**	**0.640/0.637**	**0.688/0.698**	**0.706/0.725**	**0.678/0.688**	0.723/**0.765**
NS-sv-Sum	0.609/0.571	0.596/0.487	0.674/0.62	0.693/0.675	0.667/0.633	**0.733**/0.698

4.3 Task 1: Semantic Relatedness

This task is to model the semantic similarity. A higher score indicates the higher semantic similarity. The sense evaluation metrics learned from Reisinger and Mooney (2010) compute two kinds of scores, **maximum score** for the evaluation of sense representations and **average score** for the evaluation of word representations:

$$MaxSim = \max_{w_{m_j} \in D_{wm}, w_{n_k} \in D_{wn}} \cos\left(\boldsymbol{x}_{w_{m_j}}, \boldsymbol{x}_{w_{n_k}}\right) \quad (16)$$

$$AveSim = \frac{\sum_{w_{m_j} \in D_{wm}} \sum_{w_{n_j} \in D_{wn}} \cos\left(\boldsymbol{x}_{w_{m_j}}, \boldsymbol{x}_{w_{n_k}}\right)}{|D_{w_m}| \cdot |D_{w_n}|} \quad (17)$$

where D_{w_m} is the definition set of the word w_m, and D_{w_n} is the definition set of the word w_n. Compared with the original metrics based on unsupervised techniques, we do not need to predetermine the number of sense clusters which should differ from word to word. Thus, instead of the uniform number of clusters, we use the number of senses of each word to average the word similarity.

4.4 Task 2: Contextual Word Similarity

The goal of this task is to measure the semantic relatedness with contextual information which covers the shortage in semantic relatedness task. We also adopt MaxSimC / AvgSimC metrics to compute scores for each word pair (Reisinger and Mooney, 2010). A higher score indicates the higher semantic similarity.

$$MaxSimC = d(\hat{\pi}(\boldsymbol{x}_{w_m}), \hat{\pi}(\boldsymbol{x}_{w_n})) \quad (18)$$

$$AveSimC = \frac{\sum_{w_{m_j} \in D_{wm}} \sum_{w_{n_j} \in D_{wn}} d_{c,w_{m_j}} d_{c',w_{n_k}} d\left(\boldsymbol{x}_{w_{m_j}}, \boldsymbol{x}_{w_{n_k}}\right)}{|D_{w_m}| \cdot |D_{w_n}|} \quad (19)$$

where $d_{c,w_{m_j}} = \cos(\boldsymbol{v}(c), \boldsymbol{x}_{w_m})$ is the likelihood of context c belonging to the j-th sense group of the word w_m, and $\hat{\pi}(w_m) = \arg\max_{w_{mj} \in D_{wm}} d_{c,w_{m_j}}$. We select 5 words before and after the target word in the word pairs respectively. Stopwords are removed from the context. There are total 10 word involved in as the context of the target word under the perfect condition. And the context vector of the target word as $\boldsymbol{v}(c)$ are integrated with all context word embeddings. Note that based on our word vector space, the context is not disambiguation, so the solution is the sum of all sense vectors of each context word.

4.5 Model Configuration

For our model, we set a learning rate varying with the number of the synonyms of the target word and the initialized iteration as 1.0. Moreover, the parameter λ to revise the undefined word vector is set 0.5. In all experiments except the iteration part, we choose $iter = 100$ for NS-sv and $iter = 10$ for NS-sv-sum. And for statistical significance, we choose $\sigma^{0.9}_{0.05}$, the small threshold value in Table 1.

5 RESULTS AND DISCUSSION

In this part, we show experiment results from our retrofitting method.

5.1 Baseline Methods

Tables 2 and 3 show the results of retrofitting the three standard vectors on each benchmark dataset. The second row in each table shows the performance of the baseline vectors. If there is only one sense of a word, its maximum score (*MaxSim*) and average

Table 3: Spearman's correlation for three word distributed representations (300 dimensions), Word2Vec (w2v), GloVe (glove) and FastText (FT) on SimVerb-3500, and the performance comparison with two strategies of retrofitting models, NS-sv and NS-sv-sum (MaxSim / AveSim), using $\sigma_{0.05}^{0.9}$ as the threshold.

	w2v.cb	w2v.sg	glove.42B	glove.6B	FT.wiki	FT.crawl
baseline	0.160	0.184	0.226	0.227	0.357	0.426
NS-sv	**0.592/0.569**	**0.594/0.568**	**0.604/0.591**	**0.614/0.609**	**0.623/0.615**	**0.639/0.659**
NS-sv-Sum	0.524/0.477	0.502/0.412	0.584/0.558	0.597/0.578	0.562/0.532	0.632/0.611

Table 4: Spearman's correlation for two word distributed representations (100 dimensions) on SimLex-999 (SL) and SimVerb-3500 (SV) , prior works and our approach, using $\sigma_{0.05}^{0.9}$ as the threshold.

