Learning Word Representations with Regularization from Prior Knowledge

Yan Song
Tencent AI Lab
clksong@tencent.com

Chia-Jung Lee
Microsoft
cjlee@microsoft.com

Fei Xia
University of Washington
fxia@uw.edu

Abstract

Conventional word embeddings are trained with specific criteria (e.g., based on language modeling or co-occurrence) inside a single information source, disregarding the opportunity for further calibration using external knowledge. This paper presents a unified framework that leverages pre-learned or external priors, in the form of a regularizer, for enhancing conventional language model-based embedding learning. We consider two types of regularizers. The first type is derived from topic distribution by running latent Dirichlet allocation on unlabeled data. The second type is based on dictionaries that are created with human annotation efforts. To effectively learn with the regularizers, we propose a novel data structure, trajectory softmax, in this paper. The resulting embeddings are evaluated by word similarity and sentiment classification. Experimental results show that our learning framework with regularization from prior knowledge improves embedding quality across multiple datasets, compared to a diverse collection of baseline methods.

1 Introduction

Distributed representation of words (or word embedding) has been demonstrated to be effective in many natural language processing (NLP) tasks (Bengio et al., 2003; Collobert and Weston, 2008; Turney and Pantel, 2010; Collobert et al., 2011; Mikolov et al., 2013b,d; Weston et al., 2015). Conventional word embeddings are trained with a single objective function (e.g., language modeling (Mikolov et al., 2013c) or word co-occurrence factorization (Pennington et al., 2014)), which restricts the capability of the learned embeddings from integrating other types of knowledge. Prior work has leveraged relevant sources to obtain embeddings that are best suited for the target tasks, such as Maas et al. (2011) using a sentiment lexicon to enhance embeddings for sentiment classification. However, learning word embeddings with a particular target makes the approach less generic, also implying that customized adaptation has to be made whenever a new knowledge source is considered.

Along the lines of improving embedding quality, semantic resources have been incorporated as guiding knowledge to refine objective functions in a joint learning framework (Bian et al., 2014; Xu et al., 2014; Yu and Dredze, 2014; Nguyen et al., 2016), or used for retrofitting based on word relations defined in the semantic lexicons (Faruqui et al., 2015; Kiela et al., 2015). These approaches, nonetheless, require explicit word relations defined in semantic resources, which is a difficult prerequisite for knowledge preparation.

Given the above challenges, we propose a novel framework that extends typical context learning by integrating external knowledge sources for enhancing embedding learning. Compared to a well known work by Faruqui et al. (2015) that focused on tackling the task using a retrofitting[1] framework on semantic lexicons, our method has an emphasis on joint learning where two objectives are considered for optimization simultaneously. In the meantime, we design a general-purpose infrastructure which can incorporate arbitrary external sources into learning as long as the sources can be encoded into vectors of numerical values (e.g. multi-hot vector according to the topic distributions from a topic model). In prior work by Yu and Dredze (2014) and Kiela et al. (2015), the ex-

[1]In their study, joint learning was reported to be less effective than retrofitting.

ternal knowledge has to be clustered beforehand according to their semantic relatedness (e.g., *cold*, *icy*, *winter*, *frozen*), and words of similar meanings are added as part of context for learning. This may set a high bar for preparing external knowledge since finding the precise word-word relations is required. Our infrastructure, on the other hand, is more flexible as knowledge that is learned elsewhere, such as from topic modeling or even a sentiment lexicon, can be easily encoded and incorporated into the framework to enrich embeddings.

The way we integrate external knowledge is performed by the notion of a regularizer, which is an independent component that can be connected to the two typical architectures, namely, continuous bag-of-words (CBOW) and skip-gram (SG), or used independently as a retrofitter. We construct the regularizers based on the knowledge learned from both unlabeled data and manually crafted information sources. As an example of the former, a topic model from latent Dirichlet allocation (LDA) (Blei et al., 2003) is first generated from a given corpus, based on which per-word topical distributions are then added as extra signals to aid embedding learning. As an example of the latter, one can encode a dictionary into the regularizer and thus adapt the learning process with the encoded knowledge.

Another contribution of this paper is that we propose a novel data structure, trajectory softmax, to effectively learn prior knowledge in the regularizer. Compared to conventional tree based hierarchical softmax, trajectory softmax can greatly reduce the space complexity when learning over a high-dimension vector. Our experimental results on several different tasks have demonstrated the effectiveness of our approach compared to up-to-date studies.

