Embeddings for Word Sense Disambiguation: An Evaluation Study

Ignacio Iacobacci[1], **Mohammad Taher Pilehvar**[2] **and Roberto Navigli**[1]
[1]Department of Computer Science, Sapienza University of Rome, Italy
[2]Language Technology Lab, University of Cambridge, UK
{iacobacci,navigli}@di.uniroma1.it
mp792@cam.ac.uk

Abstract

Recent years have seen a dramatic growth in the popularity of word embeddings mainly owing to their ability to capture semantic information from massive amounts of textual content. As a result, many tasks in Natural Language Processing have tried to take advantage of the potential of these distributional models. In this work, we study how word embeddings can be used in Word Sense Disambiguation, one of the oldest tasks in Natural Language Processing and Artificial Intelligence. We propose different methods through which word embeddings can be leveraged in a state-of-the-art supervised WSD system architecture, and perform a deep analysis of how different parameters affect performance. We show how a WSD system that makes use of word embeddings alone, if designed properly, can provide significant performance improvement over a state-of-the-art WSD system that incorporates several standard WSD features.

1 Introduction

Embeddings represent words, or concepts in a low-dimensional continuous space. These vectors capture useful syntactic and semantic information, such as regularities in language, where relationships are characterized by a relation-specific vector offset. The ability of embeddings to capture knowledge has been exploited in several tasks, such as Machine Translation (Mikolov et al., 2013, MT), Sentiment Analysis (Socher et al., 2013), Word Sense Disambiguation (Chen et al., 2014, WSD) and Language Understanding (Mesnil et al., 2013). Supervised WSD is based on the hypothesis that contextual information provides a good approximation to word meaning, as suggested by Miller and Charles (1991): semantically similar words tend to have similar contextual distributions.

Recently, there have been efforts on leveraging embeddings for improving supervised WSD systems. Taghipour and Ng (2015) showed that the performance of conventional supervised WSD systems can be increased by taking advantage of embeddings as new features. In the same direction, Rothe and Schütze (2015) trained embeddings by mixing words, lexemes and synsets, and introducing a set of features based on calculations on the resulting representations. However, none of these techniques takes full advantage of the semantic information contained in embeddings. As a result, they generally fail in providing substantial improvements in WSD performance.

In this paper, we provide for the first time a study of different techniques for taking advantage of the combination of embeddings with standard WSD features. We also propose an effective approach for leveraging embeddings in WSD, and show that this can provide significant improvement on multiple standard benchmarks.

2 Word Embeddings

An embedding is a representation of a topological object, such as a manifold, graph, or field, in a certain space in such a way that its connectivity or algebraic properties are preserved (Insall et al., 2015). Presented originally by Bengio et al. (2003), word embeddings aim at representing, i.e., embedding, the ideal semantic space of words in a real-valued continuous vector space. In contrast to traditional distributional techniques, such as Latent Semantic Analysis (Landauer and Dutnais, 1997, LSA) and Latent Dirichlet Allocation (Blei et al., 2003, LDA), Bengio et al. (2003) designed a feed-forward neural network capable of predicting a word given the words preceding (i.e., leading up to) that word. Collobert and Weston (2008) presented a much deeper model consisting of several layers for feature extraction, with the objective of building a general architecture for NLP tasks. A major breakthrough occurred when Mikolov et al. (2013) put forward an efficient algorithm for training embeddings, known as Word2vec. A similar model to Word2vec was presented by Pennington et al. (2014, GloVe), but instead of using latent features for representing words, it makes an explicit representation produced from statistical calculation on word countings.

Numerous efforts have been made to improve different aspects of word embeddings. One way to enhance embeddings is to represent more fine-grained semantic items, such as word senses or concepts, given that conventional embeddings conflate different meanings of a word into a single representation. Several research studies have investigated the representation of word senses, instead of words (Reisinger and Mooney, 2010; Huang et al., 2012; Camacho-Collados et al., 2015b; Iacobacci et al., 2015; Rothe and Schütze, 2015). Another path of research is aimed at refining word embeddings on the basis of additional information from other knowledge resources (Faruqui et al., 2015; Yu and Dredze, 2014). A good example of this latter approach is that proposed by Faruqui et al. (2015), which improves pre-trained word embeddings by exploiting the semantic knowledge from resources such as PPDB[1] (Ganitkevitch et al., 2013), WordNet (Miller, 1995) and FrameNet (Baker et al., 1998). In the following section we discuss how embeddings can be integrated into an important lexical semantic task, i.e., Word Sense Disambiguation.

3 Word Sense Disambiguation

Natural language is inherently ambiguous. Most commonly-used words have several meanings. In order to identify the intended meaning of a word one has to analyze the context in which it appears by directly exploiting information from raw texts. The task of automatically assigning predefined meanings to words in contexts, known as Word Sense Disambiguation, is a fundamental task in computational lexical semantics (Navigli, 2009). There are four conventional approaches to WSD which we briefly explain in the following.

