Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings

Yan Song, Shuming Shi, Jing Li, Haisong Zhang
Tencent AI Lab
{clksong,shumingshi,ameliajli,hansonzhang}@tencent.com

Abstract

In this paper, we present directional skip-gram (DSG), a simple but effective enhancement of the skip-gram model by explicitly distinguishing left and right context in word prediction. In doing so, a direction vector is introduced for each word, whose embedding is thus learned by not only word co-occurrence patterns in its context, but also the directions of its contextual words. Theoretical and empirical studies on complexity illustrate that our model can be trained as efficient as the original skip-gram model, when compared to other extensions of the skip-gram model. Experimental results show that our model outperforms others on different datasets in semantic (word similarity measurement) and syntactic (part-of-speech tagging) evaluations, respectively.

1 Introduction

Word embedding and its related techniques have shown to be vital for natural language processing (NLP) (Bengio et al., 2003; Collobert and Weston, 2008; Turney and Pantel, 2010; Collobert et al., 2011; Weston et al., 2015; Song and Lee, 2017). The skip-gram (SG) model with negative sampling (Mikolov et al., 2013a,c) is a popular choice for learning word embeddings and has had large impact in the community, for its efficient training and good performance in downstream applications. Although widely used for multiple tasks, SG model relies on word co-occurrence within local context in word prediction but ignores further detailed information such as word orders, positions.

To improve original word embedding models, there are various studies leveraging external knowledge to update word embeddings with post processing (Faruqui et al., 2015; Kiela et al., 2015; Song et al., 2017) or supervised objectives (Yu and Dredze, 2014; Nguyen et al., 2016). However, these approaches are limited by reliable semantic resources, which are hard to obtain or annotate. To overcome such limitations, there are many approaches to further exploiting the characteristics of the running text, e.g., internal structure of the context. These approaches include enlarging the projection layer with consideration of word orders (Bansal et al., 2014; Ling et al., 2015a), learning context words with different weights (Ling et al., 2015b), etc. They are advantageous of learning word embeddings in an end-to-end unsupervised manner without requiring additional resources. Yet, they are also restricted in their implementation such as that they normally require a larger hidden layer or additional weights, which demand higher computation burden and could result in gradient explosion when embedding dimensions are enlarged. Another issue is that when considering word orders, they may suffer from data sparsity problem since *n*-gram coverage is much less than word, especially in the cold start scenario for a new domain where training data is limited.

To address the aforementioned issues, in this paper, we propose a simple, but effective adaptation of the SG model, namely, directional skip-gram (DSG), with consideration of not only the word co-occurrence patterns, but also their relative positions modeled by a special "direction" vector, which indicates whether the word to be predicted is on the left or right side of the given word. Although similarly motivated as the structured skip-gram (SSG) model (Ling et al., 2015a), DSG produces word embeddings of higher quality with lower space and time complexities. Empirical study shows that DSG can be trained efficiently (as fast as SG, while much faster than SSG). To test the effectiveness of the embeddings produced by DSG, we conduct experiments on semantic (word similarity evaluation) and syntactic (part-of-speech tagging) tasks. The results confirm the superiority of DSG to other models.

Proceedings of NAACL-HLT 2018, pages 175–180
New Orleans, Louisiana, June 1 - 6, 2018. ©2018 Association for Computational Linguistics