The rest of the paper is organized as follows. In section 2, we describe in detail our framework and show how we learn the regularizer in section 3. Section 4 presents and analyzes our experimental results and section 5 surveys related work. Finally, conclusions and directions of future work are discussed in section 6.

2 Approach

Conventionally word embeddings are learned from word contexts. In this section, we describe our method of extending embedding learning to incorporate other types of information sources. Previous work has shown that many different sources can help learn better embeddings, such as semantic lexicons (Yu and Dredze, 2014; Faruqui et al., 2015; Kiela et al., 2015) or topic distributions (Maas et al., 2011; Liu et al., 2015b). To provide a more generic solution, we propose a unified framework that learns word embeddings from context (e.g., CBOW or SG) together with the flexibility of incorporating arbitrary external knowledge using the notion of a regularizer. Details are unfolded in following subsections.

2.1 The Proposed Learning Framework

Preliminaries: The fundamental principle for learning word embeddings is to leverage word context, with a general goal of maximizing the likelihood that a word is predicted by its context. For example, the CBOW model can be formulated as maximizing

$$\mathcal{L} = \frac{1}{|V|}\sum_{i=1}^{|V|} \log p(w_i \mid \sum_{0<|j|\le c} \upsilon_{i+j}), \ \forall\, w_i \in V \tag{1}$$

where υ_{i+j} refers to the embedding of a word in w_{i-c}^{i+c}, and c defines the window size of words adjacent to the word w_i. The optimization for $\mathcal{L}$ over the entire corpus is straightforward.

The left part of Figure 1 illustrates the concept of such context learning. It is a typical objective function for language modeling, where w_i is learned by the association with its neighboring words. Since context greatly affects the choice of the current word, this modeling strategy can help finding reasonable semantic relationships among words.

Regularizer: To incorporate additional sources for embedding learning, we introduce the notion of a regularizer, which is designed to encode information from arbitrary knowledge corpora.

Given a knowledge resource Ψ, one can encode the knowledge carried by a word w with $\psi(w)$, where ψ can be any function that maps w to the knowledge it encapsulates. For example, a word has a topic vector $\psi(w) = \overrightarrow{e^{(w_i)}}\Phi_{[1:K,:]}$, resulting $\psi(w) = \overrightarrow{\Phi}_w = (\phi_{1,w}, \phi_{2,w}, ..., \phi_{K,w})$, where $\Phi_{[1:K,:]}$ is the topic distribution matrix for all words with K topics; $\overrightarrow{e^{(w_i)}}$ is the standard basis vector with 1 at the i-th position in the vocabulary V. Therefore, regularization for all w with given

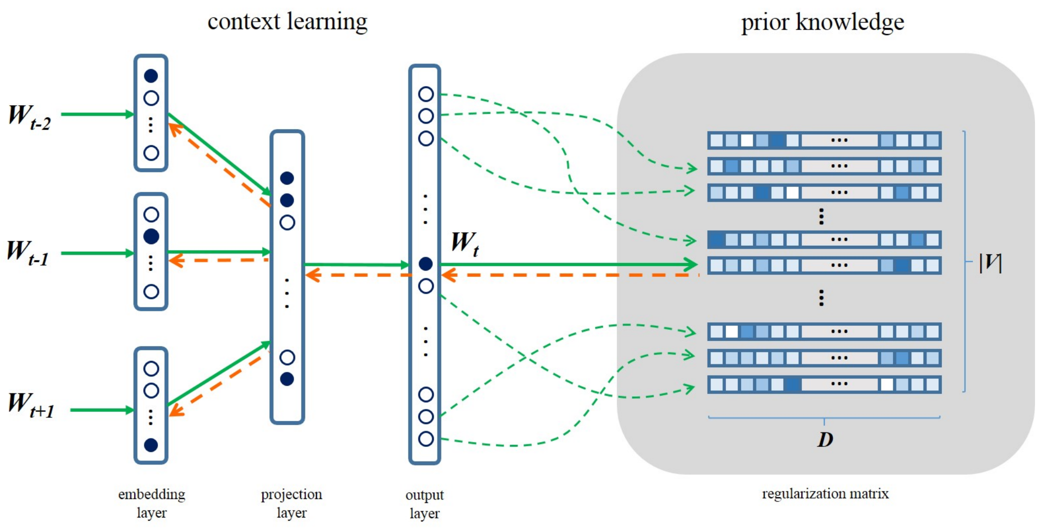

Figure 1: Illustration of joint learning word embeddings with context and regularization from prior knowledge. The green lines refer to the prediction and the red dotted lines refer to the updating process.

a knowledge source can be conceptually used to maximize $\sum_{w \in V} R(v)$, where R is the regularizer, defined as a function of the embedding v of a given word w and formulated as:

$$R(v) = \log p(\psi(w)|v), \ \forall\, w \in V, \Psi \quad (2)$$

The right part of Figure 1 shows an instantiation of a regularizer that encodes prior knowledge of vocabulary size $|V|$, each with D dimensions.