3.1 Supervised methods

These methods make use of manually sense-annotated data, which are curated by human experts. They are based on the assumption that a word's context can provide enough evidence for its disambiguation. Since manual sense annotation is a difficult and time-consuming process, something known as the ”knowledge acquisition bottleneck” (Pilehvar and Navigli, 2014), supervised methods are not scalable and they require repetition of a comparable effort for each new language. Currently, the best performing WSD systems are those based on supervised learning. It Makes Sense (Zhong and Ng, 2010, IMS) and the system of Shen et al. (2013) are good representatives for this category of systems. We provide more information on IMS in Section 4.1.

3.2 Unsupervised methods

These methods create their own annotated corpus. The underlying assumption is that similar senses occur in similar contexts, therefore it is possible to group word usages according to their shared meaning and induce senses. These methods lead to the difficulty of mapping their induced senses into a sense inventory and they still require manual intervention in order to perform such mapping. Examples of this approach were studied by Agirre et al. (2006), Brody and Lapata (2009), Manandhar et al. (2010), Van de Cruys and Apidianaki (2011) and Di Marco and Navigli (2013).

3.3 Semi-supervised methods

Other methods, called semi-supervised, take a middle-ground approach. Here, a small manually-annotated corpus is usually used as a seed for bootstrapping a larger annotated corpus. Examples of these approaches were presented by Mihalcea and Faruque (2004). A second option is to use a word-aligned bilingual corpus approach, based on the assumption that an ambiguous word in one language could be unambiguous in the context of a second language, hence helping to annotate the sense in the first language (Ng and Lee, 1996).

3.4 Knowledge-based methods

These methods are based on existing lexical resources, such as knowledge bases, semantic networks, dictionaries and thesauri. Their main feature is their coverage, since they function indepen-

[1]www.paraphrase.org/#/download

dently of annotated data and can exploit the graph structure of semantic networks to identify the most suitable meanings. These methods are able to obtain wide coverage and good performance using structured knowledge, rivaling supervised methods (Patwardhan and Pedersen, 2006; Mohammad and Hirst, 2006; Agirre et al., 2010; Guo and Diab, 2010; Ponzetto and Navigli, 2010; Miller et al., 2012; Agirre et al., 2014; Moro et al., 2014; Chen et al., 2014; Camacho-Collados et al., 2015a).

3.5 Standard WSD features

As was analyzed by Lee and Ng (2002), conventional WSD systems usually make use of a fixed set of features to model the context of a word. The first feature is based on the words in the surroundings of the target word. The feature usually represents the local context as a binary array, where each position represents the occurrence of a particular word. Part-of-speech (POS) tags of the neighboring words have also been used extensively as a WSD feature. Local collocations represent another standard feature that captures the ordered sequences of words which tend to appear around the target word (Firth, 1957). Though not very popular, syntactic relations have also been studied as a possible feature (Stetina et al., 1998) in WSD.

More sophisticated features have also been studied. Examples are distributional semantic models, such as Latent Semantic Analysis (Van de Cruys and Apidianaki, 2011) and Latent Dirichlet Allocation (Cai et al., 2007). Inasmuch as they are the dominant distributional semantic model, word embeddings have also been applied as features to WSD systems. In this paper we study different methods through which word embeddings can be used as WSD features.

3.6 Word Embeddings as WSD features

Word embeddings have become a prominent technique in distributional semantics. These methods leverage neural networks in order to model the contexts in which a word is expected to appear. Thanks to their ability in efficiently learning the semantics of words, word embeddings have been applied to a wide range of NLP applications. Several studies have also investigated their integration into the Word Sense Disambiguation setting. These include the works of Zhong and Ng (2010), Taghipour and Ng (2015), Rothe and Schütze (2015), and Chen et al. (2014), which leverage embeddings for supervised (the former three) and knowledge-based (the latter) WSD. However, to our knowledge, no previous work has investigated methods for integrating word embeddings in WSD and the role that different training parameters can play. In this paper, we put forward a framework for a comprehensive evaluation of different methods of leveraging word embeddings as WSD features in a supervised WSD system. We provide an analysis of the impact of different parameters in the training of embeddings on the WSD performance. We consider four different strategies for integrating a pre-trained word embedding in a supervised WSD system, discussed in what follows.

3.6.1 Concatenation

Concatenation is our first strategy, which is inspired by the model of Bengio et al. (2003). This method consists of concatenating the vectors of the words surrounding a target word into a larger vector that has a size equal to the aggregated dimensions of all the individual embeddings. Let w_{ij} be the weight associated with the i^{th} dimension of the vector of the j^{th} word in the sentence, let D be the dimensionality of this vector, and W be the window size which is defined as the number of words on a single side. We are interested in representing the context of the I^{th} word in the sentence. The i^{th} dimension of the concatenation feature vector, which has a size of $2WD$, is computed as follows:

$$e_i = \begin{cases} w_{i \bmod D,\, I-W+\lfloor \frac{i}{D} \rfloor} & \text{if } \lfloor \frac{i}{D} \rfloor < W \\ w_{i \bmod D,\, I-W+1+\lfloor \frac{i}{D} \rfloor} & \text{otherwise} \end{cases}$$

where *mod* is the modulo operation, i.e., the remainder after division.