2 Approach

2.1 Skip-Gram Model

The SG model (Mikolov et al., 2013b) is a popular choice to learn word embeddings by leveraging the relations between a word and its neighboring words. In detail, the SG model is to predict the context for each given word w_t, and maximizes

$$\mathcal{L}_{SG} = \frac{1}{|V|}\sum_{t=1}^{|V|}\sum_{0<|i|\leq c}\log f(w_{t+i}, w_t) \qquad (1)$$

on a given corpus with vocabulary V, where w_{t+i} denotes the context words in a window w_{t-c}^{t+c}, with c denoting the window size. Herein $f(w_{t+i}, w_t) = p(w_{t+i} \mid w_t)$, and the probability to predict context word is estimated by

$$p(w_{t+i} \mid w_t) = \frac{exp({\upsilon'_{w_{t+i}}}^{\top}\upsilon_{w_t})}{\sum_{w_{t+i}\in V} exp({\upsilon'_{w_{t+i}}}^{\top}\upsilon_{w_t})} \qquad (2)$$

where υ_{w_t} is the embedding for w_t, and υ and υ' refer to input and output embeddings, respectively. The training processing of SG model is thus to maximize $\mathcal{L}_{SG}$ over a corpus iteratively. For a large vocabulary, `word2vec` uses hierarchical softmax or negative sampling (Mikolov et al., 2013b) to address the computational complexity that requires $|V| \times d$ matrix multiplications.

2.2 Structured Skip-Gram Model

The SSG model (Ling et al., 2015a) is an adaptation of SG model with consideration of words' order. The overall likelihood of SSG model shares the same form of SG model as Equation 1, however, with an adapted $f(w_{t+i}, w_t)$ where the probability of predicting w_{t+i} considers not only the word-word relations but also its relative position to w_t. In practice, each word in w_{t-c}^{t+c} is not predicted by a single predictor operating on the output embeddings $\upsilon'_{w_{t+i}}$. Instead, w_{t+i} is predicted by $2c$ predictors according to where it appears in w_t's context. As a result, every word in SSG should have $2c$ output embeddings for the $2c$ relative positions. The probability of predicting w_{t+i} in SSG is thus formulated as

$$p(w_{t+i} \mid w_t) = \frac{exp(\sum_{r=-c}^{c} {\upsilon'_{r,w_{t+i}}}^{\top}\upsilon_{w_t})}{\sum_{w_{t+i}\in V} exp(\sum_{r=-c}^{c} {\upsilon'_{r,w_{t+i}}}^{\top}\upsilon_{w_t})} \qquad (3)$$

where $\upsilon_{r,w_{t+i}}$ defines the positional output embeddings for w_{t+i} at position r with respect to w_t. The embedding of w_t is thus updated with word order information implicitly recorded in $\upsilon_{r,w_{t+i}}$.

Model	Parameters	Operations
SG	$2\|V\|d$	$2c\mathcal{C}(n+1)o$
SSG	$(2c+1)\|V\|d$	$4c^2\mathcal{C}(n+1)o$
SSSG	$3\|V\|d$	$4c\mathcal{C}(n+1)o$
DSG	$3\|V\|d$	$2c\mathcal{C}(n+2)o$

Table 1: Complexities of different SG models. The column of "Parameters" and "Operations" reports space and time complexity, respectively. d: embedding dimension. $\mathcal{C}$: corpus size. o: unit operation of predicting and updating one word's embedding. n: the number of negative samples.

2.3 Directional Skip-Gram Model

The intuition behind this model is that word sequence is an important factor affecting the generation of our languages; a word should be biased associated with other words on its left or right side. For instance, "*merry*" and "*eve*" both co-occur frequently with "*Christmas*" in "*merry Christmas*" and "*Christmas eve*", respectively. Given the context word "Christmas", it is useful to identify the word to be predicted is on the left or right for learning the embeddings of "*merry*" and "*eve*".[1] Motivated by this, we propose a softmax function

$$g(w_{t+i}, w_t) = \frac{exp({\delta_{w_{t+i}}}^{\top}\upsilon_{w_t})}{\sum_{w_{t+i}\in V} exp({\delta_{w_{t+i}}}^{\top}\upsilon_{w_t})} \qquad (4)$$

to measure how a context word w_{t+i} is associated with w_t in its left or right context, by introducing a new vector δ for each w_{t+i} to present its relative direction to w_t. The function g shares an updating paradigm similar to negative sampling:

