Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Complementary Learning of Word Embeddings

Yan Song, Shuming Shi
Tencent AI Lab
{clksong, shumingshi}@tencent.com

Abstract

Continuous bag-of-words (CB) and skip-gram (SG) models are popular approaches to training word embeddings. Conventionally they are two standing-alone techniques used individually. However, with the same goal of building embeddings by leveraging surrounding words, they are in fact a pair of complementary tasks where the output of one model can be used as input of the other, and vice versa. In this paper, we propose complementary learning of word embeddings based on the CB and SG model. Specifically, one round of learning first integrates the predicted output of a SG model with existing context, then forms an enlarged context as input to the CB model. Final models are obtained through several rounds of parameter updating. Experimental results indicate that our approach can effectively improve the quality of initial embeddings, in terms of intrinsic and extrinsic evaluations.

1 Introduction

Word embeddings have shown to be effective in many natural language processing (NLP) tasks [Collobert *et al.*, 2011]. Conventionally, word embeddings are obtained by conducting unsupervised learning over large corpus [Mikolov *et al.*, 2013a; Pennington *et al.*, 2014]. The quality of the resulting embeddings is highly dependent on the size and quality of the training corpus, whether or not additional knowledge is considered, as well as how the learning architecture is designed.

To learn better embeddings, previous work adapted initial embeddings to specific domains and tasks [Maas *et al.*, 2011], or incorporated external information to refine objective functions [Yu and Dredze, 2014; Kiela *et al.*, 2015] as well as retrofit the initial embeddings [Faruqui *et al.*, 2015; Kiela *et al.*, 2015]. However, a key factor that brings success to these methods depends on the availability of high quality semantic sources. In a low-resource language or a specific domain, obtaining sufficient data could be more challenging.

An alternative way to improve embeddings is to enhance the learning architecture. The architecture of current most popular approaches focuses on either predicting a word given its context (CB), or predicting the context given a word (SG). While these two learning paradigms are usually performed independently, they are in fact complementary to each other where the same problem is tackled from different directions and the output of one task can be used as input to the other. Motivated by this, in this paper, we propose an enhanced architecture that considers simultaneously the two paradigms as complementary tasks in a reinforced manner. Our hypothesis is that the quality of the resulting embeddings for each individual task can be improved based on the exchanged intermediate information. In detail, our approach involves several steps for each reinforcing iteration. To start, we take the output from an SG model, then combine it with existing context to form as input to a CB model. Subsequently, the parameters of the corresponding models are updated according to the expectation of our designed reward functions. We further propose a straightforward sampling strategy that aims to ensure introducing extra information to the learning process.

The resulted embeddings are evaluated by word similarity and downstream classification tasks. Experiments show that our approach can effectively enhance initial embeddings, which outperform embeddings learned with solely individual models as well as the other learning strategies such as EM and full-gradient method. Since our approach naturally fits in a retrofitting framework, we also demonstrate better effectiveness compared to previous retrofitters [Yu and Dredze, 2014; Faruqui *et al.*, 2015; Kiela *et al.*, 2015].

2 Background of CB and SG Models

Continuous bag-of-words (CB) and skip-gram (SG) models are popular approaches to training word embeddings in `word2vec` [Mikolov *et al.*, 2013a; 2013b], where the training is done in an unsupervised manner on text corpora. The fundamental basis the two models share is to build word embeddings by leveraging the relations between neighboring words. CB focuses on maximizing the likelihood that a word is predicted from its context while SG reversely predicts the context based on a given word, as illustrated in Figure 1.

Formally, for a given corpus with token set V, the CB model can be formulated as maximizing the likelihood

$$\mathcal{L}_{CB} = \frac{1}{|V|} \sum_{t=1}^{|V|} \log p(w_t \mid w_{t-c}^{t+c}), \ \ \forall\, w_t \in V \qquad (1)$$

where w_{t-c}^{t+c} refers to the context of word w_t and c defines the window size. A projection layer takes the sum of

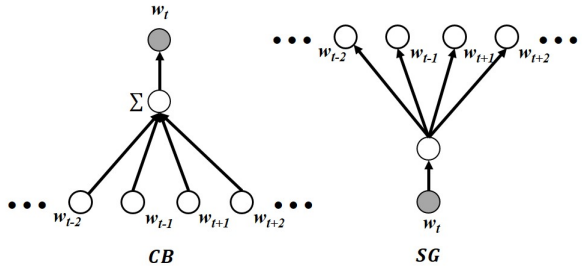


Figure 1: Illustration of CB and SG models.

the embeddings from the context words, resulting in $h = \sum_{0<|i|\leq c} v_{w_{t+i}}$ over all context words from w_{t-c} to w_{t+c}. Then the probability of predicting a given word w_t with context w_{t-c}^{t+c} is defined as

$$p(w_t \mid w_{t-c}^{t+c}) = \frac{exp({v'_{w_t}}^\top h)}{\sum_{w_t \in V} exp({v'_{w_t}}^\top h)} \quad (2)$$

where v_{w_t} is the embedding for w_t, and v and v' refer to input and output embeddings, respectively.