Joint Learning: To extend conventional embedding learning, we combine context learning from an original corpus with external knowledge encoded by a regularizer, where the shared vocabulary set forms a bridge connecting the two spaces. In particular, the objective function for CBOW with integrating the regularizer can be formulated as maximizing

$$\mathcal{L} = \frac{1}{|V|} \sum_{i=1}^{|V|} \log p(w_i, \psi(w_i) \mid \sum_{0<|j|\leq c} v_{i+j}) \quad (3)$$

where not only w_i, but also $R(w_i)$ is predicted by the context words w_{i+j} via their embeddings v_{i+j}.

Figure 1 as a whole illustrates this idea. Recall that each row of the matrix corresponds to a vector of a word in V, representing prior knowledge across D dimensions (e.g., semantic types, classes or topics). When learning/predicting a word within this framework, the model needs to predict not only the correct word as shown in the context learning part in the figure, but also the correct vector in the regularizer. In doing so, the prior knowledge will be carried to word embeddings from regularization to context learning by back-propagation through the gradients obtained from the learning process based on the regularization matrix.

Retrofitting: With joint learning as our goal, we should emphasize that the proposed framework supports simultaneous context learning and prior knowledge retrofitting with a unified objective function. This means that the retrofitters can be considered as a stand-alone component at disposal, where the external knowledge vectors are regarded as supervised-learning target and the embeddings are updated through the course of fitting to the target. In §4, we will evaluate the performance of both joint learner and retrofitter in detail.

2.2 Parameter Estimation

As shown in Equation 3, prior knowledge participates in the optimization process for predicting the current word and contributes to embedding updating during training a CBOW model. Using stochastic gradient descent (SGD), embeddings can be easily updated by both objective functions for language modeling and regularization through:

$$v^{\star}_{i+j} = v_{i+j} - \lambda \nabla_v [\log p(w_i| \sum_{0<|j|\leq c} v_{i+j}) + R(v^{\star}_i)] \quad (4)$$

where R is defined as in Eq.2 for $\psi(w_i)$. For SG model, prior knowledge is introduced in a similar way, with the difference being that context words are predicted instead of the current word.

Therefore, when learned from the context, em-

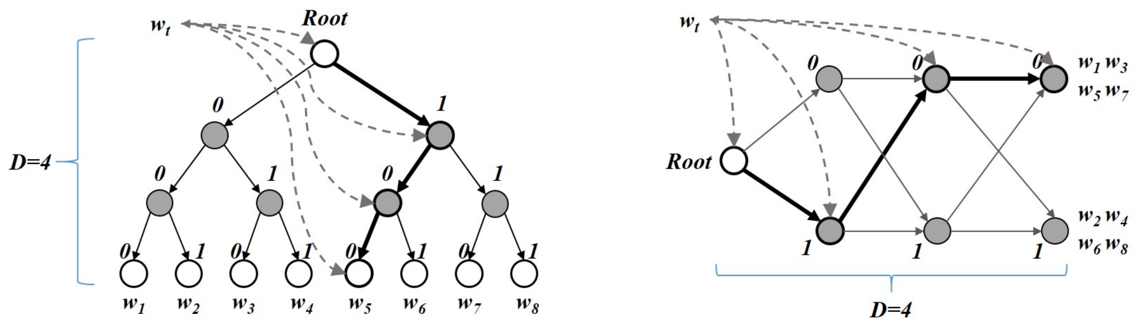

Figure 2: Comparison of hierarchical softmax (left) and trajectory softmax (right) based on an example of eight words in binary coding. The bold arrow lines refer to the path for encoding w_5 in both hierarchical and trajectory softmax.

beddings are updated in the same way as in normal CBOW and SG models. When learned from the regularizer, embeddings are updated via a supervised learning over Ψ, on the condition that Ψ is appropriately encoded by ψ. The details of how it is performed will be illustrated in the next subsection.

2.3 Trajectory Softmax

Hierarchical softmax is a good choice for reducing the computational complexity when training probabilistic neural network language models. Therefore, for context learning on the left part of Figure 1, we continue using hierarchical softmax based on Huffman coding tree (Mikolov et al., 2013a). Typically to encode the entire vocabulary, the depth of the tree falls in a manageable range around 15 to 18.