3.6.2 Average

As its name indicates, the average strategy computes the centroid of the embeddings of all the surrounding words. The formula divides each dimension by $2W$ since the number of context words is twice the window size:

$$e_i = \sum_{\substack{j=I-W \\ j\neq I}}^{I+W} \frac{w_{ij}}{2W}$$

3.6.3 Fractional decay

Our third strategy for constructing a feature vector on the basis of the context word embeddings is inspired by the way Word2vec combines the words in the context. Here, the importance of a word for our representation is assumed to be inversely proportional to its distance from the target word. Hence, surrounding words are weighted based on their distance from the target word:

$$e_i = \sum_{\substack{j=I-W \\ j \neq I}}^{I+W} w_{ij} \frac{W - |I - j|}{W}$$

3.6.4 Exponential decay

Exponential decay functions similarly to the fractional decay, which gives more importance to the close context, but in this case the weighting in the former is performed exponentially:

$$e_i = \sum_{\substack{j=I-W \\ j \neq I}}^{I+W} w_{ij} (1 - \alpha)^{|I-j|-1}$$

where $\alpha = 1 - 0.1^{(W-1)^{-1}}$ is the decay parameter. We choose the parameter in such a way that the immediate surrounding words contribute 10 times more than the last words on both sides of the window.

4 Framework

Our goal was to experiment with a state-of-the-art conventional supervised WSD system and a varied set of word embedding techniques. In this section we discuss the WSD system as well as the word embeddings used in our experiments.

4.1 WSD System

We selected It Makes Sense (Zhong and Ng, 2010, IMS) as our underlying framework for supervised WSD. IMS provides an extensible and flexible platform for supervised WSD by allowing the verification of different WSD features and classification techniques. By default, IMS makes use of three sets of features: (1) POS tags of the surrounding words, with a window of three words on each side, restricted by the sentence boundary, (2) the set of words that appear in the context of the target word after stopword removal, and (3) local collocations which consist of 11 features around the target word. IMS uses a linear support vector machine (SVM) as its classifier.

4.2 Embedding Features

We take the real-valued word embeddings as new features of IMS and introduce them into the system without performing any further modifications. We carried out experiments with three different embeddings:

- **Word2vec** (Mikolov et al., 2013): We used the Word2vec toolkit[2] to learn 400 dimensional vectors on the September-2014 dump of the English Wikipedia which comprises around three billion tokens. We chose the Skip-gram architecture with the negative sampling set to 10. The sub-sampling of frequent words was set to 10^{-3} and the window size to 10 words.
- **C&W** (Collobert and Weston, 2008): These 50 dimensional embeddings were learnt using a neural network model, consisting of several layers for feature extraction. The vectors were trained on a subset of the English Wikipedia.[3]
- **Retrofitting**: Finally, we used the approach of Faruqui et al. (2015) to retrofit our Word2vec vectors. We used the Paraphrase Database (Ganitkevitch et al., 2013, PPDB) as external knowledge base for retrofitting and set the number of iterations to 10.

5 Experiments

We evaluated the performance of our embedding-based WSD system on two standard WSD tasks: lexical sample and all-words. In all the experiments in this section we used the exponential decay strategy (cf. Section 3.6) and a window size of ten words on each side of the target word.

5.1 Lexical Sample WSD Experiment

The lexical sample WSD tasks provide training datasets in which different occurrences of a small set of words are sense annotated. The goal is for a WSD system to analyze the contexts of the individual senses of these words and to capture clues that can be used for distinguishing different senses of a word from each other at the test phase.

Datasets. As our benchmark for the lexical sample WSD, we chose the Senseval-2 (Edmonds and Cotton, 2001), Senseval-3 (Mihalcea et al., 2004), and SemEval-2007 (Pradhan et al., 2007) English Lexical Sample WSD tasks. The former two cover nouns, verbs and adjectives in their datasets whereas the latter task focuses on nouns and verbs

[2] `code.google.com/archive/p/word2vec/`

[3] `http://ronan.collobert.com/senna/`

Task	Training			Test		
	noun	verb	adjective	noun	verb	adjective
Senseval-2 (SE2)	4851	3566	755	1740	1806	375
Senseval-3 (SE3)	3593	3953	314	1807	1978	159
SemEval-07 (SE7)	13287	8987	–	2559	2292	–

Table 1: The number of sentences per part of speech in the datasets of the English lexical sample tasks we considered for our experiments.

System	SE2	SE3	SE7
IMS (2010)	65.3	72.9	87.9
Taghipour and Ng (2015)	66.2	73.4	–
AutoExtend (2015)	66.5	73.6	–
IMS + C&W	64.3	70.1	88.0
IMS + Word2vec	**69.9**	**75.2**	**89.4**
IMS + Retrofitting	65.9	72.8	88.3
C&W feature only	55.0	61.6	83.4
Word2vec feature only	65.6	69.4	87.0
Retrofitting feature only	67.2	72.7	88.0

Table 2: F1 performance on the three English lexical sample datasets. IMS + X denotes the improved IMS system when the X set of word representations were used as additional features. We also show in the last three rows the results for the IMS system when word representations were used as the only features.

only. Table 1 shows the number of sentences per part of speech for the training and test datasets of each of these tasks.

Comparison systems. In addition to the vanilla IMS system in its default setting we compared our system against two recent approaches that also modify the IMS system so that it can benefit from the additional knowledge derived from word embeddings for improved WSD performance: (1) the system of Taghipour and Ng (2015), which combines word embeddings of Collobert and Weston (2008) using the concatenation strategy (cf. Section 3.6) and introduces the combined embeddings as a new feature in addition to the standard WSD features in IMS; and (2) AutoExtend (Rothe and Schütze, 2015), which constructs a whole new set of features based on vectors made from words, senses and synsets of WordNet and incorporates them in IMS.