On the contrary, the SG model is to predict the context with a given word, formulated as maximizing the likelihood

$$\mathcal{L}_{SG} = \frac{1}{|V|} \sum_{t=1}^{|V|} \sum_{0<|i|\leq c} \log p(w_{t+i} \mid w_t), \ \forall\, w_t \in V \quad (3)$$

Conventionally, the projection layer copies the embedding of the input word, $h = v_{w_t}$. The probability to predict context word is thus estimated by

$$p(w_{t+i} \mid w_t) = \frac{exp({v'_{w_{t+i}}}^\top v_{w_t})}{\sum_{w_{t+i} \in V} exp({v'_{w_{t+i}}}^\top v_{w_t})} \quad (4)$$

For a large vocabulary $\mathcal{V}$, `word2vec` uses hierarchical softmax or negative sampling [Mikolov *et al.*, 2013b] to address the computational complexity that requires $|\mathcal{V}| \times d$ matrix multiplication, where d refers to embedding dimension.

3 The Proposed Framework

3.1 Overview

Essentially the input and output of CB and SG are complementary to each other as described in the previous section. The output of SG is naturally a good fit as input to CB and vice verse. Tackling the same task from different sides, we hypothesize that the two models can work together collaboratively with one enhancing the estimates based on information propagated from the other. In doing so, we formulate our model following the reinforcement learning paradigm, and consider CB and SG as two agents learning over a corpus composed of words that are treated as states. Thus embedding learning can be done iteratively where the two agents take turns making actions of predicting words with given context or predicting context with a given word. Meanwhile the actions taken are affected by the parameters of the two agents, which are essentially the word embeddings.[1]

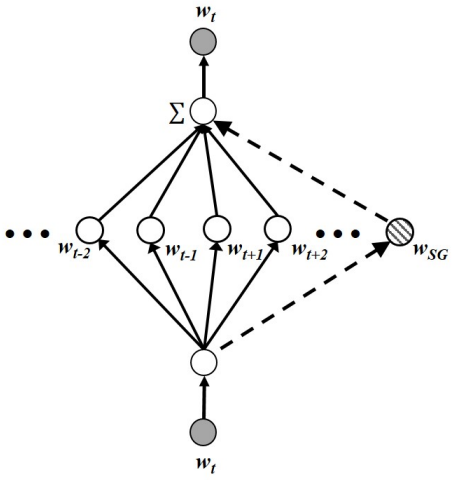


Figure 2: Illustration of CB and SG as complementary tasks.

Figure 2 illustrates our framework using CB and SG as complementary tasks to learn word embeddings. Starting from the SG model, an intermediate word w_{SG} is predicted and appended to the current context w_{t-c}^{t+c} as an additional token. Then the CB model in turn is learned with the enlarged context $w_{t-c}^{t+c} \cup w_{SG}$ to predict the current word w_t. Together the two steps form one loop of embedding update. Our model bases on the assumption that given a reasonable SG model being able to perform decent word prediction, w_{SG} should provide additional useful information to compliment existing context w_{t-c}^{t+c}. Consequently the CB model can be benefited from the extra context word and perform better prediction on w_t. The SG model is thus updated accordingly.

The rationale for our model design to start from SG rather than CB is two-fold. First, to immediately leverage the intermediate output, which is w_{SG} in our framework, the CB model can naturally expand its current context with the extra token and perform learning with the joint word bags, whereas the SG model expects to operate on single-word input. Second, since empirically SG performs better than CB on infrequent and rare words [Mikolov *et al.*, 2013b], starting from SG has the potential to initialize a state that produces more accurate intermediate output. Intuitively, our model is similar to a stacked autoencoder in a sense that the learning process is instructed to first learn $w_t \mapsto w_{t-c}^{t+c} \cup w_{SG}$, then followed by $w_{t-c}^{t+c} \cup w_{SG} \mapsto w_t$. Different from conventional autoencoders, the intermediate layer in our approach incorporates supervised information via w_{t-c}^{t+c}, which brings advantage to learning when w_{SG} is poorly generated from w_t.

[1] Alternative formulations of our approach is a full-gradient method over the intermediate context. However, modeling as reinforcement learning makes the approach more flexible since the process of adding new knowledge/actions into objective functions can be standardized and unified. We will show the superiority of our approach to the full-gradient method in later sections.