However, different from learning context words, to encode a regularizer as shown on the right part of Figure 1, using hierarchical softmax is intractable due to exponential space demand. Consider words expressed with D-dimensional vectors in a regularizer, a tree-based hierarchical softmax may require $2^D - 1$ nodes, as illustrated in the left hand side of Figure 2. Since each node contains a d-dimensional "node vector" that is to be updated through training, the total space required is $O(2^D \cdot d)$ for hierarchical softmax to encode the regularizer. When D is very large, such as $D = 50$ meaning that tree depth is 50, the space demand tends to be unrealistic as the number of nodes in the tree grows to 2^{50}.

To avoid the exponential requirement in space, in this work, we propose a trajectory softmax activation to effectively learn over the D-dimensional vectors. Our approach follows a grid hierarchical structure along a path when conducting learning in the regularizer. From the right hand side of Figure 2, we see that the same regularizer entry is encoded with a path of D nodes, using a grid structure instead of a tree one. Consequently the total space required will be reduced to $O(2 \cdot D \cdot d)$.

As a running example, Figure 2 shows that when $D = 4$, the conventional hierarchical softmax needs at least 15 nodes to perform softmax over the path, while trajectory softmax greatly reduces space to only 7 nodes. Compared to tree-based hierarchical softmax, the paths in trajectory softmax are not branches of a tree, but a fully connected grid of nodes with space complexity of $D \times |C|$ in general. Here $|C|$ refers to the number of choices on the paths for a node to the next node, and thus $|C| = 2$ is the binary case. In Figure 2, we see an activation trajectory for a sequence of "*Root→100*" for encoding word w_5. w_t is then learned and updated through the nodes on the trajectory when w_5 is predicted by w_t. The learning and updating are referred by the dashed arrow lines. Overall, trajectory softmax greatly reduces the space complexity than hierarchical softmax, especially when words sharing similar information, in which case the paths of these words will be greatly overlapped.

More formally, learning with trajectory softmax in the binary case is similar to hierarchical softmax, which is to maximize p over the path for a vector encoded in $\psi(w)$, where p is defined below with an input vector υ:

$$p(\psi(w)|\upsilon) = \prod_{i=1}^{D-1} \sigma([\![n(i+1)]\!] \cdot \upsilon_i^\top \upsilon) \qquad (5)$$

where υ_i is the inner vector in i-th node on the trajectory. $[\![n(i+1)]\!] = 1$ or -1 when $(i+1)$-th node is encoded with 0 or 1, respectively. The final update to word embedding v with the regularizer is conducted by:

$$v^* = v - \gamma(\sigma(v_i^\top v) - t_i) \cdot v_i \tag{6}$$

which is applied to $i = 1, 2, ..., D - 1$, where $\sigma(x) = \exp(x)/(1 + \exp(x))$; $t_i = [\![n(i+1)]\!]$; γ is a discount learning rate.

Since the design of trajectory softmax is compatible with the conventional hierarchical softmax, one can easily implement the joint learning by concatenating its *Root* with the terminal node in the hierachical tree. The learning process is thus to traverse all the nodes from the hierarchical tree and the trajectory path.

3 Constructing Regularizers

We consider two categories of information sources for constructing regularizers. The first type of regularizer is built based on resources without annotation. On the contrary, the second type uses text collections with annotation. For brevity, throughout the paper we refer to the former as unannotated regularizer whereas the latter is recognized as annotated regularizer.

3.1 Unannotated Regularizer

The unannotated regularizer constructs its regularization matrix based on an LDA learned topic distribution, which reflects topical salience information of a given word from prior knowledge. Using LDA not only serves our purpose of learning according to word semantics reflected by cooccurrences but can also bring in knowledge inexpensively (i.e., no annotations needed).

To start, a classic LDA is first performed on an arbitrary base corpus for retrieving word topical distribution, resulting in a topic model with K topics. All the units in the corpus are then assigned with a word-topic probability ϕ_i corresponding to topic k, based on which a matrix is formed with all $\overrightarrow{\Phi}_w$, as described in §2.1. Next we convert each $\overrightarrow{\Phi}$ into a 0-1 vector based on the maximum values in $\overrightarrow{\Phi}$. In particular, positions with maximum values are set to 1 and the rest are set to 0 (e.g. [0.1, 0.1, 0.4, 0.4] $\rightarrow$ [0, 0, 1, 1]). This converted matrix functions as the final regularization matrix as shown in right hand side of Figure 1. We set $K = 50$ in our experiments.[2] An in-house LDA implementation[3] is used for training $\Phi_{[1:K,:]}$, with 1,000 iterations.

