Retrofitting Word Representations for Unsupervised Sense Aware Word Similarities

Steffen Remus and **Chris Biemann**
Universität Hamburg
Hamburg, Germany
{remus,biemann}@informatik.uni-hamburg.de

Abstract

Standard word embeddings lack the possibility to distinguish senses of a word by projecting them to exactly one vector. This has a negative effect particularly when computing similarity scores between words using standard vector-based similarity measures such as cosine similarity. We argue that minor senses play an important role in word similarity computations, hence we use an unsupervised sense inventory resource to retrofit monolingual word embeddings, producing sense-aware embeddings. Using retrofitted sense-aware embeddings, we show improved word similarity and relatedness results on multiple word embeddings and multiple established word similarity tasks, sometimes up to an impressive margin of $+0.15$ Spearman correlation score.
Keywords: word senses, word similarity and relatedness, word sense induction

1. Introduction

Word embeddings – generated with neural networks (NN) or other factorization techniques – are a standard element in natural language processing (NLP) applications. However, an important issue is their lack of sense-awareness, i.e. a word and its vector share a bijective mapping and a potential multiplicity of word meanings is ignored. The word *iron*, for example, which may refer to an atomic element, a device for smoothing clothes, a golf club, a color, or other meanings, is represented by a single common vector in the vector space. WORDNET (Fellbaum, 1998), on the other hand, defines four different interpretations of the word *iron*, and even this can never be considered to be complete as language evolves. Assigning the same vector for each distinguished sense and using them in downstream tasks such as sentiment analysis, named entity recognition, question answering or many others, is error prone by design due to obvious misinterpretations and error propagation.

Sense inventories – of which WORDNET is probably the most well known – are required to distinguish between different word senses, and meanings, rather than words, should be represented in the vector space (Navigli, 2009; Denkowski and Lavie, 2014). We use a simple, yet effective technique to retrofit standard word embeddings to produce embedding vectors of senses using external resources as sense inventories. Our hypothesis is that retrofitting pretrained word embeddings to gain sense-aware embeddings is beneficial for word similarity computations. Using vectors of senses rather that vectors of words, we are indeed able to report substantial relative improvements for multiple word similarity tasks and for various types of retrofitted embeddings from five monolingual corpora.

Because a word maps to multiple sense vectors in this scenario, standard cosine similarity computations alone are not applicable anymore, we thus test a number of sense-aware comparison methodologies based on cosine similarity. In particular for word pairs involving minor/rare senses, we expect improvements in the sense-aware setting as the influence of the dominating major sense is diminished. Additionally, we compare our approach with two baseline approaches to supervised and unsupervised sense-aware embeddings: AUTOEXTEND (Rothe and Schütze, 2015) and ADAGRAM (Bartunov et al., 2016).

To the best of our knowledge, we are the first to employ unsupervised word sense induction techniques for retrofitting single word vectors to the multiplicity of their meanings, creating new pseudo word-sense vectors, and using those for semantic similarity. Additionally, we test standard word sense induction (WSI) techniques using word embeddings themselves in order to make the retrofitting process self-sustained. Evidence presented below indicates that word embeddings are hardly useful in word sense induction clustering, due to the fact that their neighborhoods largely consist of words referring to the dominant sense in the source corpus.

2. Related Work

A number of word similarity benchmarks exist in order to intrinsically test the semantic properties of word embeddings (Hill et al., 2014; Finkelstein et al., 2001; Bruni et al., 2014; Gerz et al., 2016). Similarities are usually computed by means of cosine similarity between two vectors, which are representations of words in an embedded vector space.

The history of word embeddings is vast, ranging from geometrical matrix factorization methods like latent semantic analysis (Landauer and Dumais, 1997, LSA) or principal component analysis (PCA), over to probabilistic topic models such as probabilistic latent semantic analysis (Hofmann, 1999, PLSA) or latent dirichlet allocation (Blei et al., 2003, LDA), to more recent approaches based on neural network (NN) architectures such as skip-gram negative-sampling (SGNS), continuous bag of words (CBOW), or global vectors (Pennington et al., 2014, GLOVE), from which the former two are both available in the WORD2VEC toolkit (Mikolov et al., 2013). In this paper, we mainly focus on embeddings generated by NNs because of their superior performance and current impact on NLP research. However, we note that our findings are also applicable to other types of embedded word vector spaces, as we shall see below.