Algorithm 1: Complementary learning of word embeddings using CB and SG.

Input : $S_1^N, \Theta_{CB}, \Theta_{SG}, \gamma_1, \gamma_2$
Output: Θ_{CB}, Θ_{SG}
1 **for** *iteration = 1 to max_iter* **do**
2 **foreach** *s from* S_1^N **do**
3 **foreach** *sampled instance* $C \in s$ **do**
4 Extract center word w_t and its context c, where $C = c \cup w_t$;
5 Obtain w_{SG} by $w_{SG} = \arg\max_{\Theta_{SG}} p(w_{SG} \mid w_t)$
6 Compute SG reward for w_{SG} $r_{SG} = \frac{1}{|c|}\sum_{w_j \in c} \log p(w_j \mid w_t; \Theta_{SG}) - \log p(w_{SG} \mid w_t; \Theta_{SG})$;
7 Compute CB reward for w_{SG} $r_{CB} = \log p(w_t \mid c \cup w_{SG}; \Theta_{CB})$
8 Combine rewards $r = \lambda r_{SG} + (1-\lambda) r_{CB}$;
9 Compute $\nabla_{\Theta_{SG}} \hat{E}[r]$, $\nabla_{\Theta_{CB}} \hat{E}[r]$;
10 Update $\Theta_{SG} \leftarrow \Theta_{SG} + \gamma_1 \nabla_{\Theta_{SG}} \hat{E}[r]$, $\Theta_{CB} \leftarrow \Theta_{CB} + \gamma_2 \nabla_{\Theta_{CB}} \hat{E}[r]$;
11 **end**
12 **end**
13 **end**

3.2 Learning Process

Algorithm 1 describes in detail the implementation of our approach. A corpus S_1^N of N sentences (or text fragments) and discount learning rates γ_1 and γ_2 are required as input. Assuming there exist initial CB and SG model, namely, Θ_{CB} and Θ_{SG}, the learning process kicks off by selecting a training sample C comprised of word w_t and its context c in running text (line 4 in Algorithm 1). Note that the lowercased c is the set of words from the capitalized C excluding w_t; that is, $c = w_{t-i}^{t-1} \cup w_{t+1}^{t+i}$ where i is the window size.

For each training sample, a predicted word w_{SG} is obtained from the SG model, which is regarded as the action taken by the SG agent (line 5). Since SG is essentially a language model, w_{SG} can be generated by choosing a word from the vocabulary with the probability in Eq. 4 maximized, given Θ_{SG}. When Θ_{SG} is trained with hierarchical softmax, choosing w_{SG} can be done effectively with hierarchical classification as in Yu and Dredze [2014]. Intuitively, we want w_{SG} to be close to c since c is the original context for w_t, but in the meantime should be different from words already in c to avoid redundancy. Thus we define reward r_{SG} to be the difference between the log likelihood of word generation, mathematically as $\frac{1}{|c|}\sum_{w_j \in c} \log p(w_j|w_t) - \log p(w_{SG}|w_t; \Theta_{SG})$, with respect to w_t, w_{SG} and c (line 6). In this way, a better estimate on w_{SG} will result in a higher reward r_{SG}.

Then we add w_{SG} as an extra word to c and use CB to predict the word w_t, which is the action taken by the CB agent. The reward for this action is defined as $r_{CB} = \log p(w_t|c \cup w_{SG})$ (line 7). Subsequently, we have the final reward $r = \lambda r_{SG} + (1-\lambda) r_{CB}$ with hyper-parameter λ adjusting the contribution of different sub-rewards (line 8). Then Θ_{SG} and Θ_{CB} can be updated according to the gradients of the rewards w.r.t. model parameters (lines 9 and 10). The entire algorithm is performed on a corpus repeatedly until the maximum iteration number (max_iter) is reached.

3.3 Parameter Estimation

Following reinforcement learning theory, the expectation $\hat{E}[r]$ for reward r is usually adopted for optimizing agents' strategies. To compute $\hat{E}[r]$ for our approach, we note that steps 6 to 10 in Algorithm 1 must repeat over all $|c|+1$ words, i.e., $c \cup w_{SG}$. Then, by defining $C' = c \cup w_{SG}$, the expectation can be written as $\hat{E}[r] = \frac{1}{|C'|}\sum_{w \in C'} r_w$. We can prove that $\hat{E}[r] = r$.[2] As a result, the learning process can be simplified to solely focus on w_{SG} disregarding other words $w \in c$. Then the policy gradients for Θ_{SG} and Θ_{CB} can be computed based on r as described in Sutton *et al.* [2000]:

