LSTMEmbed: Learning Word and Sense Representations from a Large Semantically Annotated Corpus with Long Short-Term Memories

Ignacio Iacobacci[1,2*] and **Roberto Navigli**[2]
[1]Huawei Noah's Ark Lab, London, United Kingdom
[2]Department of Computer Science, Sapienza University of Rome, Italy
ignacio.iacobacci@huawei.com
{iacobacci,navigli}@di.uniroma1.it

Abstract

While word embeddings are now a de facto standard representation of words in most NLP tasks, recently the attention has been shifting towards vector representations which capture the different meanings, i.e., senses, of words. In this paper we explore the capabilities of a bidirectional LSTM model to learn representations of word senses from semantically annotated corpora. We show that the utilization of an architecture that is aware of word order, like an LSTM, enables us to create better representations. We assess our proposed model on various standard benchmarks for evaluating semantic representations, reaching state-of-the-art performance on the SemEval-2014 word-to-sense similarity task. We release the code and the resulting word and sense embeddings at http://lcl.uniroma1.it/LSTMEmbed.

1 Introduction

Natural Language is inherently ambiguous, for reasons of communicative efficiency (Piantadosi et al., 2012). For us humans, ambiguity is not a problem, since we use common knowledge to fill in the gaps and understand each other. Therefore, a computational model suited to understanding natural language and working side by side with humans should be capable of dealing with ambiguity to a certain extent (Navigli, 2018). A necessary step towards creating such computer systems is to build formal representations of words and their meanings, either in the form of large repositories of knowledge, e.g., semantic networks, or as vectors in a geometric space (Navigli and Martelli, 2019).

In fact, Representation Learning (Bengio et al., 2013) has been a major research area in NLP over the years, and latent vector-based representations, called *embeddings*, seem to be a good candidate for coping with ambiguity. Embeddings encode lexical and semantic items in a low-dimensional continuous space. These vector representations capture useful syntactic and semantic information of words and senses, such as regularities in the natural language, and relationships between them, in the form of relation-specific vector offsets. Recent approaches, such as word2vec (Mikolov et al., 2013), and GloVe (Pennington et al., 2014), are capable of learning efficient word embeddings from large unannotated corpora. But while word embeddings have paved the way for improvements in numerous NLP tasks (Goldberg, 2017), they still conflate the various meanings of each word and let its predominant sense prevail over all others in the resulting representation. Instead, when these embedding learning approaches are applied to sense-annotated data, they are able to produce embeddings for word senses (Iacobacci et al., 2015).

A strand of work aimed at tackling the lexical polysemy issue has proposed the creation of sense embeddings, i.e. embeddings which separate the various senses of each word in the vocabulary (Huang et al., 2012; Chen et al., 2014; Iacobacci et al., 2015; Flekova and Gurevych, 2016; Pilehvar and Collier, 2016; Mancini et al., 2017, among others). One of the weaknesses of these approaches, however, is that they do not take word ordering into account during the learning process. On the other hand, word-based approaches based on RNNs that consider sequence information have been presented, but they are not competitive in terms of speed or quality of the embeddings (Mikolov et al., 2010; Mikolov and Zweig, 2012; Mesnil et al., 2013).

For example, in Figure 1 we show an excerpt of a t-SNE (Maaten and Hinton, 2008) projection of word and sense embeddings in the literature:

[*] Ignacio Iacobacci's work was mainly done at the Sapienza University of Rome.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1685–1695
Florence, Italy, July 28 - August 2, 2019.

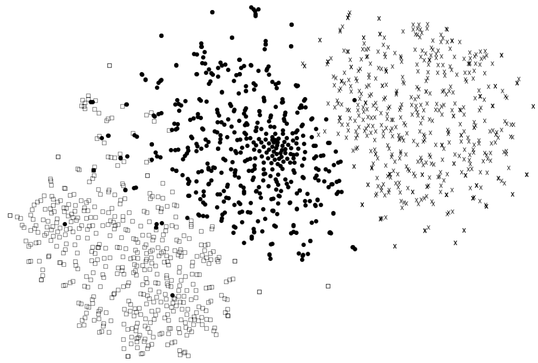

Figure 1: An example joint space where word vectors (squares) and sense vectors (dots and crosses) appear separated.