Corpus	Dataset	GloVe	re-Faruqui	GenSense	ER-Specialized	NS-sv	NS-sv-sum
Wikipedia	SL	0.265	0.421	0.446/0.417	0.445	**0.683/0.684**	0.648/0.600
	SV	0.154	0.240	0.289/0.259	0.281	**0.609/0.594**	0.557/0.526
Twitter	SL	0.122	0.295	0.290/0.244	0.365	**0.651/0.631**	0.589/0.495
	SV	0.052	0.145	0.160/0.127	0.225	**0.583/0.558**	0.529/0.473

Table 5: Spearman's correlation using GloVe vectors (100 dimensions) on the Stanford's Contextual Word Similarities (SCWS) task, prior work and our approach (MaxSimC / AvgSimC), computing with the sum of context word embeddings (SCWS-sum) and with the average of context word embeddings (SCWS-avg), using $\sigma_{0.05}^{0.9}$ as the threshold.

	SCWS-sum	SCWS-avg
glove.twitter	0.428	0.428
GenSense	0.428/**0.322**	0.428/0.272
NS-sv	**0.467** /0.313	**0.467**/0.313
NS-sv-sum	0.444/0.264	0.444/0.264

score (*AveSim*) will be the same. From Tables 2 and 3, the *MaxSim* and *AveSim* both prove the proposed model is robust and useful. The largest improvement is an increase of more than 0.3 on two word similarity tasks and FastText Crawl retrofitted vectors produce the top correlation score of 0.765. The average correlation between a human rater and the average of all other raters is 0.78 and the latest top record by Recski et al. (2016) is 0.76, which implies that our method does promote the measuring of semantic similarity of words. In contrast with NS-sv-sum, in general, NS-sv yields the better results because the connections among the synonyms are weak. That means the synonyms are the condition of the word sense during our study rather than the constraints of hierarchical semantic relations. But the experiment of vector length later also indicates that the gap of performance of NS-sv and NS-sv-sum reduces with increasing word vector length. We conjecture that this is the case as the vectors with higher dimensions have higher expressivity regarding the difference and, consequently, are less easily offset during integration.

5.2 Comparison with Prior Works

A comparison to prior works is shown in Table 4. The three previous models are: re-Faruqui[9] (Faruqui et al., 2015), GenSense[10] (Lee et al., 2018), and ER-Specialized[11] (Vulic and Glavas, 2018), trained on Wikipedia and Twitter using the GloVe tool. We run their published code to obtain word vectors. Except for GenSense which needs the weights for their semantic lexicons to discriminate senses of the target word, we applied these models with our new lexicons to decrease the difference. In addition, we kept their default parameter values without any change. Surprisingly, although the prior models still keep their superiority compared with baseline vectors, they become somewhat weak when the models only rely on the primary semantic relations. From Table 4, the outcome of this experiment suggests that the quality of our vectors significantly improves compared to each of the three models. The Spearman's correlation scores of both NS-sv and NS-sv-sum exceed GenSense by more than 0.2, which also shows a great performance under word vectors from Wikipedia corpus. And under the word vector from Twitter, NS-sv and NS-sv-sum surpass ER-Specialized by more than 0.2.