$$\nabla_{\Theta_{SG}} \hat{E}[r] = \frac{r}{|C'|} \sum_{w \in C'} \nabla_{\Theta_{SG}} \log p(w \mid w_t; \Theta_{SG}) \quad (5)$$

$$\nabla_{\Theta_{CB}} \hat{E}[r] = (1-\lambda) \nabla_{\Theta_{CB}} \log p(w_t \mid C'; \Theta_{CB}) \quad (6)$$

where updating SG model takes into account the feedback from the reward r, which encompasses the estimations for both SG and CB models. For CB model, its update is affected by the introduction of new context word w_{SG}. During the learning process, the constant improvement of Θ_{SG} tends to generate more precise w_{SG} for a given input w_t, thus enhancing CB model with the better, enlarged context.

To combine the two learning paradigms CB and SG, we can choose to use solely the predicted w_{SG} as the intermediate result. While that is an option, we, however, instruct the intermediate layer to include both w_{SG} and the original word context c, which can be viewed as pseudo output of SG model. In this way, the learning process tends to be more reliable since original reference is kept all the time, and potentially better estimation of the gradients is expected when predicting w_{SG}.

[2] Let $f(w_i) = \log p(w_i \mid w_t; \Theta_{SG})$, we have $r_{SG} = \frac{1}{|c|}\sum_{w_i \in c} f(w_i) - f(w_{SG})$. While using the notation r for the reward for w_{SG} and $r(w_i)$ for the context words, we have

$$\begin{aligned}
\hat{E}[r] &= \frac{1}{|C'|}\sum_{w_i \in C'} r(w_i) = \frac{1}{|C'|}\Big(\sum_{w_i \in c} r(w_i) + r\Big) \\
&= \frac{\lambda}{|C'|}\Big(\sum_{w_i \in c} r_{SG}(w_i) + r_{SG}\Big) + \frac{1-\lambda}{|C'|}\Big(\sum_{w_i \in C'} r_{CB}(w_i) + r_{CB}\Big) \\
&= \frac{\lambda}{|C'|}\Big(\sum_{w \in c} r_{SG}(w_i) + r_{SG}\Big) + (1-\lambda) r_{CB} \\
&= \frac{\lambda}{|C'|}\Big[\sum_{w \in c}\Big(\frac{1}{|c|}\sum_{w_i \in c} f(w_i) - f(w_{SG})\Big) + \frac{1}{|c|}\sum_{w_i \in c} f(w_i) - f(w_{SG})\Big] \\
&\quad + (1-\lambda) r_{CB} \\
&= \lambda \frac{|c|+1}{|C'|}\Big[\frac{1}{|c|}\sum_{w_i \in c} f(w_i) - f(w_{SG})\Big] + (1-\lambda) r_{CB} \\
&= \lambda r_{SG} + (1-\lambda) r_{CB} \\
&= r \qquad \text{Q.E.D.}
\end{aligned}$$

Embeddings	Trained on the small corpus			Trained on the large corpus		
	MEN-3k	Simlex-999	WS-353	MEN-3k	Simlex-999	WS-353
GloVe	48.84	18.87	59.78	66.08	26.12	61.44
CB	53.57	19.40	59.02	64.70	25.42	64.96
SG	60.28	24.97	63.03	65.70	25.57	65.60
CB $\oplus$ SG	60.63	25.34	63.01	65.46	25.86	65.29
Avg(CB, SG)	58.89	24.05	62.91	64.75	25.42	64.81
FGSG$_R$	59.75	24.15	63.14	65.65	25.78	65.71
FGSG$_P$	60.01	24.84	63.55	66.21	25.98	65.92
CTCB$_R$	51.20	19.79	55.19	64.67	24.44	63.02
CTSG$_R$	61.80	25.32	59.61	66.12	25.88	64.97
CTCB$_P$	53.33	20.69	58.17	66.80	26.33	63.88
CTSG$_P$	**63.73**	**26.28**	**65.04**	**67.00**	**27.69**	**66.81**

Table 1: Word similarity by Spearman's rank correlation ($\rho \times 100$). Best results are in bold.

3.4 Sampling Strategy

A key factor that affects the performance of our learned model is how effectively the training instances are sampled. We adopt a simple sampling strategy, where an instance C is selected for training only if $w_{SG} \cap c = \emptyset$. That is, our training set considers only instances where w_{SG} is not already in c and disregards the rest. The reason for doing so is that including something original context c already contains does not provide additional information. By the definition of r_{SG}, in the case of $w_{SG} \in c$, it leads to the learning process restricted to evaluate how good is c according to Θ_{SG}. Avoiding the samples where $w_{SG} \in c$ ensures the interaction between w_{SG} and c and thus makes the learning process more effective. Note as a special case, if learning without w_{SG}, $r_{SG} = \frac{1}{|c|}\sum_{w_j \in c} \log p(w_j|w_t)$, $r_{CB} = \log p(w_t|c)$, our approach is equivalent to training standard SG and CB models.