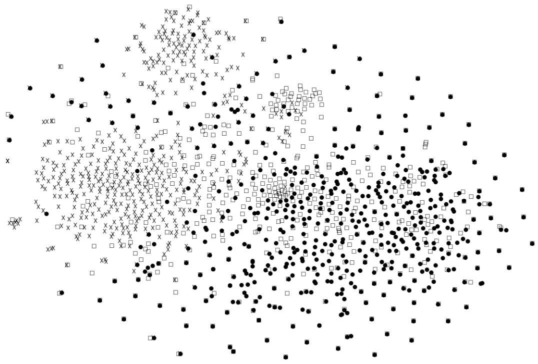

Figure 2: A shared space of words (squares) distributed across the space and two sense clusters (dots and crosses).

as can be seen, first, the ambiguous word *bank* is located close to words which co-occur with it (squares in the Figure), and, second, the closest senses of bank (dots for the financial institution meaning and crosses for its geographical meaning) appear clustered in two separated regions without a clear correlation with (potentially ambiguous) words which are relevant to them. A more accurate representation would be to have word vectors distributed across all the space with defined clusters for each set of vectors related to each sense of a target word (Figure 2).

Recently, the much celebrated Long-Short Term Memory (LSTM) neural network model has emerged as a successful model to learn representations of sequences, thus providing an ideal solution for many Natural Language Processing tasks whose input is sequence-based, e.g., sentences and phrases (Hill et al., 2016; Melamud et al., 2016; Peters et al., 2018). However, to date LSTMs have not been applied to the effective creation of sense embeddings linked to an explicit inventory.

In this paper, we explore the capabilities of the architecture of LSTMs using sense-labeled corpora for learning semantic representations of words and senses. We present four main contributions:

- We introduce LSTMEmbed, an RNN model based on a bidirectional LSTM for learning word and sense embeddings in the same semantic space, which – in contrast to the most popular approaches to the task – takes word ordering into account.

- We present an innovative idea for taking advantage of pretrained embeddings by using them as an objective during training.

- We show that LSTM-based models are suitable for learning not only contextual information, as is usually done, but also representations of individual words and senses.

- By linking our representations to a knowledge resource, we take advantage of the preexisting semantic information.

2 Embeddings for words and senses

Machine-interpretable representations of the meanings of words are key for a number of NLP tasks, and therefore obtaining good representations is an important research goal in the field, as shown by the surge of recent work on this topic.

2.1 Word Embeddings

In recent years, we have seen an exponential growth in the popularity of word embeddings. Models for learning embeddings, typically based on neural networks, represent individual words as low-dimensional vectors. Mikolov et al. (2013, word2vec) showed that word representations learned with a neural network trained on raw text geometrically encode highly latent relationships. The canonical example is the vector resulting from $king - man + woman$ found to be very close to the induced vector of $queen$. GloVe (Pennington et al., 2014), an alternative approach trained on aggregated global word-word co-occurrences, obtained similar results. While these embeddings are surprisingly good for monosemous words, they fail to represent the non-dominant senses of words properly. For instance, the representations of *bar* and pub should be similar, as well as those of bar and $stick$, but having similar representations for pub and $stick$ is undesirable. Several approaches were proposed to mitigate this issue: Yu and Dredze (2014) presented an alternative way to train word embeddings by using, in addition to common features, words having some relation in a semantic resource, like PPDB (Ganitkevitch et al., 2013) or WordNet (Miller, 1995). Faruqui et al. (2015) presented a technique applicable to pre-processed embeddings, in which vectors are updated ("retrofitted") in order to make them more similar to those which share a word type and less similar to those which do not. The word types were extracted from diverse semantic resources such as PPDB, WordNet and FrameNet (Baker et al., 1998). Melamud et al. (2016) introduced context2vec, a model based on a bidirectional LSTM for learning sentence and word embeddings. This model uses large raw text corpora to train a neural model that embeds entire sentential contexts and target words in the same low-dimensional space. Finally, Press and Wolf (2017) introduced a model, based on word2vec, where the embeddings are extracted from the output topmost weight matrix, instead of the input one, showing that those representations are also valid word embeddings.

