Tiny Word Embeddings Using Globally Informed Reconstruction

Sora Ohashi[†], **Mao Isogawa**[†], **Tomoyuki Kajiwara**[‡], **Yuki Arase**[†]
[†] Graduate School of Information Science and Technology, Osaka University
[‡] Institute for Datability Science, Osaka University
[†] {ohashi.sora, isogawa.mao, arase}@ist.osaka-u.ac.jp
[‡] kajiwara@ids.osaka-u.ac.jp

Abstract

We reduce the model size of pre-trained word embeddings by a factor of 200 while preserving its quality. Previous studies in this direction created a smaller word embedding model by reconstructing pre-trained word representations from those of subwords, which allows to store only a smaller number of subword embeddings in the memory. However, previous studies that train the reconstruction models using only target words cannot reduce the model size extremely while preserving its quality. Inspired by the observation of words with similar meanings having similar embeddings, our reconstruction training learns the global relationships among words, which can be employed in various models for word embedding reconstruction. Experimental results on word similarity benchmarks show that the proposed method improves the performance of the all subword-based reconstruction models.

1 Introduction

Word embeddings form the basis for many natural language processing (NLP) applications, e.g., text classification (Shen et al., 2018) and machine translation (Qi et al., 2018). However, widely used pre-trained word embeddings such as fastText (Bojanowski et al., 2017) are considerably large, thereby making it difficult to develop NLP applications in limited memory environments such as mobile devices. For example, fastText[1] (crawl-300d-2M-subword) requires approximately 2 GB of memory.

In previous studies, the model size has been reduced by reconstructing word embeddings from characters (Pinter et al., 2017; Kim et al., 2018) and character N-grams (Zhao et al., 2018; Sasaki et al., 2019). As the number of characters or character N-grams, is significantly smaller than that of words, reconstructing word embeddings with accuracy from these subwords can reduce the model size while preserving the performance of applications.[2] As shown in Figure 1, existing methods reconstruct word embeddings from subword embeddings and mimic the corresponding pre-trained word embeddings. These methods rely only on local information of subwords and pre-trained word embeddings.

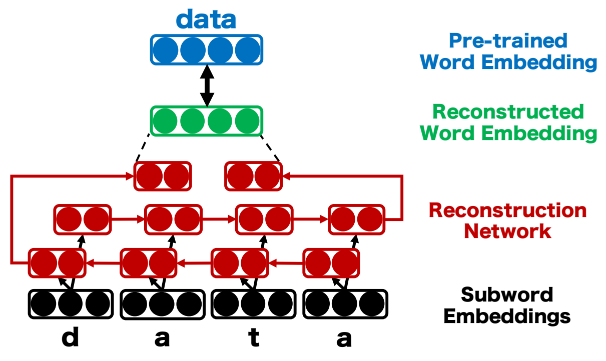
To improve the performance of word embedding reconstruction, we propose a global loss function that uses words other than the target word as clues. Inspired by the observation of words with similar meanings having similar embeddings in pre-trained word embeddings, our reconstruction training learns the similarity among word embeddings. Our method can be easily applied to any method for reconstructing a word embedding from subwords, regardless of the unit of the subword or the network structure.

Experimental results on word similarity tasks (Faruqui and Dyer, 2014) show that our global loss function improves the performance of word embedding reconstruction in all previous methods. When the proposed method was applied to the method based on a self-attention mechanism for reconstructing word embeddings from character N-grams (Sasaki et al., 2019), the model size was reduced to 74 MB (1/30) while preserving 97% of the quality of the original word embeddings. Furthermore, even if the model size was reduced to 12 MB (1/200), 86% of the quality was preserved.

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/.

[1]https://fasttext.cc/docs/en/english-vectors.html

[2]In this study, we refer to characters and character N-grams as subwords.



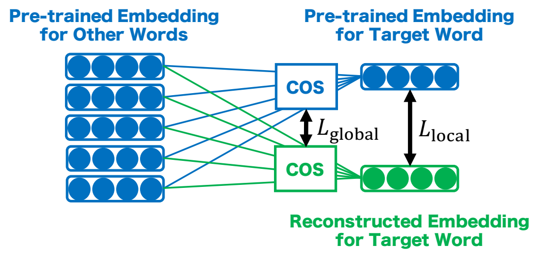

Pre-trained Embedding for Other Words
Pre-trained Embedding for Target Word
cos
cos
L_{global}
L_{local}
Reconstructed Embedding for Target Word

Figure 1: Overview of word embedding reconstruction Figure 2: Overview of our global similarity loss

2 Word Embeddings Reconstruction Based on Global Similarity Loss

We denote the pre-trained embeddings of word $w \in W$ and the randomly initialized embeddings of subword $s \in S$ as $\boldsymbol{e}_w$ and $\boldsymbol{v}_s$, respectively. As in previous studies (Pinter et al., 2017; Kim et al., 2018; Zhao et al., 2018; Sasaki et al., 2019), we reconstruct a word embedding $\boldsymbol{e}_w$ of word w as $\hat{\boldsymbol{e}}_w$, from a set of subwords $\phi(w)$ by minimizing the following loss function.

$$L_{\text{local}} = \frac{1}{d_w}||f(\phi(w)) - \boldsymbol{e}_w||_2^2, \tag{1}$$

where $f(\cdot)$ is the function for reconstructing a word embedding, *i.e.*, a reconstruction network shown in Figure 1, and d_w is the dimension of pre-trained word embeddings. As reconstruction networks, a recurrent neural network (RNN) (Pinter et al., 2017), convolutional neural network (CNN) (Kim et al., 2018), and self-attention mechanism (Sasaki et al., 2019) were used in the previous studies.