4 Experiments

In this section, we first study and characterize how training converges in terms of log likelihood for different embedding learning approaches. Then, we consider two groups of experiments to intrinsically and extrinsically evaluate the effectiveness of embeddings from an application point of view. We conduct the intrinsic evaluation based on word similarity measurement, and the extrinsic evaluation based on classification benchmarks. We prepare the latest dump of Wikipedia articles[3] as the base corpus for training word embeddings, which contains approximately 2 billion word tokens. All baseline and our embedding models are trained with the same hyper-parameters, i.e., 200 dimensions, 5 as the word frequency cutoff, a windows size of 5 words, $2 \sim 4$ iterations, using hierarchical softmax as learning strategy. This setting is used throughout all experiments. We term our approach as CTCB and CTSG, standing for embeddings from CB and SG model learned based on the proposed framework.

To conduct a comprehensive comparison, we also try full-gradient approach with the same complementary structure illustrated in Figure 2, however, without the predicted word w_{SG} since it does not require intermediate output for reward measurement. In this approach, the gradients pass through the connections between two models. As a result, the SG model (marked as FGSG) in the full-gradient approach accumulates all gradient from CB and itself, while the CB model is identical to its original counterpart because there is no extra information provided for it. Thus in later sections we do not report the result of the CB model in the full-gradient approach.

[3]https://dumps.wikimedia.org/enwiki/latest/

4.1 Performance Analysis

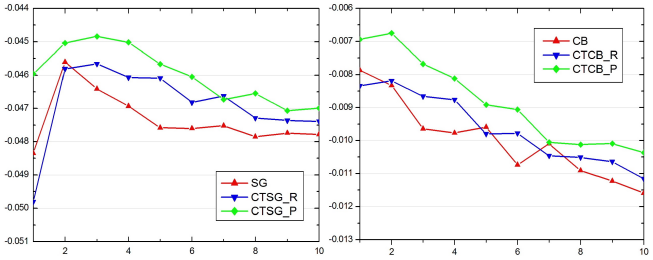
Several analyses are conducted to understand our proposed models. The first is to investigate how our model performs with different initial parameters. the analysis of the log likelihood at each training iteration is presented in Figure 3, where embedding models are compared separately according to their types, i.e., SG and CB. R and P refer to the models start from random initialization and pre-trained embeddings, respectively. In reducing the impact from randomness, each point in the figures is computed from the average of five runs.

Initial parameters are vital in reinforcement learning [Sutton and Barto, 1998]. Since our approach takes w_{SG} generated from the initial SG model, the reward highly depends on how good the estimation is over w_t and c with w_{SG}. When starting from random initialization, w_{SG} is less likely to be reliable. In this case, model updating is negatively affected by the inaccurate predicted w_{SG}. This may explain why, in Figures 3 (a) and (b), standard SG and CB models (i.e., red line) in fact outperform complementary learning with random initial embeddings (i.e., blue line) in the first iteration. With pre-trained embeddings, our approach (i.e., green line) achieves the highest log likelihood at each iteration compared to others. The final model adopted for the rest of sections is trained with 2-3 iterations, where it reaches its peak performance.

The second analysis is for training efficiency, which is an important factor affecting how our model can be applied to real applications. Observations show that the training time of our complementary learning is slightly longer than the total of training a CB model and an SG model, which is acceptable since there is a generation step in it where w_{SG} is produced.

4.2 Word Similarities

We use the MEN-3k [Bruni *et al.*, 2012], Simlex-999 [Hill *et al.*, 2015] and WS-353 [Finkelstein *et al.*, 2002] data sets



(a) SG (b) CB

Figure 3: Comparisons of log likelihood (Y-axis) against training iterations (X-axis) for SG and CB models.

to perform quantitative comparisons in the word similarity evaluation. In order to test the effectiveness of our approach, we conduct embedding learning on two corpora of different sizes. We create a small corpus by sampling 1/1000 size of the base corpus, which is referred to as the large corpus. Initial CB and SG embeddings are trained on both corpora with the aforementioned settings, yielding 2M word types from the large corpus and 113K word types from the small corpus.