5.1.1 Lexical sample WSD results

Table 2 shows the F1 performance of the different systems on the three lexical sample datasets. As can be seen, the IMS + Word2vec system improves over all comparison systems including those that combine standard WSD and embedding features (i.e., the system of Taghipour and Ng (2015) and AutoExtend) across all the datasets. This shows that our proposed strategy for introducing word embeddings into the IMS system on the basis of exponential decay was beneficial. In the last three rows of the table, we also report the performance of the WSD systems that leverage only word embeddings as their features and do not incorporate any standard WSD feature. It can be seen that word embeddings, in isolation, provide competitive performance, which proves their capability in obtaining the information captured by standard WSD features. Among different embeddings, the retrofitted vectors provide the best performance when used in isolation.

5.2 All-Words WSD Experiments

The goal in this task is to disambiguate all the content words in a given text. In order to learn models for disambiguating a large set of content words, a high-coverage sense-annotated corpus is required. Since all-words tasks do not usually provide any training data, the challenge here is not only to learn accurate disambiguation models from the training data, as is the case in the lexical sample task, but also to gather high-coverage training data and to learn disambiguation models for as many words as possible.

Training corpus. As our training corpus we opted for two available resources: SemCor and OMSTI. SemCor (Miller et al., 1994) is a manually sense-tagged corpus created by the WordNet project team at Princeton University. The dataset is a subset of the English Brown Corpus and comprises around 360,000 words, providing annotations for more than 200K content words.[4] OM-

[4]We used automatic mappings to WordNet 3.0 provided in `web.eecs.umich.edu/~mihalcea/downloads.html`.

STI[5] (One Million Sense-Tagged for Word Sense Disambiguation and Induction) was constructed based on the DSO corpus (Ng and Lee, 1996) and provides annotations for around 42K different nouns, verbs, adjectives, and adverbs.

Datasets. As benchmark for this experiment, we considered the Senseval-2 (Edmonds and Cotton, 2001), Senseval-3 (Snyder and Palmer, 2004), and SemEval-2007 (Pradhan et al., 2007) English all-words tasks. There are 2474, 2041, and 465 words for which at least one of the occurrences has been sense annotated in the Senseval-2, Senseval-3 and SemEval-2007 datasets, respectively.

Experimental setup. Similarly to the lexical sample experiment, in the all-words setting we used the exponential decay strategy (cf. Section. 4.2) in order to incorporate word embeddings as new features in IMS. For this experiment, we only report the results for the best-performing word embeddings in the lexical sample experiment, i.e., Word2vec (see Table 2).

Comparison systems. We benchmarked the performance of our system against five other systems. Similarly to our lexical sample experiment, we compared against the vanilla IMS system and the work of Taghipour and Ng (2015). In addition, we performed experiments on the nouns subsets of the datasets in order to be able to provide comparisons against two other WSD approaches: Babelfy (Moro et al., 2014) and Muffin (Camacho-Collados et al., 2015a). Babelfy is a multilingual knowledge-based WSD and Entity Linking algorithm based on the semantic network of BabelNet. Muffin is a multilingual sense representation technique that combines the structural knowledge derived from semantic networks with the distributional statistics obtained from text corpora. The system uses sense-based representations for performing WSD. Camacho-Collados et al. (2015a) also proposed a hybrid system that averages the disambiguation scores of IMS with theirs (shown as "Muffin + IMS" in our tables). We also report the results for UKB w2w (Agirre and Soroa, 2009), another knowledge-based WSD approach based on Personalized PageRank (Haveliwala, 2002). Finally, we also carried out experiments with the pre-trained models[6] that are provided with the IMS toolkit, as well as IMS trained on our two training corpora, i.e., SemCor and OMSTI.

[5]www.comp.nus.edu.sg/~nlp/corpora.html

[6]www.comp.nus.edu.sg/~nlp/sw/models.tar.gz

System	SE2	SE3	SE7
MFS baseline	60.1	62.3	51.4
IMS (Zhong and Ng, 2010)	68.2	67.6	58.3
Taghipour and Ng (2015)	–	**68.2**	–
IMS (pre-trained models)	67.7	67.5	58.0
IMS (SemCor)	62.5	65.0	56.5
IMS (OMSTI)	67.0	66.4	57.6
IMS + Word2vec (SemCor)	63.4	65.3	57.8
IMS + Word2vec (OMSTI)	**68.3**	**68.2**	**59.1**

Table 3: F1 performance on different English all-words WSD datasets.

System	SE2	SE3	SE7
MFS baseline	71.6	70.3	65.8
Babelfy	–	68.3	62.7
Muffin	–	–	66.0
Muffin + IMS	–	–	68.5
UBK w2w	–	65.3	56.0
IMS (pre-trained models)	77.5	74.0	66.5
IMS (SemCor)	73.0	70.8	64.2
IMS (OMSTI)	76.6	73.3	67.7
IMS + Word2vec (SemCor)	74.2	70.1	68.6
IMS + Word2vec (OMSTI)	**77.7**	**74.1**	**71.5**

Table 4: F1 performance in the nouns subsets of different all-words WSD datasets.