Rothe and Schütze (2015) introduced AUTOEXTEND, a supervised neural network model which enriches existing embeddings with word sense information from WORDNET or other sense inventories.[1] Here, the sense inventory is taken from WORDNET but Rothe and Schütze (2015) emphasize that any lexical or semantic resource could be used. Neelakantan et al. (2014) and Bartunov et al. (2016) present approaches that gather sense information in an unsupervised way from monolingual text by integrating the sense distinction into the learning process. We use ADAGRAM (Bartunov et al., 2016) as an additional baseline because it compares favorably to the model by Neelakantan et al. (2014). ADAGRAM's main parameter effectively regulates the maximum number of senses per word; the algorithm finds the number of senses automatically in this range, i.e. the parameter can be seen as a limit for the maximum number of induced senses.

Retrofitting is the process of augmenting a given item for a new task, i.e. in our case a post-processing objective that re-adjusts existing word embeddings (Faruqui et al., 2015). Multiple objectives have been defined on this account, e.g. Faruqui et al. (2015) or Kiela et al. (2015) use lexical resources, while, for instance, Wieting et al. (2015) directly optimizes paraphrase pair alignment from PPDB[2] (Ganitkevitch et al., 2013).

3. Methodology

In the remainder of this work we will use v to refer to a word and $\mathbf{v}$ to refer to v's corresponding word vector.

3.1. Unsupervised Sense Inventory

Our proposed method solely relies on pre-computed word-embeddings and a sense inventory resource. We follow the terminology in WORDNET and define a *synset* for a word v to be the set of related words that express the same concept, and the *sense inventory* of v to be the *collection of its synsets*, i.e. the different senses v can bear. Dorow and Widdows (2003), Pantel and Lin (2002), and more recently, Pelevina et al. (2016) use unsupervised WSI methods, which means they use or create so-called *unsupervised synsets* referring to sense-inventories, which were induced automatically from text. The simplified procedure to compute an unsupervised synset for a particular word v is as follows:

1. compute v's **top** n **nearest neighbors** (by some word-similarity notion)
2. compute a similarity score between every pairwise combination of nearest neighbors, which renders a fully connected **similarity graph**
3. compute a word **clustering**, where each cluster represents a different sense of v.

This general methodology has been proven to perform sufficiently well on a number of NLP tasks, whereas the details of this simplified procedure vary. The clustering represents the sense inventory (i.e. the collection of synsets) S_v for the word v; we refer to a particular synset or sense k of v as S_v^k. We want to stress that v is usually not contained in any of its "synsets", i.e. $S_v^k = S_v^k \setminus v$ per definition.

Following Riedl and Biemann (2017), we use an unsupervised sense inventory, pre-computed[3] by using the JOBIMTEXT (JBT) framework (Biemann and Riedl, 2013; Riedl, 2016), which can be seen as a symbolic count based model. JBT provides a graph-based sparse word similarity model, i.e. only words, and no vectors are provided. The Chinese Whispers (Biemann, 2006, CW) algorithm is used for inducing word senses based on ego networks weighted by context similarity.

3.2. Retrofitting Word Embeddings

The main goal of retrofitting word vectors is to find individual vector representations for each sense of a word. Using a sense inventory, word vectors from a particular synset are averaged, such that each sense of a word will be represented by a single individual vector. For a word v, we average all vectors of the top m words in a synset S_v^k and add the vector $\mathbf{v}$ with weight λ in order to compensate for semantic drift, for which we found strong indications in preliminary experiments:

$$\mathbf{v}_k = \lambda \mathbf{v} + (1-\lambda) \sum_{u \in \text{top}_m(S_v^k)} \mathbf{u}\,, \tag{1}$$

where λ is a scalar in $[0,1]$, $\mathbf{v}_k$ is the sense vector of the k^{th} sense of word v, and $\mathbf{u}$ is the word vector of word u. A geometric interpretation of this equation can be interpreted as to first find the center of a cluster of words in S_v^k and then shift the center by λ into the direction of the core word v. Note again that the clustering itself for any word v is performed without v itself, i.e. it is not contained S_v^k, cf. (Dorow and Widdows, 2003), hence the shifting. Using only the top m words stems from the fact that the clusters, i.e. the synsets, have different sizes. A common observation is that the largest clusters often refer to major senses and smaller clusters usually represent minor senses, i.e. senses that are underrepresented in the text corpus. To alleviate the effect of averaging noisy words in large clusters, we select only the top m words.