Basic Comparison

Results of word similarity in terms of Spearman's rank correlation ρ on both corpora are shown in Table 1, where the best performing approach is in bold. Although there are well-performed approaches, e.g., Iacobacci *et al.* [2015], they have prerequisites such as reconstructing training data with extra information. Since we aim to compare our approach with its related models or other prevailing techniques that can learn from the raw corpus, our baselines include GloVe, the original CB and SG models, the concatenated embeddings of the two (i.e., CB $\oplus$ SG), the average of CB and SG embeddings, as well as the full-gradient SG models. We experiment two ways of creating initial embeddings – randomized and pre-trained, subscripted with R and P respectively. From Table 1, SG outperforms CB across different corpora and data sets, which is consistent with previous findings by Mikolov *et al.* [2013b]. The result of concatenating SG and CB is in par with that of SG, suggesting the better of the two dominates the performance. By averaging the two base embeddings, the word similarity sits between the results of the individual, which meets our intuition. In all cases, FGSG performs similar to the original SG model. The reason behind this observation might be that, for the full-gradient method, the input, i.e., the context, feeding into CB and SG model are identical. Thus updating the FGSG is similar to repeatedly train original SG model on the same context. This proves the superiority of our approach to alternatives within the same framework.

Overall, the CB and SG models based on complementary learning outperform all baselines, with CTSG$_P$ consistently performing the best. Using pre-trained embeddings demonstrate to be more effective than randomized ones. Comparing the results on the two corpora, we see the effectiveness of our approach is applicable to collections of different sizes, with the improvements on small corpus somewhat larger.

Retrofitting

To further illustrate the effectiveness of our model, we follow prior work [Yu and Dredze, 2014; Faruqui *et al.*, 2015; Kiela *et al.*, 2015] on retrofitting, whose goal is essentially to improve the quality of initial embeddings via external knowledge. Our framework naturally fits into a retrofitting scenario since it can start from pretrained embeddings. In detail, we first train initial embeddings on a general corpus (i.e., the large corpus), and then leverage an external semantic source as guidance and perform complementary learning on it. We use PPDB [Ganitkevitch *et al.*, 2013] as our semantic source and follow the process as done in Faruqui *et al.* [2015] by clustering the paraphrase words together as a group. For example, *bretton*, *wood* and *timber* are put together after extracted from the paraphrase list in PPDB. Then we perform the complementary learning on each group, where for any word in a group, other words are served as its context.

Approach	MEN-3k	Simlex-999	WS-353
Yu and Dredze CB	65.68	30.41	70.39
Faruqui *et al.* CB	66.15	30.97	69.97
Faruqui *et al.* SG	65.97	28.39	68.02
Kiela *et al.* SG	66.34	26.25	65.92
CTCB	70.01	34.50	71.30
CTSG	**70.51**	**35.26**	**71.86**

Table 2: Comparison of word similarity results ($\rho \times 100$) with previous work for retrofitting embeddings using PPDB.

We compare our result with three previous studies on the large corpus, including relation constrained model based retrofitting [Yu and Dredze, 2014] for CB, graph based retrofitting [Faruqui *et al.*, 2015] for CB and SG, and skip-gram based retrofitting [Kiela *et al.*, 2015] for SG. The similarity results are compared in Table 2. Overall, our approach outperforms the models from previous studies as well as the baselines in Table 1 with a large margin, where the CTSG is consistently performing the best. The results indicate that our approach can be used as an effective retrofitter.

4.3 Classification Benchmarks

The extrinsic evaluation is conducted on text classification with four datasets: the 20Newsgroups (20NG)[4] for topic classification, ATIS [Hemphill *et al.*, 1990] for intent classification, TREC [Li and Roth, 2002] for question type classification and IMDB [Maas *et al.*, 2011] for sentiment classification. All datasets are organized following their standard split.

The experimental settings are described as follows. For 20NG, we follow previous work [Kiela *et al.*, 2015] to construct document level representations by averaging the embeddings from all words in a given document. A logistic regression classifier is then trained on top of the resulted document embeddings on the training set and evaluated on the test set. For ATIS and TREC, we use a bi-directional LSTM model with one hidden layer of 256 units as the classifier. For IMDB we follow the setting for ATIS and TREC, except the hidden layer is set to 1024 as suggested in Dai and Le [2015]. Embeddings from different approaches are used as input for the aforementioned classifiers. In addition to the models used previously, we add FastText [Joulin *et al.*, 2016] as a new