5.2.1 All-words WSD results

Tables 3 and 4 list the performance of different systems on, respectively, the whole and the noun-subset datasets of the three all-words WSD tasks. Similarly to our lexical sample experiment, the IMS + Word2vec system provided the best performance across datasets and benchmarks. The coupling of Word2vec embeddings to the IMS system proved to be consistently helpful. Among the two training corpora, as expected, OMSTI provided a better performance owing to its considerably larger size and higher coverage. Another point to be noted here is the difference between results of the IMS with the pre-trained models and those trained on the OMSTI corpus. Since we used the same system configuration across the two runs, we conclude that the OMSTI corpus is either substantially smaller or less representative than the corpus used by Zhong and Ng (2010) for building the pre-trained models of IMS. Despite this fact, the IMS + Word2vec system can consistently improve the performance of IMS (pre-trained models) across the three datasets. This shows that a proper introduction of word embeddings into a supervised WSD system can compensate the negative effect of using lower quality training data.

6 Analysis

We carried out a series of experiments in order to check the impact of different system parameters on the final WSD performance. We were particularly interested in observing the role that various training parameters of embeddings as well as WSD features have in the WSD performance. We used the Senseval-2 English Lexical Sample task as our benchmark for this analysis.

6.1 The effect of different parameters

Table 5 shows F1 performance of different configurations of our system on the task's dataset. We studied five different parameters: the type (i.e., w2v or Retrofitting) and dimensionality (200, 400, or 800) of the embeddings, combination strategy (concatenation, average, fractional or exponential decay), window size (5, 10, 20 and words), and WSD features (collocations, POS tags, surrounding words, all of these or none). All the embeddings in this experiment were trained on the same training data and, unless specified, with the same configuration as described in Section 4.2. As baseline we show in the table the performance of the vanilla WSD system, i.e., IMS. For better readability, we report the differences between the performances of our system and the baseline.

We observe that the addition of Word2vec word embeddings to IMS (+w2v in the table) was beneficial in all settings. Among combination strategies, concatenation and average produced the smallest gain and did not benefit from embeddings of higher dimensionality. However, the other two strategies, i.e., fractional and exponential decay, showed improved performance with the increase in the size of the employed embeddings, irrespective of the WSD features. The window size showed a peak of performance when 10 words were taken in the case of standard word embeddings. For retrofitting, a larger window seems to have been beneficial, except when no standard WSD features were taken. Another point to note here is that, among the three WSD features, POS proved to be the most effective one while due to the nature of the embeddings, the exclusion of the Surroundings features in addition to the inclusion of the embeddings was largely beneficial in all the configurations. Furthermore, we found that the best configurations for this task were the ones that excluded Surroundings, and included w2v embeddings with a window of 10 and 800 dimensions with exponential decay strategy (70.2% of F1 performance) as well as the configuration used in our experiments, with all the standard features, and w2v embeddings with 400 dimensions, a window of 10 and exponential decay strategy (69.9% of F1 performance).

The retrofitted embeddings provided lower performance improvement when added on top of standard WSD features. However, when they were used in isolation (shown in the right-most column), the retrofitted embeddings interestingly provided the best performance, improving the vanilla WSD system with standard features by 2.8 percentage points (window size 5, dimensionality 800). In fact, the standard features had a destructive role in this setting as the overall performance was reduced when they were combined with the retrofitted embeddings. Finally, we point out the missing values in the configuration with 800 dimensions and a window size of 20. Due to the nature of the concatenation strategy, this configuration greatly increased the number of features from embeddings only, reaching 32000 (800 x 2 x 20) features. Not only was the concatenation strategy unable to take advantage of the increased dimensionality, but also it was not able to scale.

These results show that a state-of-the-art supervised WSD system can be constructed without incorporating any of the conventional WSD features, which in turn demonstrates the potential of retrofitted word embeddings for WSD. This finding is interesting, because it provides the basis for further studies on how synonymy-based semantic knowledge introduced by retrofitting might play a role in effective WSD, and how retrofitting might be optimized for improved WSD. Indeed, such studies may provide the basis for re-designing the standard WSD features.

6.2 Comparison of embedding types

We were also interested in comparing different types of embeddings in our WSD framework. We tested for seven sets of embeddings with dif-