3.3. Sense-aware word-similarity

We tested different procedures for computing sense-aware similarities between any two words u and v with senses u_k and v_l:

$$\text{sim}(u,v) = \arg\max_k \ \cos(\mathbf{u}_k, \mathbf{v}) \tag{2}$$

$$\text{sim}(u,v) = \arg\max_l \ \cos(\mathbf{u}, \mathbf{v}_l) \tag{3}$$

$$\text{sim}(u,v) = \arg\max_{k,\,l} \ \cos(\mathbf{u}_k, \mathbf{v}_l)\,. \tag{4}$$

Equations (2-4) involve finding the closest senses k and l for the words u and v in vector space. We compare these measures to the standard, sense un-aware cosine similarity $\cos(\mathbf{u}, \mathbf{v})$.

[1] http://www.cis.lmu.de/~sascha/AutoExtend/

[2] The paraphrase database: http://www.cis.upenn.edu/~ccb/ppdb/.

[3] http://ltmaggie.informatik.uni-hamburg.de/jobimviz, (Ruppert et al., 2015)

4. Experimental Setup

4.1. Word-similarity Benchmark Datasets

Hill et al. (2014) raise the point that a strong distinction must be made between similarity and relatedness. While related words roughly fit into the same topic, similar words are more specific, they fit into the same topic and constitute (partial) substitutability. Consider for example the words *student* and *professor*, which are certainly considered related but not similar because there are only few contexts in which the two words can be exchanged, hence they are considered highly dissimilar due their antonymic nature while *teacher* and *professor* might be exchangeable, and are thus considered equally related but more similar.

The WORDSIM353 (Finkelstein et al., 2001) dataset provides relatedness scores of 353 noun pairs and the SIMLEX999 dataset provides similarity scores for 666 noun pairs, 222 verb pairs and 111 adjective pairs. Particularly the latter's emphasis is to model opposite meanings (antonym-like) as highly non-similar, e.g. *student* and *professor* have a low similarity score in SIMLEX999, but a high relatedness score in WORDSIM353.

Another dataset is the MEN[4] dataset (Bruni et al., 2014), which models, analog to WORDSIM353, relatedness or association rather than similarity. Bruni et al. (2014) randomly sampled $3,000$ word pairs from words that occur at least 700 times in the ukWaC + Wackypedia combined corpora.[5] MEN comprises of inter part-of-speech word pairs, e.g. pairs like (*apple-N*, *orange-A*) or (*bear-V*, *boxer-N*). It is also worthy to note that MEN comes in two forms, *a*) in a lemmatized form with POS tags, and *b*) in natural form. We report results on the lemma form with POS-tag information.

Another dataset, the SIMVERB dataset (Gerz et al., 2016), can be interpreted as a larger version of the verb part of SIMLEX999, containing $3,500$ verb pairs, allowing more meaningful benchmarking with more and better represented examples.

4.2. Embedding Matrices

WORD2VEC applies a neural language modeling approach, where the goal is to predict a word w_i at position i given its context c_i (CBOW) or vice versa (SGNS) (Mikolov et al., 2013). A projection matrix is learned during this process, which has been shown to be beneficial in various NLP tasks. We use pre-trained word vectors provided by Mikolov et al. (2013), which were trained on Google News texts containing 6 Billion words.[6] Additionally, we use the GLOVE[7] (Pennington et al., 2014, global vectors) embeddings.