To improve word embedding reconstruction, we employ a loss function based on global information in addition to the loss function of Equation (1), which depends only on the local information of the target word, w. As shown in Figure 2, we consider the relationship between the reconstructed and pre-trained embeddings for the target word and the relationship between the reconstructed embedding of the target word and the pre-trained embedding of other words. We define the global loss function using cosine similarity[3] among word embeddings as follows:

$$L_{\text{global}} = \frac{1}{n}\sum_{g \in W}\left(\cos(\hat{\boldsymbol{e}}_w, \boldsymbol{e}_g) - \cos(\boldsymbol{e}_w, \boldsymbol{e}_g)\right)^2. \tag{2}$$

In this study, we sample n words from W in each training batch. To balance the similarity distribution among the selected n words, we first select the top-$n/2$ words that have a high cosine similarity to the target word, and then randomly select the rest from the training batch. Finally, we minimize the loss function that combines Equations (1) and (2), as given below:

$$L = L_{\text{local}} + L_{\text{global}}. \tag{3}$$

3 Experiment

We evaluate the effectiveness of the globally informed reconstruction of word embeddings using word similarity tasks[4] (Faruqui and Dyer, 2014). Our experiment employs the following five datasets: Rubenstein-Goodenough dataset (RG, 65 word-pairs) (Rubenstein and Goodenough, 1965), Miller-Charles dataset (MC, 30 word-pairs) (Miller and Chales, 1991), WordSim-353 (WS, 353 word-pairs) (Finkelstein et al., 2002), MEN test collection (MEN, $3,000$ word-pairs) (Bruni et al., 2012), and Stanford Rare Word Similarity dataset (RW, $2,034$ word-pairs) (Luong et al., 2013). The performance of each method is evaluated using micro averaged Spearman's rank correlation coefficient between the cosine similarities of the word embeddings and gold-standard similarities.

[3]We also tried the mean squared error, but the loss function based on the cosine similarity achieved higher performance.
[4]https://github.com/mfaruqui/eval-word-vectors/

	Character	
	ρ	Size
Character RNN	0.534	14
+ Global Loss	**0.540**	
Character CNN	0.594	25
+ Global Loss	**0.602**	

	Small		Medium		Large	
	ρ	Size	ρ	Size	ρ	Size
Bag of N-gram	0.191	12	0.597	74	0.697	223
+ Global Loss	**0.210**		**0.605**		**0.698**	
N-gram SAM	0.494	12	0.684	74	0.692	223
+ Global Loss	**0.618**		**0.699**		**0.708**	

Table 1: Model size (MB) and micro averaged Spearman's ρ. Original fastText: $\rho = 0.719$ $(2,230$ MB)

3.1 Implementation Details

We employed the 300-dimensional pre-trained fastText[2] (Bojanowski et al., 2017) as the original word embeddings. Each reconstruction network was trained using 100k words based on the descending order of frequency. Words containing hyphens and numbers were excluded, and we only targeted words consisting of 26 lowercase Latin alphabets.

We sampled $n = 10$ words for the global loss calculation. To minimize the loss, we adopted Adam (Kingma and Ba, 2015) ($\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$) with a batch size of 50. The training was stopped after 5 epochs without improvement in the training loss.

3.2 Baseline Methods

We applied the global loss function to the following four methods that reconstruct pre-trained word embeddings from subwords.

Character RNN (Pinter et al., 2017): This method uses characters as subwords. It employs a bidirectional long short-term memory of 512 hidden dimensions based on a 32-dimensional embedding layer.

Character CNN (Kim et al., 2018): This method uses characters as subwords. It employs a CNN with a filter width r of $[1, 2, ..., 7]$, stride width of 3, and number of $\max(200, 50r)$ filters.

Bag of N-gram (Zhao et al., 2018): This method uses character N-grams as subwords. It outputs the element-wise average of a 300-dimensional embedding of each subword.

N-gram SAM (Sasaki et al., 2019): This method uses character N-grams as subwords. It employs a self-attention mechanism that computes weighted averages of subword embeddings.