[4]The "bydate" version on the web site: http://qwone.com/~jason/20Newsgroups/

Embeddings	20NG	ATIS	TREC	IMDB
FastText$_T$	82.56	96.30	88.60	89.65
GloVe$_S$	82.45	96.30	90.20	90.08
GloVe$_T$	81.08	95.30	89.00	90.14
CB$_S$	82.33	96.08	90.00	90.20
SG$_S$	82.41	96.30	91.00	90.18
CB$_T$	82.82	95.86	90.20	91.14
SG$_T$	82.58	95.41	89.60	90.70
CB$_{S\oplus T}$	82.42	96.19	90.20	90.51
SG$_{S\oplus T}$	82.85	95.86	90.80	90.86
CB$_{Avg(S,T)}$	82.56	95.97	89.80	90.34
SG$_{Avg(S,T)}$	82.10	95.74	90.20	90.71
FGSG$_{S\to S}$	82.45	96.09	90.60	90.02
FGSG$_{T\to T}$	82.78	95.52	89.20	90.89
FGSG$_{S\to T}$	82.84	96.19	91.20	90.44
CTCB$_{S\to S}$	82.72	96.19	90.20	90.45
CTSG$_{S\to S}$	82.65	96.42	91.00	91.08
CTCB$_{T\to T}$	**83.49***	96.08	91.20	**92.23***
CTSG$_{T\to T}$	**83.27***	95.74	90.40	**92.05***
CTCB$_{T\to S}$	82.51	95.63	90.00	90.29
CTSG$_{T\to S}$	82.80	95.52	89.40	90.11
CTCB$_{S\to T}$	82.98*	**97.42***	**92.00***	91.09
CTSG$_{S\to T}$	82.90	**97.20***	91.20	90.84

Table 3: Results of classification benchmarks using different embeddings. * indicates t-test significance at $p < 0.05$ level.

baseline to learn in-domain embeddings with given labels.

We experiment with two setups to understand the effectiveness of our approach for in-domain and out-of-domain scenarios. In the first setup, the pre-trained and final embeddings are trained on the same corpus, which is either the in-domain data (20NG, ATIS, TREC, IMDB) or out-of-domain data (Wikipedia). The aim of this setup is to evaluate our approach with respect to its capability of enhancing the embeddings. In the second setup, the initial embeddings are pre-trained on one corpus, based on which the final embeddings are trained with complementary learning on another corpus. This setup is similar to the retrofitting task, to evaluate how good our approach is in capturing domain knowledge when the base models are trained elsewhere. We denote the out-of-domain corpus as S and the in-domain corpus as T. For example, in 20NG task, $S \to T$ refers to that initial embeddings are pre-trained on Wikipedia and learned with the complementary framework on 20NG, and $T \to S$ for the reverse.

The classification results are reported in Table 3. One obvious observation is that CB$_T$ and SG$_T$ tend to outperform CB$_S$ and SG$_S$ respectively when in-domain data is relatively large and in a contrary when in-domain data is limited. Therefore, we consider the in-domain runs as the baseline methods for 20NG and IMDB and out-of-domain runs as the baseline methods for ATIS and TREC, against which a ten-partition two-tailed paired t-test at $p < 0.05$ level is conducted and compared on other methods. The t-test is performed within each corresponding embedding type. That is, for example, we compare CTCB against CB$_T$ and CTSG against SG$_T$.

Consistent with the results on word similarity, the embeddings trained with complementary learning outperform the original ones in all settings. In detail, embeddings learned by the complementary framework outperform their baseline approaches in both in-domain (e.g., CTCB$_{T\to T}$ > CB$_T$) and out-of-domain settings (e.g., CTCB$_{S\to S}$ > CB$_S$), which is also true for FGSG models and other baselines such as FastText and GloVe. It suggests that taking the context in the target domain is an effective way to incorporate domain knowledge, while our approach is more effective in doing so.

Specifically, in-domain CTCB embeddings (CTCB$_{T\to T}$) tops all other results when there is enough in-domain data for embedding learning, and cross domain CTCB embeddings (CTCB$_{S\to T}$) perform the best when in-domain data is not enough, which indicates that data availability in the target domain is crucial in learning domain-oriented embeddings. We also consider baselines that are based on simple aggregated S and T embeddings. Interestingly, by averaging the two embeddings, the performance of SG model is somewhat better than either of the two individually for some datasets, e.g. IMDB, but is still lower than complementary learning. One possible reason leads to the results is that, the SG model with complementary learning provide extra context information for the CB model, so that for the classification task, the learned embeddings can have better representations for some contexts in the test data where they are not captured in the training data. This may explain the superiority of CTCB in all tasks, especially for the cross domain setting (CTCB$_{S\to T}$) where in-domain data is limited. For example, in the TREC classification task, we found that many "*what is*" queries are ambiguous and easy to be misclassified, such as "*what is Teflon?*" in which case "*Teflon*" is a new word never appears in the training data. The SG model trained on Wikipedia can generate "*titanium*" as w_{SG} to complement the context of "*Teflon*", where "*titanium*" appears in the training data and is associated with the same class of the test query. As a result, the embedding of "Teflon" is updated in our model with the above information and the query is thus correctly classified.