Schwartz et al. (2015) defined the context of a word to be the *symmetric pattern* it occurs with, and applied WORD2VEC to those pairs. A symmetric pattern is a shallow pattern in the form of *'X or Y'*, *'X and Y'*, *'X as well as Y'*, *'X rather than Y'*, where particular instances of *X* and *Y* occur in both positions, e.g. *'cats and dogs'* and *'dogs and cats'* are considered instances of a symmetric pattern, while *'point of view'* for example cannot be altered without losing its meaning, *'X of Y'* is thus considered an asymmetric pattern. Some symmetric patterns are considered to be particularly indicative for antonymy, e.g. *'either X or Y'* or *'rather X than Y'* are typical to be filled by words with opposite, or strongly different meanings, e.g. *'either black or white'*. Schwartz et al. (2015) used symmetric patterns to build an antonym-sensitive embedding model from monolingual corpora. We use their 10K dimension model built on an 8G words corpus[8], and refer to this embedding type as SYMPAT. We also tested the 300 and 500 dimensional vectors provided by Schwartz et al. (2015) but the 10K version achieved the best results among the SYMPAT embeddings.

Wieting et al. (2015) used PPDB pairs to train a projection matrix called PARAGRAM. The matrices are initialized with the GLOVE embeddings and retro-fitted to match with PPDB. By using paraphrases obtained via round-trip translations, the model is already guided to represent synonymous expressions with similar vectors, as opposed to expressions with opposite meanings. Wieting et al. (2015) further tuned the hyper-parameters, resulting in PARAGRAMWS optimized on WORDSIM353 and PARAGRAMSL optimized for SIMLEX999.

The embeddings are thus tuned for either relatedness or similarity and constitute a strong baseline.

Additionally, we also make use of two LSA embeddings trained on English corpora provided by Günther et al. (2015).[9] Both models are based on a 2-Billion-word corpus and use a *positive pointwise mutual information* weighting scheme (PPMI) before applying singular value decomposition (SVD) with 300 target dimensions and a vocabulary of 100K words. We refer to the model based on a bag-of-word representation of documents as LSABOW, and to the model applying a HAL-like context representation[10] as LSAHAL, following the terminology of Günther et al. (2015).

Many other NN embedding models have been published, e.g. (Wieting et al., 2016; Recski et al., 2016; Mrkšić et al., 2016), however, we deliberately do not go into details here since these supervised models are out of the scope of this work; we focus on the relative improvement of monolingual embeddings by exploiting unsupervised WSI methods. We are thus independent of any manually developed resource and do not rely on the existence of parallel text.

5. Results

We follow previous work and use the Spearman rank-correlation coefficient ρ throughout the evaluation. We evaluated all datasets for all methods but restrict our discussion to the most interesting results. Selecting the top $m = 5$ cluster words for averaging proved most useful; in

[4] https://staff.fnwi.uva.nl/e.bruni/MEN

[5] http://wacky.sslmit.unibo.it/

[6] We use the 300 dimensional model trained on Google news. The model and the source code is available at https://code.google.com/p/word2vec/.

[7] We use the 6 Billion word, 300 dimensional model available at http://nlp.stanford.edu/projects/glove/.

[8] http://homes.cs.washington.edu/~roysch/papers/sp_embeddings/sp_embeddings.html

[9] Models are available for download under http://www.lingexp.uni-tuebingen.de/z2/LSAspaces/.

[10] HAL, hyperspace analogue to language; a context representation similar to the skip-gram notion

	AUTOEXTEND	ADAGRAM	SGNS	SGNS-S	GLOVE	GLOVE-S	SYMPAT	SYMPAT-S	LSA_{BOW}	LSA_{BOW-S}	LSA_{HAL}	LSA_{HAL-S}	PARAGRAMSL	PARAGRAMSL-S	PARAGRAMWS	PARAGRAMWS-S
SIMLEX999	0.45	0.29	0.44	0.46	0.37	**0.41**	0.54	0.55	0.30	**0.39**	0.27	**0.38**	0.68	0.64	0.66	0.64
MEN	0.72	0.67	0.77	0.78	0.73	**0.77**	0.53	**0.68**	0.67	**0.70**	0.71	**0.74**	0.77	**0.80**	0.80	0.81
SIMVERB	0.43	0.27	0.36	**0.39**	0.23	**0.30**	0.37	**0.45**	0.15	**0.22**	0.19	**0.28**	0.53	0.53	0.51	0.50
WORDSIM353	0.58	0.61	0.70	0.69	0.61	**0.65**	0.47	**0.62**	0.67	0.66	0.59	**0.63**	0.72	0.73	0.77	0.75
SIMLEX999-N	0.44	0.33	0.45	**0.50**	0.39	**0.47**	0.48	**0.55**	0.32	**0.46**	0.34	**0.44**	0.68	0.66	0.64	0.64
MEN-N	0.72	0.68	0.77	0.79	0.76	**0.80**	0.57	**0.74**	0.71	0.73	0.73	**0.76**	0.78	**0.81**	0.80	0.82