3.2 Annotated Regularizer

We use three sources for training annotated regularizers in this work. Two of the sources are semantic lexicons, namely, the Paraphrase Database (PPDB)[4] (Ganitkevitch et al., 2013) and synonyms in the WordNet (WN$_{syn}$)[5] (Miller, 1995). They are used in the word similarity task. The third source is a semantic dictionary, SentiWordNet 3.0 (SWN) (Baccianella et al., 2010), which is used in the sentiment classification task. All of the three sources were created with annotation efforts, where either lexical or semantic relations were provided by human experts beforehand.

Before constructing the regularizer, we need encode each word in the sources as a vector according to its relations to other words or predefined information. For PPDB and WN$_{syn}$, we use them in different ways for joint learning and retrofitting. In order to optimize the efficiency in joint learning, we compress the word relations with topic representations. We use an LDA learner to get topic models for the lexicons[6], with $K = 50$. Therefore, the word relations are transferred into topic distributions that are learned from their cooccurrences defined in the lexicon. The way we construct regularization matrix may be lossy, risking losing information that is explicitly delivered in the lexicon. However, it provides us effective encodings for words, and also yields better learning performance empirically in our experiments. In retrofitting, we directly use words' adjacent matrices extracted from their relations defined in the lexicons, then take the adjacent vector for each word as the regularization vector.

The SWN includes 83K words (147K words and phrases in total). Every word in SWN has two scores for its degree towards positive and negative polarities. For example, the word "pretty" receives 0.625 and 0 for positive and negative respectively, which means it is strongly associated with positive sentiment. The scores range from 0 to 1 with step

[2]We experimented with other numbers for K, and their performance didn't vary too much when $K > 40$. We didn't include this comparison due to the similar results.

[3]It is a Markov Chain Monte Carlo (MCMC) based LDA using Gibbs sampling.

[4]We use PPDB-XL in this paper.

[5]We use WN$_{syn}$ because in our experiment only using synonyms perform better than using synonyms, hypernyms and hyponyms.

[6]The lexicons are organized in the similar way as in Faruqui et al. (2015), where synonyms are grouped together and treated as a document for LDA learning.

Embeddings		MEN-3k		SimLex-999		WordSim-353	
		γ	ρ	γ	ρ	γ	ρ
LDA		57.17	58.86	20.39	22.12	55.48	54.81
CBOW		62.93	65.84	28.34	28.31	68.50	66.67
Yu and Dredze (2014)	+PPDB	65.35	65.84	35.56	33.30	72.75	72.43
	+WN$_{syn}$	65.20	65.74	36.15	33.65	**72.79**	**72.58**
This work	+LDA	**67.33**	**69.51**	29.79	29.78	71.19	69.58
	+PPDB	65.25	66.87	**36.43**	33.28	69.45	68.89
	+WN$_{syn}$	64.42	66.98	33.86	**33.69**	66.13	67.11
SG		64.79	66.71	26.97	26.59	68.88	67.80
Kiela et al. (2015)	+PPDB	61.13	60.04	36.47	34.29	70.14	68.76
	+WN$_{syn}$	57.02	59.84	29.02	29.99	63.61	61.22
This work	+LDA	65.02	65.32	25.19	24.04	66.16	69.21
	+PPDB	**70.83**	**71.35**	**37.10**	35.72	**73.94**	**73.11**
	+WN$_{syn}$	66.58	68.14	36.72	**35.91**	68.50	67.90

Table 1: Word similarity results for **joint learning** on three datasets in terms of Pearson's coefficient correlation (γ) and Spearman's rank correlation (ρ) in percentages. Higher score indicates better correlation of the model with respect to the gold standard. Bold indicates the highest score for each embedding type.

of 0.125 for both positive and negative polarities. Therefore there are 9 different degrees for a word to be annotated for the two sentiments. For encoding this dictionary, we design a 18-dimension vector, in which the first 9 dimension represents the positive sentiment while the last 9 for negative sentiment. A word is thus encoded into a binary form where the corresponding dimension is set to 1 with others 0. For the aforementioned word "*pretty*", its encoded vector will be "*000001000 000000000*", in which the score 0.625 of positive activates the 6th dimension in the vector. In doing so, we form a $83K \times 18$ regularization matrix for the SWN dictionary.