2.2 Sense Embeddings

In contrast to the above approaches, each of which aims to learn representations of lexical items, sense embeddings represent individual word senses as separate vectors. One of the main approaches for learning sense embeddings is the so-called knowledge-based approach, which relies on a predefined sense inventory such as WordNet, BabelNet[1] (Navigli and Ponzetto, 2012) or Freebase[2]. SensEmbed[3] (Iacobacci et al., 2015) uses Babelfy[4], a state-of-the-art tool for Word Sense Disambiguation and Entity Linking, to build a sense-annotated corpus which, in turn, is used to train a vector space model for word senses with word2vec. SensEmbed exploits the structured knowledge of BabelNet's sense inventory along with the distributional information gathered from text corpora. Since this approach is based on word2vec, the model suffers from the lack of word ordering while learning embeddings. An alternative way of learning sense embeddings is to start from a set of pretrained word embeddings and split the vectors into their respective senses. This idea was implemented by Rothe and Schütze (2015) in AutoExtend, a system which learns embeddings for lexemes, senses and synsets from WordNet in a shared space. The synset/lexeme embeddings live in the same vector space as the word embeddings, given the constraint that words are sums of their lexemes and synsets are sums of their lexemes. AutoExtend is based on an auto-encoder, a neural network that mimics the input and output vectors. However, Mancini et al. (2017) pointed out that, by constraining the representations of senses, we cannot learn much about the relation between words and senses. They introduced SW2V, a model which extends word2vec to learn embeddings for both words and senses in the same vector space as an emerging feature, rather than via constraints on both representations. The model was built by exploiting large corpora and knowledge obtained from WordNet and BabelNet. Their basic idea was to extend the CBOW architecture of word2vec to represent both words and senses as different inputs and train the model in order to predict the word and its sense in the middle. Nevertheless, being based on word2vec, SW2V also lacks a notion of word ordering.

Other approaches in the literature avoid the use of a predefined sense inventory. The vectors learned by such approaches are identified as multi-prototype embeddings rather than senses, due to the fact that these vectors are only identified as different from one another, while there is no clear identification of their inherent sense. Several approaches have used this idea: Huang et al. (2012) introduced a model which learned multi vectors per word by clustering word context representations. Neelakantan et al. (2014) extended word2vec and included a module which induced new sense vectors if the context in which a word occurred was too different from the previously seen contexts for the same word. A similar approach was introduced by Li and Jurafsky (2015), which used a Chinese Restaurant Process as a way to induce new senses. Finally, Peters et al. (2018) presented ELMo, a word-in-context representation model based on a deep bidirectional language model. In contrast to the other related approaches, ELMo does not have a token dictionary, but rather

[1] https://babelnet.org

[2] http://developers.google.com/freebase

[3] http://lcl.uniroma1.it/sensembed/

[4] http://babelfy.org

each token is represented by three vectors, two of which are contextual. These models are, in general, difficult to evaluate, due to their lack of linkage to a lexical-semantic resource.

In marked contrast, LSTMEmbed, the neural architecture we present in this paper, aims to learn individual representations for word senses, linked to a multilingual lexical-semantic resource like BabelNet, while at the same time handling word ordering, and using pretrained embeddings as objective.

3 LSTMEmbed

Many approaches for learning embeddings are based on feed-forward neural networks (Section 2). However, recently LSTMs have gained popularity in the NLP community as a new de facto standard model to process natural language, by virtue of their context and word-order awareness. In this section we introduce LSTMEmbed, a novel method to learn word and sense embeddings jointly and which is based on the LSTM architecture.

3.1 Model Overview

At the core of LSTMEmbed is a bidirectional Long Short Term Memory (BiLSTM), a kind of recurrent neural network (RNN) which uses a set of gates especially designed for handling long-range dependencies. The bidirectional LSTM (BiLSTM) is a variant of the original LSTM (Hochreiter and Schmidhuber, 1997) that is particularly suited for temporal problems when access to the complete context is needed. In our case, we use an architecture similar to Kawakami and Dyer (2015), Kågebäck and Salomonsson (2016) and Melamud et al. (2016), where the state at each time step in the BiLSTM consists of the states of two LSTMs, centered in a particular timestep, accepting the input from previous timesteps in one LSTM, and the future timesteps in another LSTM. This is particularly suitable when the output corresponds to the analyzed timestep and not to the whole context.

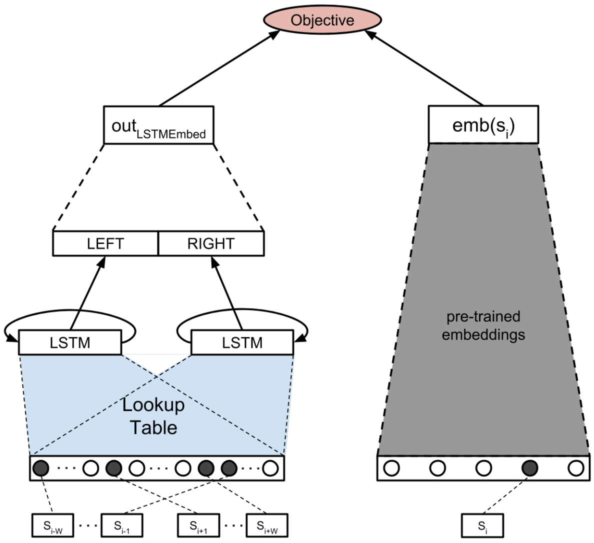

Figure 3: The LSTMEmbed architecture.