Table 1: Spearman correlation scores on the different datasets and embeddings. Sense-aware similarities are marked with '-S'. The best performing method is underlined or marked bold. We distinguish underlined values to be the winning system with a slight margin (< 0.03) and bold face values with a larger margin. We marked PARAGRAMSL and PARAGRAMWS for SIMLEX999 and WORDSIM353 in gray, since the method's hyperparameters were optimized on the respective dataset, thus, the results are not comparable. The lower part evaluates only the noun pair parts of the datasets, as indicated by -N.

our experiments we found fluctuating best performing values between top 3 and top 10, with 5 always being among the best values. Also, Equation (4) distinguished itself as the best performing method with $\lambda = 0.5$. Other similarity computations, Eq. (2;3), perform non-satisfactory, sometimes even with a decline in performance. In the remainder of this work we refer to embeddings with the suffix -S to the sense-aware similarities which performed best in our previous experiments using the fixed parameters $m = 5$ and $\lambda = 0.5$.

We report AUTOEXTEND (Rothe and Schütze, 2015) and ADAGRAM (Bartunov et al., 2016) scores for comparison. Table 1 shows the final results using sense-unaware similarities, i.e. standard cosine similarity, and our new sense-aware similarities based on the JBT sense inventory.

The results clearly show that sense-aware similarities perform consistently better or comparable to their sense-unaware counterparts. The average improvement for most sense-unaware systems to their sense-aware counterpart is roughly between 0.02 and 0.05 points of spearman correlation. Particularly, previously inferior embeddings, e.g. GLOVE or both LSA embeddings, gain most and more consistent from this representation. The loss of performance with the PARAGRAM family of embeddings is mainly due to the fact that they already have been optimized for synonymy and antonymy. Injecting the JBT sense inventory – which has no special treatment for antonyms – attracts related terms, i.e. apparently antonymous, non-similar, but related words. In fact, this happens to a large extent on adjectives, causing the largest losses. When looking at the performance for nouns (lower part of Table 1 for datasets where nouns were available), we see consistent improvements across all datasets.

We observe minor sense selections in $3,953$ out of $7,734$ examples across all datasets for SGNS-S, that is roughly 52%. Summarizing, in about half of the example word pairs a minor sense was selected. This is most consistent across nouns, and varies for verbs and adjectives, which could be attributed to coverage issues[11], or inadequate clusterings for adjectives and verbs, since the JBT sense clustering mainly focusses on nouns.

For illustration of adequacy, consider the word pair (*iron, vitamin*) taken from the SIMLEX999 dataset. Figure 1 provides details for this example word pair, which includes all scores and a description of the induced sense inventory. We can see that the manually assigned SIMLEX999 score is in the mid-range (5.55 out of 10), standard cosine similarity ranks[12] this example at position 212 with a similarity score of 0.22, which is rather low. This can be verified by looking at the figure, i.e. on the innermost unit circle, the angle between *vitamin* and *iron* is quite large. The sense-aware similarity score selects a link between two suitable minor senses. The visualization shows the two words and their cluster terms, as well as the averaged cluster centers on the unit circle. The projection was done with T-SNE (Maaten and Hinton, 2008). For better illustration, we mapped cluster terms for each word on a different circle, but note that each circle preserves directions and represents a scaled unit circle. In this visualization, it is easily recognizable that the vectors for *iron* and *vitamin* are far apart, whereas the retrofitted vectors *iron2* and *vitamin3* are close by in terms of their cosine similarity.