4 Experiments

The resulting word embeddings based on joint learning as well as retrofitting are evaluated intrinsically and extrinsically. For intrinsic evaluation, we use word similarity benchmark to directly test the quality of the learned embeddings. For extrinsic evaluation, we use sentiment analysis as a downstream task with different input embeddings. Regularizers based on LDA, PPDB and WN$_{syn}$ are used in word similarity experiment, while SentiWordNet regularization is used in sentiment analysis. The experimental results will be discussed in §4.1 and §4.2.

We experiment with three learning paradigms, namely CBOW, SG and GloVe. GloVe is only tested in retrofitting since our regularizer is not compatible with GloVe learning objective in joint learning. In all of our retrofitting experiments, we only train the regularizer with one iteration, consistent with Kiela et al. (2015).

The base corpus that we used to train initial word embeddings is from the latest articles dumped from Wikipedia and newswire[7], which contains approximately 8 billion words. When training on this corpus, we set the dimension of word embeddings to be 200 and cutoff threshold of word frequency threshold to be 5 times of occurrence. These are common setups shared across the following experiments.

4.1 Word Similarities Evaluation

We use the MEN-3k (Bruni et al., 2012), SimLex-999 (Hill et al., 2015) and WordSim-353 (Finkelstein et al., 2002) datasets to perform quantitative comparisons among different approaches to generating embeddings. The cosine scores are computed between the vectors of each pair of words in the datasets[8]. The measures adopted are Pearson's coefficient of product-moment correlation (γ) and Spearman's rank correlation (ρ), which reflect how

[7]This corpus is constructed by the script demo-train-big-model-v1.sh from https://storage.googleapis.com/google-code-archive-source/v2/code.google.com/word2vec/source-archive.zip

[8]For LDA embeddings (topic distributions), we tried Jenson-Shannon divergence, which is much worse than cosine scores in measuring the similarity. Therefore we still use cosine for LDA embeddings.

Embeddings		MEN-3k		SimLex-999		WordSim-353	
		γ	ρ	γ	ρ	γ	ρ
GloVe		66.84	66.97	28.87	27.52	59.78	61.46
Faruqui et al. (2015)	+PPDB	66.98	67.04	29.25	28.25	61.44	63.35
	+WN$_{syn}$	64.29	63.92	27.32	24.39	57.40	58.88
This work	+LDA	59.65	60.23	22.25	22.70	55.65	57.57
	+PPDB	**68.99**	**68.99**	**31.35**	**29.85**	**62.31**	**63.96**
	+WN$_{syn}$	66.72	66.84	29.78	28.47	59.62	61.34
CBOW		62.93	65.84	28.34	28.31	68.50	66.67
Yu and Dredze (2014)	+PPDB	65.08	65.52	36.16	34.01	**72.75**	72.39
	+WN$_{syn}$	**65.34**	65.77	35.68	33.33	72.72	**72.74**
Faruqui et al. (2015)	+PPDB	65.07	67.55	37.07	35.02	71.76	71.18
	+WN$_{syn}$	63.71	66.44	30.15	29.83	71.24	69.39
This work	+LDA	50.07	56.64	21.47	23.01	41.56	47.27
	+PPDB	65.30	**67.68**	**37.34**	**35.74**	72.01	72.05
	+WN$_{syn}$	63.89	66.74	33.96	33.82	68.70	66.91
SG		64.79	66.71	26.97	26.59	68.88	67.80
Kiela et al. (2015)	+PPDB	**67.38**	69.05	32.49	31.84	71.59	69.82
	+WN$_{syn}$	64.58	67.02	29.43	28.12	69.15	68.36
Faruqui et al. (2015)	+PPDB	65.44	67.02	34.12	33.72	71.24	70.31
	+WN$_{syn}$	65.65	66.71	28.25	27.61	70.21	69.47
This work	+LDA	64.02	65.33	24.64	24.28	59.43	60.60
	+PPDB	67.17	**69.09**	**34.93**	**34.57**	**72.63**	**71.15**
	+WN$_{syn}$	65.62	67.38	29.96	29.82	69.70	68.91

Table 2: Word similarity results for **retrofitting** on three datasets in terms of Pearson's coefficient correlation (γ) and Spearman's rank correlation (ρ) in percentages. Higher score indicates better correlation of the model with respect to the gold standard. Bold indicates the highest score for each embedding type.

close the similarity scores to human judgments.