Figure 3 illustrates our model architecture. In marked contrast to the other LSTM-based approaches in the literature, we use sense-tagged text to provide input contexts of the kind $s_{i-W}, \ldots, s_{i-1}$ (the preceding context) and $s_{i+1}, \ldots, s_{i+W}$ (the posterior context), where s_j ($j \in [i-W, \ldots, i+W]$) is either a word or a sense tag from an existing inventory (see Section 4.1 for details). Each token is represented by its corresponding embedding vector $\mathbf{v}(s_j) \in \mathbb{R}^n$, given by a shared look-up table, which enables representations to be learned taking into account the contextual information on both sides. Next, the BiLSTM reads both sequences, i.e., the preceding context, from left to right, and the posterior context, from right to left:

$$\begin{aligned} o_l &= lstm_l(\mathbf{v}(s_{i-W}), ..., \mathbf{v}(s_{i-1})) \\ o_r &= lstm_r(\mathbf{v}(s_{i+1}), ..., \mathbf{v}(s_{i+W})) \end{aligned} \quad (1)$$

The model has one extra layer. The concatenation of the output of both LSTMs is projected linearly via a dense layer:

$$out_{LSTMEmbed} = \mathbf{W}^o(o_l \oplus o_r) \quad (2)$$

where $\mathbf{W}^o \in \mathbb{R}^{2m \times m}$ is the weights matrix of the dense layer with m being the dimension of the LSTM.

Then, the model compares $out_{LSTMEmbed}$ with $\mathbf{emb}(s_i)$, where $\mathbf{emb}(s_i)$ is a pretrained embedding vector of the target token (see Section 4.1 for an illustration of the pretrained embeddings that we use in our experiments), and, depending on the annotation and the pretrained set of embeddings used, this could be either a word, or a sense. At training time, the weights of the network are modified in order to maximize the similarity between $out_{LSTMEmbed}$ and $\mathbf{emb}(s_i)$. The loss function is calculated in terms of cosine similarity:

$$loss = 1 - \mathcal{S}(\vec{v_1}, \vec{v_2}) = 1 - \frac{\vec{v_1} \cdot \vec{v_2}}{\|\vec{v_1}\|\|\vec{v_2}\|} \quad (3)$$

Once the training is over, we obtain latent semantic representations of words and senses jointly in the same vector space from the look-up table, i.e., the embedding matrix between the input and the LSTM, with the embedding vector of an item s given by $\mathbf{v}(s)$.

In comparison to a standard BiLSTM, the novelties of LSTMEmbed can be summarized as follows:

- Using a sense-annotated corpus which includes both words and senses for learning the embeddings.
- Learning representations of both words and senses, extracted from a single look-up table, shared between both left and right LSTMs.
- A new learning method, which uses a set of pretrained embeddings as the objective, which enables us to learn embeddings for a large vocabulary.

4 Evaluation

We now present an experimental evaluation of the representations learned with LSTMEmbed. We first provide implementation details (Section 4.1), and then, to show the effectiveness of our model on a broad range of tasks, report on two sets of experiments: those involving sense-level tasks (Section 4.2) and those concerned with the word level (Section 4.3).

4.1 Implementation Details

Training data. We chose BabelNet (Navigli and Ponzetto, 2012) as our sense inventory.[5] BabelNet is a large multilingual encyclopedic dictionary and semantic network, comprising approximately 16 million entries for concepts and named entities linked by semantic relations. As training corpus we used the English portion of BabelWiki,[6] a multilingual corpus comprising the English Wikipedia (Scozzafava et al., 2015). The corpus was automatically annotated with named entities and concepts using Babelfy (Moro et al., 2014), a state-of-the-art disambiguation and entity linking system, based on the BabelNet semantic network. The English section of BabelWiki contains 3 billion tokens and around 3 million unique tokens.

Learning embeddings. LSTMEmbed was built with the Keras[7] library using Theano[8] as backend. We trained our models with an Nvidia Titan X Pascal GPU. We set the dimensionality of the look-up table to 200 due to memory constraints. We discarded the 1,000 most frequent tokens and set the batch size to 2048. The training was performed for one epoch. As optimizer function we used Adaptive Moment Estimation or Adam (Kingma and Ba, 2014).