5 Related Work

Word representations have brought significant benefits to many NLP tasks and been extensively studied for years [Collobert *et al.*, 2011; Mikolov *et al.*, 2013a]. Approaches including multi-task learning [Collobert and Weston, 2008; Collobert *et al.*, 2011], language modeling [Mikolov *et al.*, 2010], and global matrix factorization [Pennington *et al.*, 2014] are well performed embedding learning techniques.

To enhance word embeddings, there are studies focusing on using additional data sources to adapt initial embeddings to specific domains or tasks [Maas *et al.*, 2011]. The task of joint learning focuses on improving embedding quality by incorporating external information and refining objective functions [Yu and Dredze, 2014; Kiela *et al.*, 2015]. There are also retrofitting approaches leverage word relations defined in semantic lexicons to help adjusting and refining embeddings [Faruqui *et al.*, 2015; Song *et al.*, 2017]. Although embedding quality can be improved through extra guidance, one issue in these approaches is that they had to rely on collecting large amounts of annotated sources, which sets a high bar for success. The approach proposed in this paper with comple-

mentary learning offers an effective solution to this issue.

6 Conclusion and Future Work

In this paper, we proposed a complementary learning framework to improve the quality of word embeddings in a reinforcing manner. The results outperform standard and aggregated embeddings in terms of word similarity and classification benchmarks. We showed that our approach can effectively capture knowledge from a different information source, e.g., semantic lexicons or in-domain data. Thus it can be used independently as a retrofitter to enhance word embeddings or a domain adaptation method that incorporates target domain knowledge into embeddings. Overall, this work proved that reinforcement learning can effectively help taking extra information into embedding learning. Moving forward, there are many ways in which this work could be extended, such as learning better embeddings for knowledge graph and utilizing the approach in multilingual and cross-lingual settings.

References

[Bruni *et al.*, 2012] Elia Bruni, Gemma Boleda, Marco Baroni, and Nam Khanh Tran. Distributional Semantics in Technicolor. In *Proceedings of ACL*, pages 136–145, Jeju Island, Korea, July 2012.

[Collobert and Weston, 2008] Ronan Collobert and Jason Weston. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of ICML*, pages 160–167, New York, NY, USA, 2008. ACM.

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

[Dai and Le, 2015] Andrew M. Dai and Quoc V. Le. Semi-supervised Sequence Learning. In *NIPS*, pages 3079–3087, December 2015.

[Faruqui *et al.*, 2015] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of NAACL-HLT*, pages 1606–1615, Denver, Colorado, May–June 2015.

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

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

[Hemphill *et al.*, 1990] Charles T. Hemphill, John J. Godfrey, and George R. Doddington. The ATIS Spoken Language Systems Pilot Corpus. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '90, pages 96–101, 1990.

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

[Iacobacci *et al.*, 2015] Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. SensEmbed: Learning Sense Embeddings for Word and Relational Similarity. In *Proceedings of ACL*, pages 95–105, Beijing, China, July 2015.

[Joulin *et al.*, 2016] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. *arXiv preprint*, abs/1607.01759, 2016.

[Kiela *et al.*, 2015] Douwe Kiela, Felix Hill, and Stephen Clark. Specializing Word Embeddings for Similarity or Relatedness. In *Proceedings of EMNLP*, pages 2044–2048, Lisbon, Portugal, September 2015.

[Li and Roth, 2002] Xin Li and Dan Roth. Learning Question Classifiers. In *COLING*, pages 1–7, 2002.

[Maas *et al.*, 2011] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of ACL*, pages 142–150, Portland, Oregon, USA, June 2011.

[Mikolov *et al.*, 2010] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent Neural Network based Language Model. In *INTERSPEECH*, pages 1045–1048, 2010.

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

[Mikolov *et al.*, 2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, pages 3111–3119. 2013.

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

[Song *et al.*, 2017] Yan Song, Chia-Jung Lee, and Fei Xia. Learning Word Representations with Regularization from Prior Knowledge. In *Proceedings of CoNLL*, pages 143–152, Vancouver, Canada, August 2017.

[Sutton and Barto, 1998] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[Sutton *et al.*, 2000] Richard S. Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *NIPS*, pages 1057–1063. MIT Press, 2000.

[Yu and Dredze, 2014] Mo Yu and Mark Dredze. Improving Lexical Embeddings with Semantic Knowledge. In *ACL*, pages 545–550, Baltimore, Maryland, June 2014.