We computed cross correlation scores between the methods, e.g. the Spearman correlation score between SGNS and SGNS-S embeddings yields $\rho = 0.85$. This suggests that the individual scores differ, although final SIMLEX999 correlation scores do not seem to benefit drastically (e.g. $+0.02$ difference for SGNS to SGNS-S).

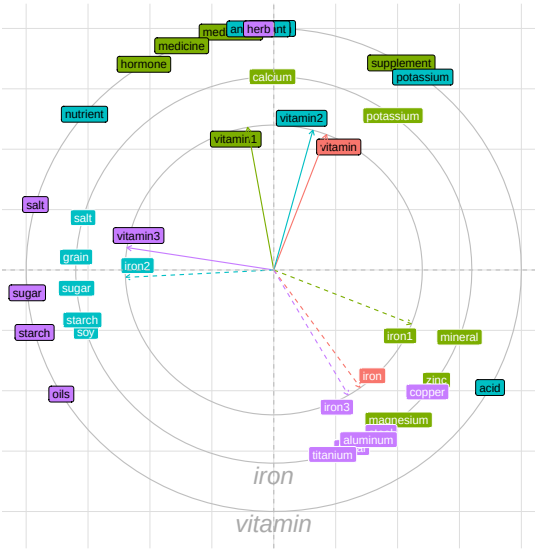
6. Native Sense Clustering

In order to make the retrofitting process independent of external (be it induced or manually compiled) sense inventories, we build a sense-inventory directly from word embeddings and provide anecdotal evidence and insights of its failure. For proof of concept we focus on a single NN embedding, for which we chose the SGNS word embedding matrix because of its popularity. We follow the general

[11] Coverage is around 98% for SYMPAT and 99% for others.

[12] Note that Spearman correlation compares ranks.

SimLex999 / rg(SimLex999)	5.55 / 622
SGNS / rg(SGNS)	0.22 / 212
SGNS-S / rg(SGNS-S)	0.59 / 488
$\lvert S^k_{\texttt{iron}} \rvert$	$\{88, \underline{5}, 98\}$
$\lvert S^k_{\texttt{vitamin}} \rvert$	$\{59, 88, \underline{53}\}$



$S_{\texttt{iron}} = \{ \{\texttt{zinc}, \texttt{calcium}, \texttt{magnesium}, \texttt{potassium}, \texttt{mineral}\}, \underline{\{\texttt{sugar}, \texttt{salt}, \texttt{grain}, \texttt{soy}, \texttt{starch}\}}, \{\texttt{steel}, \texttt{metal}, \texttt{copper}, \texttt{aluminum}, \texttt{titanium}\} \}$

$S_{\texttt{vitamin}} = \{ \{\texttt{supplement}, \texttt{hormone}, \texttt{pill}, \texttt{medication}, \texttt{medicine}\}, \{\texttt{nutrient}, \texttt{calcium}, \texttt{antioxidant}, \texttt{acid}, \texttt{potassium}\}, \underline{\{\texttt{sugar}, \texttt{salt}, \texttt{oils}, \texttt{starch}, \texttt{herb}\}} \}$

Figure 1: Scores achieved by sense-aware and sense-unaware word similarity computation for the word pair (*iron, vitamin*). rg(·) refers to the rank regarding the SimLex999 dataset; selected senses by the method are underlined. The visualization is based on terms on the unit circle. Every circle represents a unit circle in a joint plot for illustration purposes. The inner circle shows the different sense vectors as well as the original word vectors, the middle circle shows synset terms generated by the word *iron*, and the outer circle represents synset terms generated by the word *vitamin*.

procedure to build sense inventories as explained in Section 3.1., i.e. for a particular word v, we first create a similarity matrix M of the top k nearest neighbors[13] in terms of cosine similarity, we then applied a clustering algorithm to the similarity matrix M, which yields a clustering of words that can be directly interpreted as the sense-inventory for v: S_v.

[13]For proof of concept, we used $k = 500$, which is commonly known to be reasonable value.