As regards the objective embeddings $\mathbf{emb}(s_i)$ used for training, we chose 400-dimension sense embeddings trained using word2vec's SkipGram architecture with negative sampling on the BabelWiki corpus and recommended parameters for the SkipGram architecture: window size of 10, negative sampling set on 10, sub-sampling of frequent words set to 10^3.

4.2 Sense-based Evaluation

Our first set of experiments was aimed at showing the impact of our joint word and sense model in tasks where semantic, and not just lexical, relatedness is needed. We analyzed two tasks, namely Cross-Level Semantic Similarity and Most Frequent Sense Induction.

Comparison systems. We compared the performance of LSTMEmbed against alternative approaches to sense embeddings: SensEmbed (Iacobacci et al., 2015), which obtained semantic representations by applying word2vec to the English Wikipedia disambiguated with Babelfy; Nasari (Camacho-Collados et al., 2015), a technique for rich semantic representation of arbitrary concepts present in WordNet and Wikipedia pages; AutoExtend (Rothe and Schütze, 2015) which, starting from the word2vec word embeddings learned from GoogleNews[9], infers the representation of senses and synsets from WordNet; DeConf, an approach introduced by Pilehvar and Collier (2016) that decomposes a given word representation into its constituent sense representations by exploiting WordNet.

[5] We used version 4.0 as available from the website.

[6] http://lcl.uniroma1.it/babelfied-wikipedia/

[7] https://keras.io

[8] http://deeplearning.net/software/theano/index.html

[9] https://code.google.com/archive/p/word2vec/

Model	Pearson	Spearman
MeerkatMafia	**0.389***	0.380
SemantiKLU	0.314	0.327
SimCompass	0.356	0.344
AutoExtend	0.362	0.364
SensEmbed	0.316	0.333
SW2V	0.311	0.308
Nasari	0.244	0.220
DeConf	0.349	0.356
LSTMEmbed	0.380*	**0.400**

Table 1: Pearson and Spearman correlations on the CLSS word-to-sense similarity task. * Not statistically significant difference (χ^2, $p < 0.05$).

Experiment 1: Cross-Level Semantic Similarity. To best evaluate the ability of embeddings to discriminate between the various senses of a word, we opted for the SemEval-2014 task on Cross-Level Semantic Similarity (Jurgens et al., 2014, CLSS), which includes word-to-sense similarity as one of its sub-tasks. The CLSS word-to-sense similarity dataset comprises 500 instances of words, each paired with a short list of candidate senses from WordNet with human ratings for their word-sense relatedness. To compute the word-to-sense similarity we used our shared vector space of words and senses, and calculated the similarity using the cosine distance.

We included not only alternative sense-based representations but also the best performing approaches on this task: MeerkatMafia (Kashyap et al., 2014), which uses Latent Semantic Analysis (Deerwester et al., 1990) and WordNet glosses to get word-sense similarity measurements; SemantiKLU (Proisl et al., 2014), an approach based on a distributional semantic model trained on a large Web corpus from different sources; SimCompass (Banea et al., 2014), which combines word2vec with information from WordNet.

The results are given as Pearson and Spearman correlation scores in Table 1. LSTMEmbed achieves the state of the art by surpassing, in terms of Spearman correlation, alternative sense embedding approaches, as well as the best systems built specifically for the CLSS word-to-sense similarity task. In terms of Pearson, LSTMEmbed is on a par with the current state of the art, i.e., MeerkatMafia.

Model	P@1	P@3	P@5
AutoExtend	22.8	52.0	56.6
SensEmbed	38.4	56.1	63.0
SW2V	**39.7**	**60.3**	**67.5**
Nasari	27.4	40.2	44.6
DeConf	30.1	55.8	64.3
LSTMEmbed	39.0	59.2	66.0

Table 2: Precision on the MFS task (percentages).

Experiment 2: Most Frequent Sense Induction. In a second experiment, we employed our representations to induce the most frequent sense (MFS) of the input words, which is known to be a hard-to-beat baseline for Word Sense Disambiguation systems (Navigli, 2009). The MFS is typically computed by counting the word sense pairs in an annotated corpus such as SemCor (Miller et al., 1993).