For both joint learning and retrofitting, we test our approach with using PPDB and WN$_{syn}$ as the prior knowledge applied to our regularizer. Considering that LDA can be regarded as soft clustering for words, it is very hard to present words with deterministic relations like in PPDB and WN$_{syn}$, therefore we do not apply retrofitting on LDA results for previous studies.

The evaluation results are shown in Table 1 and Table 2 for joint learning and retrofitting, respectively. Each block in the tables indicates an embedding type and its corresponding enhancement approaches. For comparison, we also include the results from the approaches proposed in previous studies, i.e., Yu and Dredze (2014)[9] for CBOW, Kiela et al. (2015)[10] for SG and Faruiqui et al. (2015)[11] for all initial embeddings. Their settings are equal to that used in our approach.

Table 1 shows that directly using LDA topic distributions as embeddings can give reasonable results for word similarities. Because LDA captures word co-occurrences globally so that words share similar contexts are encoded similarly via topic distributions. This is a good indication showing that LDA could be a useful guidance to help our regularize to incorporate global information.

For other joint learning results in Table 1, our approach shows significant gain over the baselines, the same for the approaches from previous studies (Yu and Dredze, 2014; Faruqui et al., 2015). However, using WN$_{syn}$ in Kiela et al. (2015) does not help, this may owe to the fact that using the words defined in WN$_{syn}$ as contexts will affect the real context learning and thus deviate the joint objective function. Interestingly, using LDA in regularizer significantly boosts the performance on MEN-3k, even better than that with using semantic lexicons. The reason might be that LDA enhances word embeddings with the relatedness inherited in topic distributions.

[9]https://github.com/Gorov/JointRCM

[10]We re-implemented their approach in our own code.

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

For retrofitting, Table 2 shows that our approach demonstrates its effectiveness for enhancing initial embeddings with prior knowledge. It performs consistently better than all other approaches in a wide range of settings, including three embedding types on three datasets, with few exceptions. Since retrofitting only updates those words in the external sources, e.g., LDA word list or lexicons, it is very sensitive to the quality of the corresponding sources. Consequently, it can be observed from our experiment that unannotated knowledge, i.e., topic distributions, is not an effective source as a good guidance. In contrast, PPDB, which is of high quality of semantic knowledge, outperforms other types of information in most cases.

4.2 Sentiment Classification Evaluation

We perform sentiment classification on the IMDB review data set (Maas et al., 2011), which has 50K labeled samples with equal number of positive and negative reviews. The data set is pre-divided into training and test sets, with each set containing 25K reviews. The classifier is based on a bi-directional LSTM model as described in Dai and Le (2015), with one hidden layer of 1024 units. Embeddings from different approaches are used as inputs for the LSTM classifier. For determining the hyper-parameters (e.g., training epoch and learning rate), we use 15% of the training data as the validation set and we apply early stopping strategy when the error rate on the validation set starts to increase. Note that the final model for testing is trained on the entire training set.

As reported in Table 3, the embeddings trained by our approach work effectively for sentiment classification. Both joint learning and retrofitting with our regularizer outperform other baseline approaches from previous studies, with joint learning being somewhat better than retrofitting. Overall, our joint learning with CBOW achieves the best performance on this task. A ten-partition two-tailed paired t-test at $p < 0.05$ level is performed on comparing each score with the baseline result for each embedding type. Considering that sentiment is not directly related to word meaning, the results indicate that our regularizer is capable of incorporating different type of knowledge for a specific task, even if it is not aligned with the context learning. This task demonstrates the potential of our framework for encoding external knowledge and using it to enrich the representations of words, without the requirement to build a task-specific, customized model.

Embeddings		Accuracy
Maas et al. (2011)		88.89
GloVe		90.66
Faruqui et al. (2015)	+Retro	90.43
This work	+Retro	**90.89**
CBOW		91.29
Yu and Dredze (2014)	+Joint	91.14
	+Retro	90.71
Faruqui et al. (2015)	+Retro	90.77
This work	+Joint	**92.09***
	+Retro	91.81*
SG		91.30
Faruqui et al. (2015)	+Retro	91.03
Kiela et al. (2015)	+Joint	91.45
	+Retro	91.14
This work	+Joint	**92.07***
	+Retro	91.42

Table 3: Sentiment classification results on IMDB data set (Maas et al., 2011). Bold indicates the highest score for each embedding type. * indicates t-test significance at $p < 0.05$ level when compared with the baseline.