$$\begin{aligned} \upsilon_{w_t}^{(new)} &= \upsilon_{w_t}^{(old)} - \gamma(\sigma(\upsilon_{w_t}^{\top}\delta_{w_{t+i}}) - \mathcal{D})\delta_{w_{t+i}} \\ \delta_{w_{t+i}}^{(new)} &= \delta_{w_{t+i}}^{(old)} - \gamma(\sigma(\upsilon_{w_t}^{\top}\delta_{w_{t+i}}) - \mathcal{D})\upsilon_{w_t} \end{aligned}$$

where σ denotes the sigmoid function and γ the discounting learning rate. Particularly, $\mathcal{D}$ is the target label specifying the relative direction of w_{t+i} given w_t, defined as

$$\mathcal{D} = \begin{cases} 1 & i < 0 \\ 0 & i > 0 \end{cases}$$

according on the relative position of w_{t+i} respect to w_t. The final model is defined as Equation 1 with $f(w_{t+i}, w_t) = p(w_{t+i} \mid w_t) + g(w_{t+i}, w_t)$.

[1] Although SSG can also model this case because "merry" and "eve" are normally associated with "Christmas" at fixed positions, the intention of this example is to illustrate that word sequence can be effectively modeled by distinguishing left and right context.

Dimension	200
Window size	5
Frequency cut-off	5
Negative samples	5
Starting learning rate	0.025
Iteration	5

Table 2: Model settings for training embeddings.

2.4 Complexity Analysis

To qualitatively analyze the efficiency of our proposed model, we draw Table 1, which compares the complexity of the aforementioned SG models. The *Parameters* column reports parameter size, which refers to the space complexity. The *Operations* column reports the number of operations in computation, referring to the time complexity. Note that the above complexity analysis is based on negative sampling. If using hierarchical softmax, one can replace $n + 1$ into h, which represents the average depth of the hierarchical tree.

Compared to SG model, the SSG model demands obviously higher complexity in terms of both space and time when context gets larger, while every word in the DSG model only requires one extra operation in addition to the original SG model. Thus, if one enlarges the context, the DSG model could have similar speed of SG model.

To fairly compare the efficiency of our model and SSG, we additionally propose a simplified SSG (SSSG) model that only models left and right context for a given word. Instead of having $2c$ output embeddings in SSG, each word in SSSG has only two output embeddings representing left and right context. This is an approximation of our model within the SSG framework. On the output side, SSSG has two “word” vectors respectively for left and right context, while DSG has one “word” vector and one “direction” vector. As a result, the direction vector of DSG can be used to explicitly predict whether the context is on the left or right in word prediction, while SSSG doesn’t.

3 Experiments

We use intrinsic and extrinsic evaluations to evaluate the effectiveness of different embeddings. To test and verify our analysis in §2.4, the efficiencies of aforementioned SG models are investigated based on their training speed. The setups for all experiments are illustrated as follows.

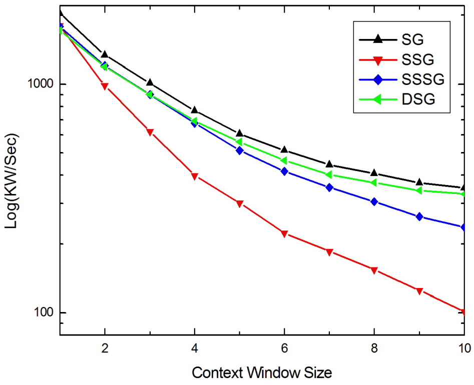

Figure 1: Comparisons of training speed in logarithm against different context window size. KW/Sec refers to thousand words per second.

Dataset. The embeddings were trained on the latest dump of Wikipedia articles[2], which contains approximately 2 billion word tokens.

Comparison. Since the focus of this paper is to enhance the SG model, we mainly consider the SG model (Mikolov et al., 2013b), SSG model (Ling et al., 2015a) and its simplified version SSSG model, as baselines for comparison.