To induce a MFS using sense embeddings, we identified – among all the sense embeddings of an ambiguous word – the sense which was closest to the word in terms of cosine similarity in the vector space. We evaluated all the sense embedding approaches on this task by comparing the induced most frequent senses against the MFS computed for all those words in SemCor which have a minimum number of 5 sense annotations (3731 words in total, that we release with the paper), so as to exclude words with insufficient gold-standard data for the estimates. We carried out our evaluation by calculating precision@K ($K \in \{1, 3, 5\}$). Table 2 shows that, across all the models, SW2V performs the best, leaving LSTMEmbed as the best runner-up approach.

4.3 Word-based Evaluation

While our primary goal was to show the effectiveness of LSTMEmbed on tasks in need of sense information, we also carried out a second set of experiments focused on word-based evaluations with the objective of demonstrating the ability of our joint word and sense embedding model to tackle tasks traditionally approached with word-based models.

Experiment 3: Synonym Recognition. We first experimented with synonym recognition: given a target word and a set of alternative words, the objective of this task was to select the member from

Model	Accuracy	
	TOEFL-80	ESL-50
word2vec	87.00	62.00
GloVe	88.75	60.00
Jauhar et al. (2015)	80.00	**73.33***
MSSG	78.26	57.14
Li and Jurafsky (2015)	82.61	50.00
MUSE	88.41	64.29
LSTMEmbed	**92.50**	72.00*

Table 3: Synonym Recognition: accuracy (percentages). * Not statistically significant difference (χ^2, $p < 0.05$).

the set which was most similar in meaning to the target word. The most likely synonym for a word w given the set of candidates $\mathcal{A}_w$ is calculated as:

$$Syn\left(w, \mathcal{A}_w\right) = \arg\max_{v \in \mathcal{A}_w} Sim\left(w, v\right) \qquad (4)$$

where Sim is the pairwise word similarity:

$$Sim\left(w_1, w_2\right) = \max_{\substack{s_1 \in \mathcal{S}_{w_1} \\ s_2 \in \mathcal{S}_{w_2}}} cosine\left(\vec{s_1}, \vec{s_2}\right) \qquad (5)$$

where $\mathcal{S}_{w_i}$ is the set of words and senses associated with the word w_i. We consider all the inflected forms of every word, with and without all its possible senses.

In order to evaluate the performance of LSTMEmbed on this task, we carried out experiments on two datasets. The first one, introduced by Landauer and Dumais (1997), is extracted directly from the synonym questions of the TOEFL (Test of English as a Foreign Language) questionnaire. The test comprises 80 multiple-choice synonym questions with four choices per question. The second one, introduced by Turney (2001), provides a set of questions extracted from the synonym questions of the ESL test (English as a Second Language). Similarly to TOEFL, it comprises 50 multiple-choice synonym questions with four choices per question.

Several related efforts used this kind of metric to evaluate their representations. We compare our approach with the following:

- Multi-Sense Skip-gram (Neelakantan et al., 2014, MGGS), an extension of the Skip-gram model of word2vec capable of learning multiple embeddings for a single word. The model makes no assumption about the number of prototypes.

- Li and Jurafsky (2015), a multi-sense embeddings model based on the Chinese Restaurant Process.

- Jauhar et al. (2015), a multi-sense approach based on expectation-maximization style algorithms for inferring word sense choices in the training corpus and learning sense embeddings while incorporating ontological sources of information.

- Modularizing Unsupervised Sense Embeddings (Lee and Chen, 2017, MUSE), an unsupervised approach that introduces a modularized framework to create sense-level representation learned with linear-time sense selection.

In addition, we included in the comparison two off-the-shelf popular word embedding models: GoogleNews, a set of word embeddings trained with word2vec, from a corpus of newspaper articles, and Glove.6B[10], a set of word embeddings trained on a merge of 2014 English Wikipedia dump and the corpus from Gigaword 5, for a total of 6 billion tokens.

In Table 3 we report the performance of LSTMEmbed together with the alternative approaches (the latter obtained from the respective publications). We can see that, on the TOEFL task, LSTMEmbed outperforms all other approaches, including the word-based models. On the ESL task, LSTMEmbed is the runner-up approach across systems and only by a small margin. The performance of the remaining models is considerably below ours.