Collocations				✓						✓			✓				
POS							✓			✓			✓				
Surroundings				✓			✓						✓				
Dimensionality			200	400	800	200	400	800	200	400	800	200	400	800	200	400	800
System	Strategy	Window															
IMS				62.4			63.7			62.0			65.2			–	
+ w2v	Con	5	+0.1	+0.4	+0.1	-0.1	+0.3	+0.2	+0.1	+0.5	+0.1	-0.2	+0.1	+0.1	46.9	48.7	44.2
		10	-0.1	+0.5	+0.3	-0.1	+0.5	0.0	+0.6	+1.0	+0.5	-0.1	+0.1	-0.1	48.6	51.1	49.7
		20	-0.2	+0.4	—	-0.3	+0.3	—	+0.7	+1.5	—	-0.5	+0.4	—	52.5	54.1	—
+ w2v	Avg	5	+0.8	+1.0	+1.0	+1.3	+1.3	+1.4	+3.9	+4.2	+4.1	+1.7	+1.4	+1.6	58.3	59.9	61.3
		10	+0.8	+0.9	+0.9	+0.6	+0.7	+0.8	+3.6	+3.7	+3.9	+0.6	+0.6	+0.7	63.7	64.1	64.7
		20	+0.3	+0.3	+0.3	+0.5	+0.3	+0.4	+2.4	+2.3	+2.3	+0.2	+0.2	+0.2	62.7	63.1	63.5
+ w2v	Frac	5	+3.9	+4.9	+5.2	+4.2	+4.6	+5.3	+6.3	+6.6	+6.8	+3.0	+3.6	+3.8	61.2	63.1	64.8
		10	+4.9	+5.8	+5.7	+4.6	+5.2	+5.1	+5.9	+7.0	+7.4	+3.6	+4.3	+4.0	61.3	63.8	65.2
		20	+4.4	+4.5	+4.7	+3.7	+4.0	+4.3	+4.8	+6.1	+5.4	+3.2	+3.3	+3.4	61.2	63.4	63.9
+ w2v	Exp	5	+4.1	+5.0	+5.2	+4.1	+4.7	+5.0	+6.1	+6.1	+6.4	+2.9	+3.5	+3.7	62.3	64.7	64.9
		10	+5.4	**+6.6**	+6.4	+4.9	+5.8	**+6.0**	+7.2	+7.7	**+8.2**	+4.1	**+4.7**	+4.6	63.2	65.6	66.9
		20	+5.2	+5.6	+5.9	+4.4	+5.1	+4.9	+6.1	+7.0	+6.8	+3.9	+4.3	+4.2	61.9	64.4	65.2
+ Ret	Con	5	-0.1	-0.1	-0.1	-0.1	-0.1	0.0	+0.1	+0.1	-0.1	-0.1	+0.1	+0.1	50.7	53.5	50.9
		10	+0.1	0.0	0.0	-0.3	0.0	0.0	+0.1	+0.2	+0.1	0.0	0.0	0.0	52.1	54.2	53.4
		20	0.0	0.0	—	-0.2	0.0	—	+0.7	+0.3	—	0.0	-0.1	—	53.7	54.8	—
+ Ret	Avg	5	+0.1	0.0	-0.1	+0.1	0.0	-0.1	+0.8	+0.8	+0.7	+0.1	0.0	+0.1	60.7	60.3	60.5
		10	-0.2	-0.1	0.0	-0.2	-0.3	0.0	+0.7	+0.7	+0.5	0.0	+0.1	+0.1	58.9	58.4	58.2
		20	-0.1	+0.1	+0.1	-0.2	-0.2	-0.2	+0.5	+0.4	+0.4	0.0	0.0	0.0	56.5	56.0	55.5
+ Ret	Frac	5	+1.4	+1.3	+1.2	+1.2	+1.0	+0.9	+3.3	+3.1	+2.9	+0.5	+0.3	+0.3	66.5	67.3	67.7
		10	+1.7	+1.4	+1.2	+1.5	+1.4	+1.2	+5.2	+4.7	+4.5	+0.7	+0.8	+0.6	64.4	66.2	66.1
		20	+2.2	+2.2	+1.8	+2.2	+1.8	+2.0	+6.7	+6.4	+5.9	+1.3	+1.2	+1.0	64.0	64.2	64.7
+ Ret	Exp	5	+1.1	+1.1	+1.1	+0.8	+0.8	+0.7	+2.7	+2.6	+2.2	+0.3	+0.3	+0.3	66.8	67.7	**68.0**
		10	+1.5	+1.3	+1.0	+1.2	+1.1	+1.0	+4.4	+4.2	+3.8	+0.7	+0.7	+0.3	65.9	67.2	67.5
		20	+1.8	+1.7	+1.5	+1.7	+1.5	+1.5	+6.3	+5.9	+5.4	+1.1	+0.8	+0.7	65.1	65.8	66.5

Table 5: F1 performance of different models on the Senseval-2 English Lexical Sample task. We show results for varied dimensionality (200, 400, and 800), window size (5, 10 and 20 words) and combination strategy, i.e., Concatenation (Con), Averaging (Avg), Fractional decay (Frac), and Exponential decay (Exp). To make the table easier to read, we highlight each cell according to the relative performance gain in comparison to the IMS baseline (top row in the table).

ferent dimensionalities and learning techniques: Word2vec embeddings trained on Wikipedia, with the Skip-gram model for dimensionalities 50, 300 and 500 (for comparison reasons) and CBOW with 300 dimensions, Word2vec trained on the Google News corpus with 300 dimensions and the Skip-gram model, the 300 dimensional embeddings of GloVe, and the 50 dimensional C&W embeddings. Additionally we include experiments on a non-embedding model, a PMI-SVD vector space model trained by Baroni et al. (2014).