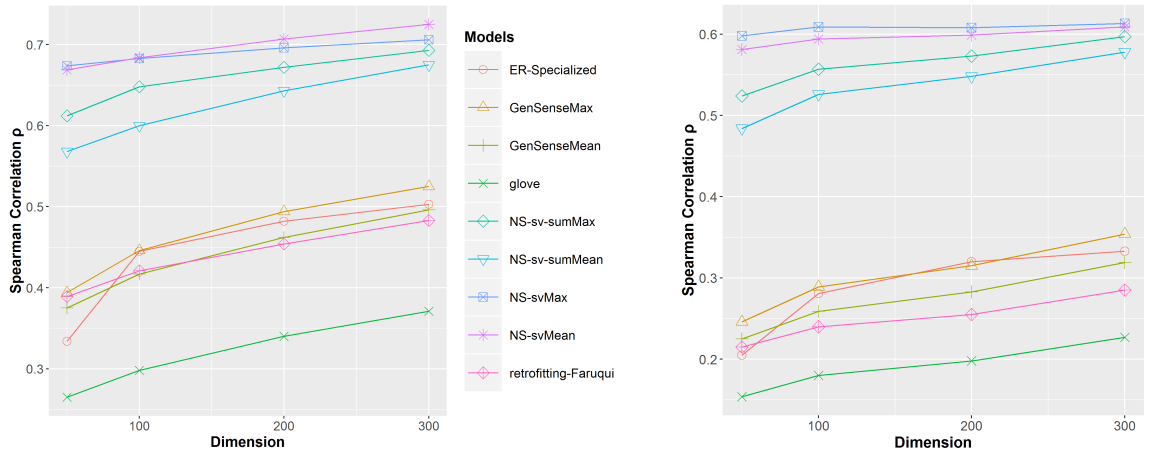
5.3 Contextual Word Similarity

Table 5 shows the Spearman's correlation of SCWS dataset. With the contextual information, sense

[9]https://github.com/mfaruqui/retrofitting

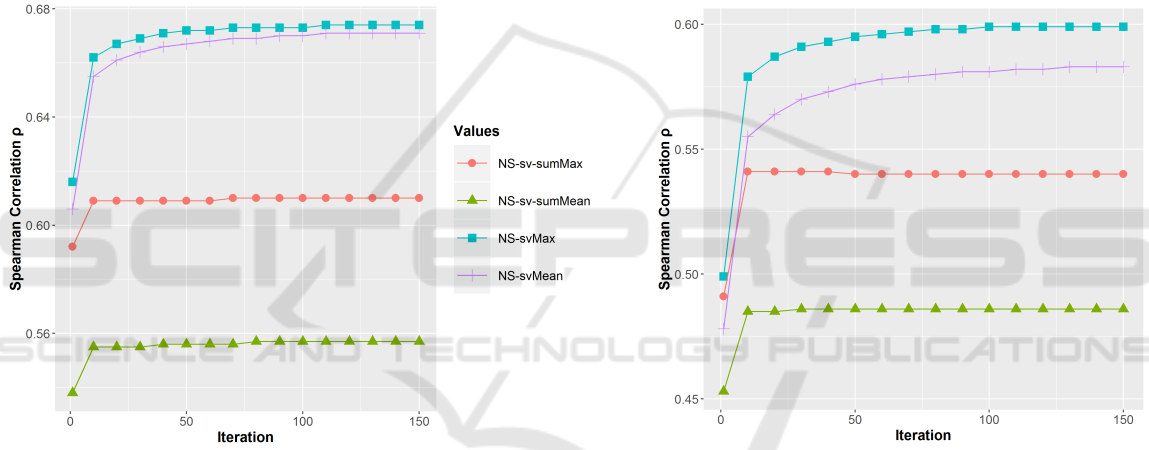
[10]https://github.com/y95847frank/GenSense

[11]https://github.com/codogogo/explirefit



(a) Simlex-999 (b) SimVerb-3500

Figure 2: Spearman's correlation on word similarity tasks in different vector dimensions using GloVe vectors (from 50 to 300 dimensions).



(a) SimLex-999 (b) SimVerb-3500

Figure 3: The influence of iteration on word similarity tasks using GloVe vectors (50 dimensions).

embedding models outperform the word embedding model GloVe and retrofittig model GenSense. GloVe as the baseline method is based on word level so its score is only the similarity between two words in the dataset. The metrics *MaxSimC* and *AveSimC* take the context of the target word into account which are generally used. However, for the trained sense vectors, it is hard to manage the polysemous context word embeddings of a target word. We inexactly use the average sense vectors of each context word. This limits the accuracy of our models. Furthermore, from Table 5, we find that *MaxSimC* is superior to *AveSimC*. And for context word vectors, the sum and the average of the vectors have little difference on discriminating the distinct senses.