5 Related Work

Early research on representing words as distributed continuous vectors dates back to Rumelhart et al. (1986). Recent previous studies (Collobert and Weston, 2008; Collobert et al., 2011) showed that, the quality of embeddings can be improved when training multi-task deep models on task-specific corpora, domain knowledge that is learned over the process. Yet one downside is that huge amounts of labeled data is often required. Another methodology is to update embeddings by learning with external knowledge. Joint learning and retrofitting are two mainstreams of this methodology. Leveraging semantic lexicons (Yu and Dredze, 2014; Bian et al., 2014; Faruqui et al., 2015; Liu et al., 2015a; Kiela et al., 2015; Wieting et al., 2015; Nguyen et al., 2016) or word distributional information (Maas et al., 2011; Liu et al., 2015b) has been proven as effective in enhancing word embeddings, especially for specific downstream tasks. Bian et al. (2014) proposed to improve embedding learning with different kinds of knowledge, such as morphological, syntactic and semantic information. Wieting et al. (2015) improves embeddings by leveraging paraphrase pairs from the PPDB for learning phrase embeddings in the paraphrasing task. In a similar way, Hill et al. (2016) uses learned word embeddings as supervised knowledge for learning phrase embeddings.

Although our approach is conceptually similar to previous work, it is different in several ways. For leveraging unlabeled data, the regularizer in this work is different from applying topic distributions as word vectors (Maas et al., 2011) or treating topics as conditional contexts (Liu et al., 2015b). For leveraging semantic knowledge, our regularizer does not require explicit word relations as used in previous studies (Yu and Dredze, 2014; Faruqui et al., 2015; Kiela et al., 2015), but takes encoded information of words. Moreover, in order to appropriately learn the encoded information, we use trajectory softmax to perform the regularization. As a result, it provides a versatile data structure to incorporate any vectorized information into embedding learning. The above novelties make our approach versatile so that it can integrate different types of knowledge.

6 Conclusion and Future Work

In this paper we proposed a regularization framework for improving the learning of word embeddings with explicit integration of prior knowledge. Our approach can be used independently as a retrofitter or jointly with CBOW and SG to encode prior knowledge. We proposed trajectory softmax for learning over the regularizer, which can greatly reduce the space complexity compared to hierarchical softmax using the Huffman coding tree, which enables the regularizer to learn over a long vector. Moreover, the regularizer can be constructed from either unlabeled data (e.g., LDA trained from the base corpus) or manually crafted resources such as a lexicon. Experiments on word similarity evaluation and sentiment classification show the benefits of our approach.

For the future work, we plan to evaluate the effectiveness of this framework with other types of prior knowledge and NLP tasks. We also want to explore different ways of encoding external knowledge for regularization.

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), Valletta, Malta.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.* 3:1137–1155.

Jiang Bian, Bin Gao, and Tie-Yan Liu. 2014. Knowledge-Powered Deep Learning for Word Embedding. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases - Volume 8724*. New York, NY, USA, ECML PKDD 2014, pages 132–148.

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

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam Khanh Tran. 2012. Distributional Semantics in Technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Jeju Island, Korea, pages 136–145.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*. ACM, New York, NY, USA, ICML '08, pages 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research* 12:2493–2537.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised Sequence Learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. pages 3079–3087.

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*. Denver, Colorado, pages 1606–1615.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing Search in Context: the Concept Revisited. *ACM Transaction on Information Systems* 20(1):116–131.

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. Atlanta, Georgia, pages 758–764.

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* 4:17–30.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating Semantic Models with Genuine Similarity Estimation. *Computational Linguistics* 41(4):665–695.

Douwe Kiela, Felix Hill, and Stephen Clark. 2015. Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 2044–2048.

Quan Liu, Hui Jiang, Si Wei, Zhen-Hua Ling, and Yu Hu. 2015a. Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China, pages 1501–1511.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015b. Topical Word Embeddings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI'15, pages 2418–2424.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA, pages 142–150.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint* abs/1301.3781.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting Similarities among Languages for Machine Translation. *arXiv preprint* abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013c. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013d. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Atlanta, Georgia, pages 746–751.

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

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany, pages 454–459.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pages 1532–1543.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning Representations by Backpropagating Errors. *Nature* pages 533–536.

Peter D. Turney and Patrick Pantel. 2010. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research* 37(1):141–188.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. *arXiv preprint* abs/1502.05698.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From Paraphrase Database to Compositional Paraphrase Model and Back. *Transactions of the Association for Computational Linguistics* 3:345–358.

Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. RC-NET: A General Framework for Incorporating Knowledge into Word Representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, New York, NY, USA, CIKM '14, pages 1219–1228.

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)*. Baltimore, Maryland, pages 545–550.