	SGNS		JBT		
related term	cos	rg(·)	#ctx	rg(·)	sense description
putter	0.46	17	36	128	golf sports
wood	0.47	11	119	15	
copper	0.37	252	206	9	metallic elements
aluminum	0.35	427	206	8	
salt	0.23	23731	31	158	nutrition
fiber	0.20	47072	77	38	
steam	0.12	416270	28	181	smoothing clothes
shirt	0.12	415080	–	–	

Table 2: Cosine similarity (cos) and similarity by number of shared contexts (#ctx), next to the relative rank regarding cos for SNGS and #ctx for JBT with respect to the query word '*iron*'.

Since words cannot be expected to have a fixed number of senses, we tested two *graph based clustering algorithms*, where the number of clusters, i.e. the number of senses, is not a parameter but will be determined by the algorithms themselves. Because of its symmetry, M can be directly interpreted as an adjacency matrix for an undirected graph. We experimented with the following graph clustering algorithms: 1.) CW: Chinese Whispers (Biemann, 2006); and 2.) MCL: Markov Clustering (van Dongen, 2000).[14] In general, graph clustering algorithms perform best if the adjacency matrix of the graph is sparse. In order to sparsify M, we prune by a threshold parameter τ_{cos}, i.e. we set values $M_{ij} = 0$ if M_{ij} is lower than τ_{cos}. Apart from that, we use the default parameter settings suggested by Biemann (2006) for CW or van Dongen (2000) for MCL. As a post-clustering step, singleton clusters are merged into one 'residual' cluster, i.e. clusters which contain only a single element – which occur frequently for large τ_{cos} – eventually form the 'residual' sense.

Results by Anecdotal Evidence: Cleary, the parameter k, which defines the top k nearest neighbors of a word v, and thus the size of M, implicitly also controls the vocabulary of the sense inventory of v. Manual inspection of those nearest terms revealed, that in case of SGNS and other NN word embeddings, the immediate neighborhood of a word v consists mainly of one dominating sense. For illustrative purposes, consider the example given in Table 2, where we highlight scores and ranks for the polysemous word *iron* with regards to some hand-selected words representing different senses of *iron*. Here, mainly terms referring to a *golf sports* related sense can be found in the immediate vicinity of *iron* (large cosine similarity, small rank), while other terms referring to common senses are

[14]Note that we also tested other clustering algorithms, such as *K-Means* and *Self-Organizing-Maps* for comparison purposes, but we report results only for CW and MCL since they do not depend on the number of clusters as parameter input and yield visually better clusters.

	τ_{cos}						
	0.3	0.4	0.5	0.6	0.7	0.8	0.9
MCL	1.7	13.5	34.5	50.4	38.5	10.2	2.4
CW	1.4	3.0	7.9	16.2	15.0	4.0	1.5

Table 3: Average number of clusters for all words in all datasets, varying $\tau_{\cos}$, the two graph clustering algorithms, based on SGNS embedding vectors.

	SIMLEX999
SGNS-S$_{clnat}$	0.42
SYMPAT-S$_{clnat}$	0.48
PARAGRAMSL-S$_{clnat}$	0.67
PARAGRAMWS-S$_{clnat}$	0.57

Table 4: Selected results for native sense induction by clustering on four embeddings and SIMLEX999.

farther away (small cosine similarity, large rank). The selected terms for representative senses seem to have a similar cosine similarity to $iron$ though. This is not an isolated incident, we have observed this effect consistently for multiple polysemous terms.[15] This suggests the confirmation of the observations which Faruqui et al. (2016) or Schnabel et al. (2015) already noted: Within neural word embeddings, the frequency rank of a word's neighbor strongly depends on the frequency of the word itself. This is clearly an issue because the frequency of a word's sense naturally correlates with the frequency of a word's occurrence.

Table 3 shows the average number of clusters for all words across all datasets for varying τ_{cos}. Based on those results, we fix $\tau_{cos} = 0.8$ and CW, as this best resembles the sense inventory of the JBT resource, where also CW is used, producing 3.73 senses on average. Selected results of the native clustering compared to SIMLEX999 in Table 4 show a decline in performance w.r.t. the sense-unaware similarity values in Table 1. Failure can be attributed to the local structure of the neighborhood as explained above.