5.4 Comparison of Word Vector Length

We tested our model and prior works with different dimensions, using GloVe word vectors involving 400K vocabularies. Figure 2 illustrates the different upward trend. The correlation ρ of each model rises continuously and stably from 50 to 300 on Simlex-999 task. On SimVerb-3500 task, the correlation ρ also increases with increasing word vector length, but nearly keeps steady after 100 dimensions. NS-sv performs well in all experimental word vector lengths and the gap with NS-sv-sum decreases constantly. We have reason to believe that with the word vectors of which the length is long enough, the performance of SV-sv and SV-sv-sum should tend to be constant. However, 300-dimensional vectors are the most common used because of their high accuracy and acceptable

data size, so we did not try the vectors with more than 300 dimensions which is barely used in practice. It is worth mentioning that even with 50 dimensions the sense vector generated by our model can still be highly accurate, suggesting to use the low dimensional vectors for applications instead of training the large word vectors with much resource consumption.

5.5 The Influence of Iteration

Figure 3 shows the Spearman's correlation score generated by our model for different number of iteration. Convergence is the only standard in our experiments to stop iteration. And according to the experiment of iteration, we finally decided to select $iter = 100$ as the default iteration for NS-sv model and $iter = 10$ for the default setup of NS-sv-sum model. From Figure 3, we can see both NS-sv and NS-sv-sum have obvious improvements from 1 to 10 iterations. And compared with NS-sv, NS-sv-sum approaches convergence already after only few iterations. However, its performance is not as good as the one of NS-sv in this case, mainly due to the use of only 50-dimensional vectors. The experiment of vector length revealed that NS-sv and NS-sv-sum will perform probably both great on the vectors with higher dimension, while with low dimension the integration among the synonyms may lose features, negatively affecting the final outcome. When all synonyms contribute together, it approaches convergence faster, which is the reason why NS-sv-sum converges faster while NS-sv needs more time.

6 CONCLUSION

In this paper, we presented a technique to retrofit word vectors from the word level to the more fine-grained level of word sense by two strategies named NS-sv and NS-sv-sum in this paper. This technique only employs primary semantic knowledge to improve the performance of the pre-trained word vectors and partitions the meaning of words into multiple senses. This is a post-processing approach, which avoids retraining on a large corpus, and can be applied on any pre-trained word vectors. The retrofitting procedure does not have many hyperparameters, yielding steady and efficient results in a multitude of application scenarios without costly hyperparameter optimization.

Our model is proposed for semantic similarity of words. It consists of four parts: (1) We provide the synonym and antonym sets of the most commonly used words in Wiktionary and use them as training examples for our negative-sampling-based neural network model; (2) we take advantage of the semantic knowledge from external resources (the synonym and antonym sets) to refine the word representations and construct sense representations; (3) our model significantly reduces the number of hyperparameters and, based on experiments, the default parameters do typically not have to be modified even if the input and the environment change; (4) the retrofitted word vectors can achieve the highest score 0.765 on SimLex-999. Our experiments have provided evidence for the effectiveness of our model on word similarity tasks. It outperforms the baseline methods and the previous work on three popular types of pre-trained word vectors of commonly used dimensions.

ACKNOWLEDGEMENTS

Rui Zhang was supported by the China Scholarship Council for 4 years of study at the University of Southern Denmark.

REFERENCES

Athiwaratkun, B. and Wilson, A. G. (2017). Multimodal word distributions. *arXiv preprint arXiv:1704.08424*.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL '98, August 10-14, 1998, Université de Montréal, Montréal, Quebec, Canada. Proceedings of the Conference.*, pages 86–90.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). Dbpedia-a crystallization point for the web of data. *Web Semantics: science, services and agents on the world wide web*, 7(3):154–165.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *TACL*, 5:135–146.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Bruni, E., Tran, N., and Baroni, M. (2014). Multimodal distributional semantics. *J. Artif. Intell. Res.*, 49:1–47.

Camacho-Collados, J. and Pilehvar, M. T. (2018). From word to sense embeddings: A survey on vector representations of meaning. *CoRR*, abs/1805.04032.

Camacho-Collados, J., Pilehvar, M. T., and Navigli, R. (2016). Nasari: Integrating explicit knowledge and

corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence*, 240:36–64.

Chang, K., Yih, W., and Meek, C. (2013). Multi-relational latent semantic analysis. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1602–1612.

Chen, T., Xu, R., He, Y., and Wang, X. (2015). Improving distributed representation of word sense via wordnet gloss composition and context clustering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 15–20.