Settings. Different models share the same hyperparameters in training word embeddings, which are presented in Table 2.

3.1 Training Speed

Figure 1 illustrates the training speed of different SG models, i.e., SG, SSG, SSSG, and DSG, given various size of context window.[3] Compared to the original SG model, SSG model shows a relatively large drop of speed when enlarging the context window, while there is much less drop observed for the DSG model. Overall, the curves of four models roughly comply with the qualitative analysis in Table 1. When starting with only one context word, the SSG, SSSG, and DSG model share similar training speed since their time complexities are not affected by the limited context window size under this circumstance. When enlarging the context window, the speed gap between the SSG and SG model is getting larger while the gap between DSG and SG becomes smaller.[4]

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

[3] The numbers on the curves are obtained when running on an Intel Xeon CPU E5-2680 v4 with 12 threads.

[4] Note that the derivations in Table 1 represents the upper bound of the complexities, where every two words co-occur in a context window, which hardly happens in real scenarios. As a result, the observed gaps are slightly smaller from what are presented in Table 1.

	MEN-3k	SimLex-999	WS-353
CBOW	70.96	34.32	69.25
CWin	**74.28**	36.06	72.21
SG	71.90	34.35	70.11
SSG	71.26	31.80	69.46
SSSG	72.07	33.62	70.90
DSG	73.76	**36.10**	**72.60**

Table 3: Word similarity results ($\rho \times 100$) from embeddings trained on the large corpus.

	MEN-3k	SimLex-999	WS-353
CBOW	58.23	26.67	64.40
CWin	59.68	25.19	62.82
SG	60.19	27.14	65.23
SSG	55.42	24.00	61.95
SSSG	62.70	26.55	66.10
DSG	**63.18**	**27.51**	**66.71**

Table 4: Word similarity results ($\rho \times 100$) from embeddings trained on the small corpus.

3.2 Word Similarity Evaluation

As a conventional intrinsic evaluation, word similarity test is performed on the MEN-3k (Bruni et al., 2012), SimLex-999 (Hill et al., 2015) and WordSim-353 (Finkelstein et al., 2002) datasets for quantitative comparisons among different embeddings. The Spearman's rank correlation (ρ) (Zar, 1998) is adopted to measure how close the similarity scores to human judgments on the three datasets. Besides SG, SSG and SSSG, we also include CBOW and CWin[5] as reference baselines in this word similarity evaluation.

Table 3 reports word similarity results when the embeddings are trained on the entire Wiki corpus. Besides, we created a small corpus by sampling 0.1% Wiki data to simulate the cold-start scenario that limited data is used to train word embeddings. The word similarity performance of all models on this small corpus is reported in Table 4. Overall, the results of all models are worse when trained on the small dataset, especially the models taking structure information of context into account, such as CWin and SSG. The reason may be largely due to that modeling order dependence is sensitive to data sparsity, hence CWin model fails to generate meaningful representations for low-frequency words, which are prevalent on small corpus. This observation indicates that data sparsity problem is critical in learning word embeddings. Nevertheless, DSG yields robust results on different scale of training data, which suggests that our model provides an effective solution to learn embeddings with exploiting the structure in context, while not severely suffered from the data sparsity problem. Particularly among all SG models, DSG produces the best performance when trained on either the large or the small corpus. This fact further proves the effectiveness of distinguishing left and right context for SG embeddings.

3.3 Part-of-Speech Tagging

Besides the intrinsic evaluation to test the embeddings semantically, we also evaluate different embeddings syntactically with an extrinsic evaluation: part-of-speech (POS) tagging. Following Ling et al. (2015a), this task is performed in both news and social media data. For news data, we use Wall Street Journal (WSJ) proportion from the Penn Treebank (Marcus et al., 1993) and follow the standard split of 38,219/5,527/5,462 sentences for training, development, and test, respectively. The social media data is based on ARK dataset (Gimpel et al., 2011), which contains manual POS annotations on English tweets. The standard split of ARK contains 1,000/327/500 tweets as training/development/test set, respectively.