Table 6 lists the performance of our system with different word representations in vector space on the Senseval-2 English Lexical Sample task. The results corroborate the findings of Levy et al. (2015) that Skip-gram is more efficient in captur-

Word representations	Dim.	Combination strategy			
		Concatenation	Average	Fractional	Exponential
Skip-gram - GoogleNews	300	65.5	65.5	69.4	**69.6**
GloVe	300	61.7	66.3	66.7	**68.3**
CBOW - Wiki	300	65.1	65.4	**68.9**	68.8
Skip-gram - Wiki	300	65.2	65.6	68.9	**69.7**
PMI - SVD - Wiki	500	65.5	65.3	**67.3**	66.8
Skip-gram - Wiki	500	65.1	65.6	69.1	**69.9**
Collobert & Weston	50	58.6	**67.3**	62.9	64.3
Skip-gram - Wiki	50	65.0	65.7	68.3	**68.6**

Table 6: F1 percentage performance on the Senseval-2 English Lexical Sample dataset with different word representations models, vector dimensionalities (Dim.) and combination strategies.

ing the semantics than CBOW and GloVe. Additionally, the use of embeddings with decay fares well, independently of the type of embedding. The only exception is the C&W embeddings, for which the average strategy works best. We attribute this behavior to the nature of these embeddings, rather than to their dimensionality. This is shown in our comparison against the 50-dimensional Skip-gram embeddings trained on the Wikipedia corpus (bottom of Table 6), which performs well with both decay strategies, outperforming C&W embeddings.

7 Conclusions

In this paper we studied different ways of integrating the semantic knowledge of word embeddings in the framework of WSD. We carried out a deep analysis of different parameters and strategies across several WSD tasks. We draw three main findings. First, word embeddings can be used as new features to improve a state-of-the-art supervised WSD that only uses standard features. Second, integrating embeddings on the basis of an exponential decay strategy proves to be more consistent in producing high performance than the other conventional strategies, such as vector concatenation and centroid. Third, the retrofitted embeddings that take advantage of the knowledge derived from semi-structured resources, when used as the only feature for WSD can outperform state-of-the-art supervised models which use standard WSD features. However, the best performance is obtained when standard WSD features are augmented with the additional knowledge from Word2vec vectors on the basis of a decay function strategy. Our hope is that this work will serve as the first step for further studies on re-designing standard WSD features. We release at `https://github.com/iiacobac/ims_wsd_emb` all the codes and resources used in our experiments in order to provide a framework for research on the evaluation of new VSM models in the WSD framework. As future work, we plan to investigate the possibility of designing word representations that best suit the WSD framework.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Starting Grant MultiJEDI No. 259234.

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank For Word Sense Disambiguation. In *Proceedings of the 12th Conference of the EACL*, pages 33–41, Athens, Greece.

Eneko Agirre, David Martínez, Oier López de Lacalle, and Aitor Soroa. 2006. Two graph-based algorithms for state-of-the-art wsd. In *Proceedings of the 2006 EMNLP*, pages 585–593, Sydney, Australia.

Eneko Agirre, Aitor Soroa, and Mark Stevenson. 2010. Graph-based word sense disambiguation of biomedical documents. *Bioinformatics*, 26(22):2889–2896.

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. 2014. Random Walks for Knowledge-based Word Sense Disambiguation. *Comp. Ling.*, 40(1):57–84.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of the 36th ACL*, pages 86–90, Montreal, Quebec, Canada.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd ACL*, volume 1, pages 238–247, Baltimore, Maryland.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, 3:1137–1155.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the EACL*, pages 103–111, Athens, Greece.

Jun Fu Cai, Wee Sun Lee, and Yee Whye Teh. 2007. NUS-ML:Improving Word Sense Disambiguation Using Topic Features. In *Proceedings of the SemEval-2007*, pages 249–252, Prague, Czech Republic.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015a. A Unified Multilingual Semantic Representation of Concepts. In *Proceedings of the 53rd ACL*, volume 1, pages 741–751, Beijing, China.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015b. NASARI: a novel approach to a semantically-aware representation of items. In *Proceedings of the 2015 NAACL*, pages 567–577, Denver, Colorado.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 EMNLP*, pages 1025–1035, Doha, Qatar.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th ICML*, pages 160–167, Helsinki, Finland.

Antonio Di Marco and Roberto Navigli. 2013. Clustering and diversifying web search results with graph-based word sense induction. *Comp. Ling.*, 39(3):709–754.

Philip Edmonds and Scott Cotton. 2001. Senseval-2: Overview. In *The Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, Toulouse, France.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of the 2015 NAACL*, pages 1606–1615, Denver, Colorado.

J. R. Firth. 1957. A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis (special volume of the Philological Society)*, 1952-59:1–32.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings 2013 NAACL*, pages 758–764, Atlanta, Georgia.

Weiwei Guo and Mona Diab. 2010. Combining Orthogonal Monolingual and Multilingual Sources of Evidence for All Words WSD. In *Proceedings of the 48th ACL*, pages 1542–1551, Uppsala, Sweden.

Taher H. Haveliwala. 2002. Topic-sensitive PageRank. In *Proceedings of the 11th international conference on World Wide Web*, pages 517–526, Honolulu, Hawai.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving Word Representations Via Global Context And Multiple Word Prototypes. In *Proceedings of 50th ACL*, volume 1, pages 873–882, Jeju Island, South Korea.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: Learning Sense Embeddings for Word and Relational Similarity. In *Proceedings of the 53rd ACL*, volume 1, pages 95–105, Beijing, China.