Erk, K. and Padó, S. (2008). A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 897–906.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E. H., and Smith, N. A. (2015). Retrofitting word vectors to semantic lexicons. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1606–1615.

Faruqui, M., Tsvetkov, Y., Rastogi, P., and Dyer, C. (2016). Problems with evaluation of word embeddings using word similarity tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 30–35, Berlin, Germany. Association for Computational Linguistics.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2002). Placing search in context: the concept revisited. *ACM Trans. Inf. Syst.*, 20(1):116–131.

Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Gerz, D., Vulic, I., Hill, F., Reichart, R., and Korhonen, A. (2016). Simverb-3500: A large-scale evaluation set of verb similarity. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2173–2182.

Goikoetxea, J., Soroa, A., and Agirre, E. (2015). Random walks and neural network language models on knowledge bases. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1434–1439.

Grave, E., Mikolov, T., Joulin, A., and Bojanowski, P. (2017). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 427–431.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hill, F., Reichart, R., and Korhonen, A. (2015). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

Kiela, D., Hill, F., and Clark, S. (2015). Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 2044–2048.

Lee, Y., Yen, T., Huang, H., Shiue, Y., and Chen, H. (2018). Gensense: A generalized sense retrofitting model. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1662–1671.

Lengerich, B. J., Maas, A. L., and Potts, C. (2018). Retrofitting distributional embeddings to knowledge graphs with functional relations. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 2423–2436.

Li, J. and Jurafsky, D. (2015). Do multi-sense embeddings improve natural language understanding? *arXiv preprint arXiv:1506.01070*.

Li, W. and McCallum, A. (2005). Semi-supervised sequence modeling with syntactic topic models. In *AAAI*, volume 5, pages 813–818.

Liu, Q., Jiang, H., Wei, S., Ling, Z., and Hu, Y. (2015a). Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1501–1511.

Liu, Y., Liu, Z., Chua, T.-S., and Sun, M. (2015b). Topical word embeddings. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

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

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. (2018). Advances in pre-training distributed

word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

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 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119.

Navigli, R. and Ponzetto, S. P. (2012). Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2015). Efficient non-parametric estimation of multiple embeddings per word in vector space. *CoRR*, abs/1504.06654.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Pilehvar, M. T. and Collier, N. (2016). De-conflated semantic representations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1680–1690.

Qiu, L., Tu, K., and Yu, Y. (2016). Context-dependent sense embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 183–191.

Rastogi, P., Van Durme, B., and Arora, R. (2015). Multiview lsa: Representation learning via generalized cca. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 556–566, Denver, Colorado. Association for Computational Linguistics.

Recski, G., Iklódi, E., Pajkossy, K., and Kornai, A. (2016). Measuring semantic similarity of words using concept networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 193–200.

Řehůřek, R. and Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Reisinger, J. and Mooney, R. J. (2010). Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics.

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Schütze, H. (1998). Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.

Shiue, Y. and Ma, W. (2017). Improving word and sense embedding with hierarchical semantic relations. In *2017 International Conference on Asian Language Processing, IALP 2017, Singapore, December 5-7, 2017*, pages 350–353.

Soares, V. H. A., Campello, R. J. G. B., Nourashrafeddin, S., Milios, E., and Naldi, M. C. (2019). Combining semantic and term frequency similarities for text clustering. *Knowledge and Information Systems*.

Sujono, D. (2015). word2vecpy. https://github.com/deborausujono/word2vecpy.

Van de Cruys, T., Poibeau, T., and Korhonen, A. (2011). Latent vector weighting for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1012–1022. Association for Computational Linguistics.

Vasilescu, F., Langlais, P., and Lapalme, G. (2004). Evaluating variants of the lesk approach for disambiguating words. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*.

Vulic, I. and Glavas, G. (2018). Explicit retrofitting of distributional word vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 34–45.

Well, A. D. and Myers, J. L. (2003). *Research design & statistical analysis*. Psychology Press.

Wieting, J., Bansal, M., Gimpel, K., and Livescu, K. (2016). Charagram: Embedding words and sentences via character n-grams. *arXiv preprint arXiv:1607.02789*.

Yang, X. and Mao, K. (2016). Learning multi-prototype word embedding from single-prototype word embedding with integrated knowledge. *Expert Syst. Appl.*, 56:291–299.

Yu, L., Wang, J., Lai, K. R., and Zhang, X. (2017). Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 534–539.

Yu, M. and Dredze, M. (2014). Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 545–550.