POS prediction is conducted by a bidirectional LSTM-CRF (Huang et al., 2015; Lample et al., 2016) taking the produced embeddings as input. LSTM state size is setting to 200. For WSJ, we use the aforementioned embeddings trained from the Wiki corpus. For ARK, we prepare a Twitter corpus (TWT) to build embeddings. This data contains 100 million tweets collected through Twitter streaming API[6], followed by preprocessing using the toolkit described in Owoputi et al. (2013). The TWT embeddings are trained under the same procedure as the Wiki embeddings. Similar to word similarity task, we use CBOW, SG, CWin, SSG and SSSG as baselines in this task.

Results are reported in Table 5. We observe that the DSG embeddings can best indicate POS tags in comparison. It suggests that by exploring word context in left and right directions, DSG model can effectively capture syntactic information, which is

[5] Continuous window model, the counterpart of SSG proposed in Ling et al. (2015a).

[6] https://developer.twitter.com/en/docs/tweets/filter-realtime/overview

	WSJ		ARK	
	Dev	Test	Dev	Test
CBOW	96.86	97.01	89.36	88.36
CWin	96.98	97.25	90.03	89.94
SG	96.95	97.12	89.26	88.77
SSG	97.08	97.31	90.05	90.15
SSSG	97.01	97.19	89.83	89.78
DSG	**97.16**	**97.37**	**90.12**	**90.43**

Table 5: POS tagging results (accuracy) on WSJ and ARK datasets.

useful in predicting POS tags. Although embeddings trained on TWT could be affected by the noisiness and informal nature of tweets, POS taggers with DSG embeddings achieve the best accuracy on ARK data. This observation indicates that, when learning word embeddings with context structures on noisy data, DSG has its superiority to other models such as SSG and SSSG.

4 Conclusions

This paper presents DSG, a simple yet effective extension to the SG model for learning word embeddings. Given an input word, our model jointly predicts its context words as well as their direction to the given word. It is analyzed and experimented that our model can be trained as fast as the original SG model. Experiments on word similarity evaluation and POS tagging demonstrate that DSG produces better semantic and syntactic representations when it is compared with competitive baselines. More importantly, it is also proved that DSG can effectively predict word similarities when trained on small dataset and is therefore less sensitive to data sparsity than existing methods.

Acknowledgements

We thank the three anonymous reviewers for their insightful comments on this work.

References

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring Continuous Word Representations for Dependency Parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Baltimore, Maryland, pages 809–815.

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

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.

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.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA, pages 42–47.

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

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv preprint* abs/1508.01991.

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*. Association for Computational Linguistics, Lisbon, Portugal, pages 2044–2048.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition.

In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, pages 260–270.

Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015a. Two/Too Simple Adaptations of Word2Vec for Syntax Problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 1299–1304.

Wang Ling, Yulia Tsvetkov, Silvio Amir, Ramon Fermandez, Chris Dyer, Alan W Black, Isabel Trancoso, and Chu-Cheng Lin. 2015b. Not All Contexts Are Created Equal: Better Word Representations with Variable Attention. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 1367–1372.

Mitch Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* 19(2):313–330.

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, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. 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. 2013c. 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.

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.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia, pages 380–390.

Yan Song and Chia-Jung Lee. 2017. Learning user embeddings from emails. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Valencia, Spain, pages 733–738.

Yan Song, Chia-Jung Lee, and Fei Xia. 2017. Learning word representations with regularization from prior knowledge. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Vancouver, Canada, pages 143–152.

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 pre-print* abs/1502.05698.

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.

Jerrold H Zar. 1998. Spearman rank correlation. *Encyclopedia of Biostatistics* .