Experiment 4: Outlier detection. Our second word-based evaluation was focused on outlier detection, a task intended to test the capability of the learned embeddings to create semantic clusters, that is, to test the assumption that the representation of related words should be closer than the representations of unrelated ones. We tested our model on the 8-8-8 dataset introduced by Camacho-Collados and Navigli (2016), containing eight clusters, each with eight words and eight possible outliers. In our case, we extended the

[10] https://nlp.stanford.edu/projects/glove/

Model	Corpus	Sense	8-8-8	
			OPP	Acc.
word2vec*	UMBC	-	92.6	73.4
	Wikipedia	-	93.8	70.3
	GoogleNews	-	94.7	70.3
GloVe*	UMBC	-	81.6	40.6
	Wikipedia	-	91.8	56.3
AutoExtend	GoogleNews	✓	82.8	37.5
SensEmbed	Wikipedia	✓	**98.0**	**95.3**
SW2V	Wikipedia	✓	48.4	37.5
Nasari	Wikipedia	✓	94.0	76.3
DeConf	GoogleNews	✓	93.8	62.5
LSTMEmbed	Wikipedia	✓	96.1	78.1

Table 4: Outlier detection task (* reported in Camacho-Collados and Navigli (2016)).

similarity function used in the evaluation to consider both the words in the dataset and their senses, similarly to what we had done in the synonym recognition task (cf. Equation 5). We can see from Table 4 that LSTMEmbed ranks second below SensEmbed in terms of both measures defined in the task (accuracy, and outlier position percentage, which considers the position of the outlier according to the proximity of the semantic cluster), with both approaches outperforming all other word-based and sense-based approaches.

5 Analysis

The objective embedding $\mathbf{emb}$ we used in our work uses pretrained sense embeddings obtained from word2vec trained on BabelWiki, as explained in Section 4.1. Our assumption was that training with richer and meaningful objective embeddings would enhance the representation delivered by our model in comparison to using word-based models. We put this hypothesis to the test by comparing the performance of LSTMEmbed equipped with five sets of pretrained embeddings on a word similarity task. We used the WordSim-353 (Finkelstein et al., 2002) dataset, which comprises 353 word pairs annotated by human subjects with a pairwise relatedness score. We computed the performance of LSTMEmbed with the different pretrained embeddings in terms of Spearman correlation between the cosine similarities of the LSTMEmbed word vectors and the WordSim-353 scores.

Model	Objective	Dim.	WS353
word2vec	-	-	0.488
GloVe	-	-	0.557
LSTMEmbed	random (baseline)	50	0.161
	word2vec	50	0.573
	word2vec + retro	50	0.569
	GoogleNews	300	0.574
	GloVe.6B	300	0.577
	SensEmbed	400	**0.612**

Table 5: Spearman correlation on the Word Similarity Task.

The first set of pretrained embeddings is a 50-dimension word space model, trained with word2vec Skip-gram with the default configuration. The second set consists of the same vectors, retrofitted with PPDB using the default configuration. The third is the GoogleNews set of pretrained embeddings. The fourth is the GloVe.6B word space model. Finally, we tested our model with the pretrained embeddings of SensEmbed. As a baseline we included a set of normalized random vectors. As is shown in Table 5, using richer pretrained embeddings improves the resulting representations given by our model. All the representations obtain better results compared to word2vec and GloVe trained on the same corpus, with the sense embeddings from SensEmbed, a priori the richest set of pretrained embeddings, attaining the best performance.

6 Conclusions

We presented LSTMEmbed, a new model based on a bidirectional LSTM for learning embeddings of words and senses jointly, and which is able to learn semantic representations on a par with, or better than, state-of-the-art approaches. We draw three main findings. Firstly, we have shown that our semantic representations are capable to properly reflect the similarity between word and sense representations, showing state-of-the-art performance in the sense-aware tasks of word-to-sense similarity and most frequent sense induction. Secondly, our approach is also able to attain high performance in standard word-based semantic evaluations, namely, synonym recognition and outlier detection. Finally, the introduction of an output layer which predicts pretrained embeddings enables us to use larger vocabularies instead of using the slower softmax. We release the word and sense embeddings at the following URL: http://lcl.uniroma1.it/LSTMEmbed.

Our model shows potential for further applications. We did, in fact, explore alternative configurations, for instance, using several layers or replacing the LSTMs with Gated Recurrent Units (Cho et al., 2014) or the Transformer architecture (Vaswani et al., 2017). Trying more complex networks is also within our scope and is left as future work.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487 under the European Union's Horizon 2020 research and innovation programme.

The authors gratefully acknowledge the support of NVIDIA Corporation Hardware Grant.

References

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90.

Carmen Banea, Di Chen, Rada Mihalcea, Claire Cardie, and Janyce Wiebe. 2014. SimCompass: Using Deep Learning Word Embeddings to Assess Cross-level Similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 560–565, Dublin, Ireland.