7. Conclusion

We confirmed our initial hypothesis that 'sense inventories do help for word similarity' and presented consistent improvements over all tested embeddings and datasets using pre-existing sense-inventory resources. This holds particularly for embeddings trained on monolingual text. On a general level, we have shown how to operationalize word sense induction for a semantic task, here for word similarity, by creating appropriate representations of words for the task on top of generic, previously available, representations. Contrary to most prior work in this area, we did not use manually-defined sense inventories or lexical resources, but an unsupervised graph-based sense induction scheme. Additionally, we confirmed prior findings and conclude that direct clustering of a word's nearest neighbors in an NN embedding is not helpful for WSI, but other methodologies are required here. The source code as well as the sense aware vectors for the datasets are provided as open source software under a permissive license.[16] We would like to follow up on this line of work and devise similar schemes for relation extraction, learning of semantic hierarchies, and short text similarity.

[15] We leave a more thorough analysis for future work.

[16] https://github.com/uhh-lt/senseasim

8. References

Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. (2016). Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 130–138, Cadiz, Spain.

Biemann, C. and Riedl, M. (2013). Text: Now in 2D! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.

Biemann, C. (2006). Chinese whispers - an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing*, pages 73–80, NY, USA.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bruni, E., Tran, N. K., and Baroni, M. (2014). Multimodal Distributional Semantics. *Journal of Artificial Intelligence Research*, 49(1):1–47.

Denkowski, M. and Lavie, A. (2014). Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, MD, USA.

Dorow, B. and Widdows, D. (2003). Discovering corpus-specific word senses. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics*, pages 79–82, Budapest, Hungary.

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

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.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Language, Speech, and Communication. The MIT Press.

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

Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, GA, USA.

Gerz, D., Vulić, 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*, pages 2173–2182, Austin, TX, USA.

Günther, F., Dudschig, C., and Kaup, B. (2015). LSAfun - An R package for computations based on Latent Semantic Analysis. *Behavior Research Methods*, 47(4):930–944.

Hill, F., Reichart, R., and Korhonen, A. (2014). SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*.

Hofmann, T. (1999). Probabilistic Latent Semantic Analysis. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 289–296, Stockholm, Sweden.

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*, pages 2044–2048, Lisbon, Portugal.

Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato's problem: the Latent Semantic Analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using {t-SNE}. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the International Conference on Machine Learning*, ICLR 2013, pages 1310–1318, Scottsdale, AZ, USA.

Mrkšić, N., Ó Séaghdha, D., Thomson, B., Gašić, M., Rojas-Barahona, L. M., Su, P.-H., Vandyke, D., Wen, T.-H., and Young, S. (2016). Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148, San Diego, CA, USA.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10:1–10:69.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2014). Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar.

Pantel, P. and Lin, D. (2002). Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD 2002, pages 613–619, Edmonton, AL, Canada.

Pelevina, M., Arefiev, N., Biemann, C., and Panchenko, A. (2016). Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin, Germany.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

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, Berlin, Germany.

Riedl, M. and Biemann, C. (2017). There's no 'count or predict' but task-based selection for distributional models. In *Proceedings of the 12th International Conference on Computational Semantics (IWCS)*, Montpellier, France.

Riedl, M. (2016). *Unsupervised Methods for Learning and Using Semantics of Natural Language*. Ph.D. thesis, Technische Universität, Darmstadt.

Rothe, S. and Schütze, H. (2015). Autoextend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1793–1803, Beijing, China.

Ruppert, E., Kaufmann, M., Riedl, M., and Biemann, C. (2015). Jobimviz: A web-based visualization for graph-based distributional semantic models. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 103–108, Beijing, China.

Schnabel, T., Labutov, I., Mimno, D., and Joachims, T. (2015). Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, Lisbon, Portugal.

Schwartz, R., Reichart, R., and Rappoport, A. (2015). Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 258–267, Beijing, China.

van Dongen, S. M. (2000). *Graph Clustering by Flow Simulation*. Ph.D. thesis, Universiteit Utrecht.

Wieting, J., Bansal, M., Gimpel, K., and Livescu, K. (2015). From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*, 3:345–358.

Wieting, J., Bansal, M., Gimpel, K., and Livescu, K. (2016). Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1515, Austin, TX, USA.