Matt Insall, Todd Rowland, and Eric W. Weisstein. 2015. “Embedding”. From MathWorld–A Wolfram Web Resource (access Sep 11, 2015) `http://mathworld.wolfram.com/Embedding.html`.

Thomas K. Landauer and Susan T. Dutnais. 1997. A Solution to Platos Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104(2):211–240.

Yoong Keok Lee and Hwee Tou Ng. 2002. An Empirical Evaluation of Knowledge Sources and Learning Algorithms for Word Sense Disambiguation. In *Proceedings of the 2002 EMNLP*, volume 10, pages 41–48, Philadelphia, Pennsylvania.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *TACL*, 3:211–225.

Suresh Manandhar, Ioannis P. Klapaftis, Dmitriy Dligach, and Sameer S. Pradhan. 2010. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of SemEval-2010*, pages 63–68, Uppsala, Sweden.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of Recurrent-neural-network Architectures and Learning Methods for Spoken Language Understanding. In *INTERSPEECH*, pages 3771–3775, Lyon, France.

Rada Mihalcea and Ehsanul Faruque. 2004. Senselearner: Minimally Supervised Word Sense Disambiguation for All Words in Open Text. In *Proceedings of ACL/SIGLEX Senseval-3*, volume 3, pages 155–158, Barcelona, Spain.

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. 2004. The Senseval-3 English Lexical Sample Task. In *Proceedings of ACL/SIGLEX Senseval-3*, pages 25–28, Barcelona, Spain.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

George A. Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G Thomas. 1994. Using a Semantic Concordance for Sense Identification. In *Proceedings of the Workshop on HLT*, pages 240–243, Plainsboro, New Jersey.

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation. In *COLING*, pages 1781–1796, Mumbai, India.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Comm. ACM*, 38(11):39–41.

Saif Mohammad and Graeme Hirst. 2006. Determining Word Sense Dominance Using a Thesaurus. In *Proceedings of the 11th Conference of EACL*, pages 121–128, Trento, Italy.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *Transactions of the ACL*, 2:231–244.

Roberto Navigli. 2009. Word sense disambiguation: a survey. *ACM COMPUTING SURVEYS*, 41(2):1–69.

Hwee Tou Ng and Hian Beng Lee. 1996. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-based Approach. In *Proceedings of the 34th Meeting on ACL*, pages 40–47, Santa Cruz, California.

Siddharth Patwardhan and Ted Pedersen. 2006. Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts. In *Proceedings of the EACL 2006 Workshop Making Sense of Sense*, volume 1501, pages 1–8, Trento, Italy.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 EMNLP*, pages 1532–1543, Doha, Qatar.

Mohammad Taher Pilehvar and Roberto Navigli. 2014. A Large-scale Pseudoword-based Evaluation Framework for State-of-the-Art Word Sense Disambiguation. *Computational Linguistics*, 40(4):837–881.

Simone Paolo Ponzetto and Roberto Navigli. 2010. Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th ACL*, pages 1522–1531, Uppsala, Sweden.

Sameer S. Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 Task 17: English Lexical Sample, SRL and All Words. In *Proceedings of the SemEval-2007*, pages 87–92, Prague, Czech Republic.

Joseph Reisinger and Raymond J. Mooney. 2010. Multi-Prototype Vector-Space Models of Word Meaning. In *Proceedings of the 2010 Annual Conference of the NAACL*, pages 109–117, Los Angeles, California.

Sascha Rothe and Hinrich Schütze. 2015. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd ACL*, volume 1, pages 1793–1803, Beijing, China.

Hui Shen, Razvan Bunescu, and Rada Mihalcea. 2013. Coarse to Fine Grained Sense Disambiguation in Wikipedia. In **SEM 2013: The Secound Joint Conference on Lexical and Computational Semantics*, pages 22–31, Atlanta, Georgia.

Benjamin Snyder and Martha Palmer. 2004. The Senseval-3 English All-Words Task. In *Proceedings of ACL/SIGLEX Senseval-3*, pages 41–43, Barcelona, Spain.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 EMNLP*, pages 1631–1642, Seattle, USA.

Jiri Stetina, Sadao Kurohashi, and Makoto Nagao. 1998. General Word Sense Disambiguation Method Based on a Full Sentential Context. In *Usage of WordNet in Natural Language Processing, Proceedings of COLING-ACL Workshop*, Montreal, Quebec, Canada.

Kaveh Taghipour and Hwee Tou Ng. 2015. Semi-Supervised Word Sense Disambiguation Using Word Embeddings in General and Specific Domains. In *Proceedings of the 2015 Annual Conference of the NAACL*, pages 314–323, Denver, Colorado.

Tim Van de Cruys and Marianna Apidianaki. 2011. Latent Semantic Word Sense Induction and Disambiguation. In *Proceedings of the 49th ACL*, volume 1, pages 1476–1485, Portland, Oregon.

Mo Yu and Mark Dredze. 2014. Improving Lexical Embeddings with Semantic Knowledge. In *Proceedings of the 52nd ACL*, volume 2, pages 545–550, Baltimore, Maryland.

Zhi Zhong and Hwee Tou Ng. 2010. It Makes Sense: A Wide-coverage Word Sense Disambiguation System for Free Text. In *Proceedings of the 48th ACL*, pages 78–83, Uppsala, Sweden.