Y. Bengio, A. Courville, and P. Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

José Camacho-Collados and Roberto Navigli. 2016. Find the word that does not belong: A framework for an intrinsic evaluation of word vector representations. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 43–50, Berlin, Germany.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. NASARI: a novel approach to a semantically-aware representation of items. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 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 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035, Doha, Qatar.

Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, Colorado.

Lev Finkelstein, Gabrilovich Evgeniy, Matias Yossi, Rivlin Ehud, Solan Zach, Wolfman Gadi, and Ruppin Eytan. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Lucie Flekova and Iryna Gurevych. 2016. Supersense embeddings: A unified model for supersense interpretation, prediction, and utilization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2029–2041.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 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, Atlanta, Georgia.

Yoav Goldberg. 2017. *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers.

Felix Hill, KyungHyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics, Volume 4*, pages 17–30.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

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 the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 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 Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 95–105, Beijing, China.

Sujay Kumar Jauhar, Chris Dyer, and Eduard Hovy. 2015. Ontologically Grounded Multi-sense Representation Learning for Semantic Vector Space Models. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 683–693, Denver, Colorado.

David Jurgens, Mohammad Taher Pilehvar, and Roberto Navigli. 2014. SemEval-2014 Task 3: Cross-Level Semantic Similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 17–26, Dublin, Ireland.

Mikael Kågebäck and Hans Salomonsson. 2016. Word Sense Disambiguation using a Bidirectional LSTM. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*, Osaka, Japan. The COLING 2016 Organizing Committee.

Abhay Kashyap, Lushan Han, Roberto Yus, Jennifer Sleeman, Taneeya Satyapanich, Sunil Gandhi, and Tim Finin. 2014. Meerkat Mafia: Multilingual and Cross-Level Semantic Textual Similarity Systems. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 416–423, Dublin, Ireland.

Kazuya Kawakami and Chris Dyer. 2015. Learning to represent words in context with multilingual supervision. *CoRR*, abs/1511.04623.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211.

Guang-He Lee and Yun-Nung Chen. 2017. MUSE: Modularizing Unsupervised Sense Embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 327–337, Copenhagen, Denmark.

Jiwei Li and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1722–1732, Lisbon, Portugal.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Massimiliano Mancini, Jose Camacho-Collados, Ignacio Iacobacci, and Roberto Navigli. 2017. Embedding words and senses together via joint knowledge-enhanced training. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 100–111.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. *context2vec*: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of RNN architectures and learning methods for spoken language understanding. In *INTERSPEECH-2013*, pages 3771–3775, Lyon, France.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH-2010*, Makuhari, Japan.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239, Miami, Florida. IEEE.

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross Bunker. 1993. A semantic concordance. In *Proceedings of the Workshop on Human Language Technology*, pages 21–24, Plainsboro, New Jersey.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics, Volume 2*, pages 231–244.

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

Roberto Navigli. 2018. Natural language understanding: Instructions for (present and future) use. In *Proc. of IJCAI*, pages 5697–5702.

Roberto Navigli and Federico Martelli. 2019. An Overview of Word and Sense Similarity. *Natural Language Engineering*, 25(6).

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. *AI*, 193:217–250.

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

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 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.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana.

Steven T. Piantadosi, Harry Tily, and Edward Gibson. 2012. The communicative function of ambiguity in language. *Cognition*, 122(3):280 – 291.

Mohammad Taher Pilehvar and Nigel Collier. 2016. De-conflated semantic representations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1680–1690, Austin, Texas.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain.

Thomas Proisl, Stefan Evert, Paul Greiner, and Besim Kabashi. 2014. SemantiKLUE: Robust Semantic Similarity at Multiple Levels Using Maximum Weight Matching. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 532–540, Dublin, Ireland.

Sascha Rothe and Hinrich Schütze. 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.

Federico Scozzafava, Alessandro Raganato, Andrea Moro, and Roberto Navigli. 2015. Automatic identification and disambiguation of concepts and named entities in the multilingual wikipedia. In *AIxIA*, pages 357–366.

Peter D. Turney. 2001. Mining the Web for Synonyms: PMI-IR Versus LSA on TOEFL. In *Proceedings of the Twelth European Conference on Machine Learning (ECML)*, pages 491–502, Freiburg, Germany. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008. Curran Associates, Inc.

Mo Yu and Mark Dredze. 2014. Improving Lexical Embeddings with Semantic Knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 545–550, Baltimore, Maryland.