For the methods based on character N-grams, we experimented with the following three settings depending on the length of the N-gram.[5]

- Small: Only $N = 3$
- Medium: Both $N = 3$ and $N = 4$
- Large: From $N = 3$ to $N = 5$

3.3 Experimental Results

According to Table 1, the proposed method improves the performance of word similarity estimation for all the settings. Especially, in the case of "Small" setting in the N-gram SAM model, the proposed method improves the performance by 25%. These experimental results indicate the effectiveness of the proposed method that considers global relationships among words. For the "Medium" setting of the N-gram SAM model, the model size can be reduced to 74MB, which is approximately $1/30$ (3.3%), while preserving 97% of the performance of the original fastText. Further, for the "Small" setting, the model size can be reduced by a factor of 200 to 12MB, which is approximately 0.5% of that of the original model. Nonetheless, the model preserves 86% of the performance of the original model.

[5]Following Zhao et al. (2018) and Sasaki et al. (2019), the length of the character N-gram is measured with special characters added to the beginning and end of the word.

	$n = 0$	10	20	50
Character RNN	0.534	0.540	**0.562**	0.553
Character CNN	0.594	0.602	0.613	**0.615**
Bag of N-gram	0.191	0.210	0.212	**0.213**
N-gram SAM	0.494	0.618	**0.619**	0.618

Table 2: Spearman's ρ for each sample size n

	Baseline	+ Global Loss
Character RNN	0.619	**0.679**
Character CNN	0.806	**0.817**
Bag of N-gram	0.701	**0.709**
N-gram SAM	0.740	**0.847**

Table 3: Precision@5 of the nearest neighbors

fastText	glasgow	edinburgh	birmingham	nyc	uk	...
N-gram SAM ($n = 0$)	lon	lond	canton	meron	anton	...
N-gram SAM ($n = 10$)	glasgow	chicago	edinburgh	atlanta	brooklyn	...
fastText	influenza	pneumonia	bronchitis	illness	measles	...
N-gram SAM ($n = 0$)	litis	lam	tis	sine	flue	...
N-gram SAM ($n = 10$)	influenza	pneumonia	pneumonias	smallpox	diphtheria	...

Table 4: Nearest Neighbors of the word "london" (upper section) and "flu" (lower section)

3.4 Effect of Sample Size

In Table 1, we sampled $n = 10$ words for the global loss calculation. In this section, we investigate the effect of the number of word samples on the performance of word similarity estimation. Table 2 shows the performance when the sample size was changed to $n = 10$, 20, and 50. Notably, $n = 0$ is a baseline model that does not perform global loss calculation.

According to Table 2, a large sample size for global loss calculation is not always effective. Overall, the $n = 20$ model performs better than the $n = 10$ model, but there is no large improvement beyond $n = 20$. Nevertheless, the proposed method consistently performs better than the $n = 0$ baseline model regardless of the sample size.

3.5 Analysis of Nearest Neighbors

To analyze the reconstructed embeddings, we investigated words having similar meaning using 100k high-frequency words in fastText. Table 3 shows the precision@5 for similar word searches. As the correct similar words, we collected the top 5 words with cosine similarity in the original fastText. In this analysis, the sample size of the global loss calculation is $n = 10$, and the methods based on character N-grams use the "Small" settings.

Table 3 shows that the proposed method always improves the performance of searching for similar words. Table 4 lists similar-word searches for the word "london" and "flu." According to the upper section of the table, the original fastText lined up the big cities of the United Kingdom like "london" such as "glasgow" and "birmingham." However, in the N-gram SAM ($n = 0$) in the "Small" setting, words such as "lon" and "lond" that are similar on the surface but are significantly different in meaning are collected at the top. In the N-gram SAM ($n = 10$), by considering the relationship among words, semantically similar words such as "glasgow" and "edinburgh" succeeded in acquiring embeddings similar to "london." The lower section of the table is an example of "flu." The baseline model failed in reconstruction and lists irrelevant words. However, the proposed method successfully finds words that are semantically similar to "flu," such as "influenza" and "pneumonia."

4 Conclusion

We proposed a loss function that considers global relationships among words for the reconstruction of pre-trained word embeddings from subword embeddings. Experimental results on word similarity benchmarks show that the proposed method improves the performance of all the reconstruction networks. By applying the proposed method, we can compress 2GB of fastText to 12MB while preserving the quality of the original word embeddings. This method will help develop NLP applications in limited memory environments, e.g., mobile devices.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

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*, pages 136–145.

Manaal Faruqui and Chris Dyer. 2014. Community Evaluation and Exchange of Word Vectors at wordvectors.org. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 19–24.

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

Yeachan Kim, Kang-Min Kim, Ji-Min Lee, and SangKeun Lee. 2018. Learning to Generate Word Representations using Subword Information. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2551–2561.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations*.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better Word Representations with Recursive Neural Networks for Morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113.

George A. Miller and Walter G. Chales. 1991. Contextual Correlates of Semantic Similarity. *Language and Cognitive Processes*, 6(1):1–28.

Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 2017. Mimicking Word Embeddings using Subword RNNs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 102–112.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and Why Are Pre-Trained Word Embeddings Useful for Neural Machine Translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 529–535.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10):627–633.

Shota Sasaki, Jun Suzuki, and Kentaro Inui. 2019. Subword-based Compact Reconstruction of Word Embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3498–3508.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 440–450.

Jinman Zhao, Sidharth Mudgal, and Yingyu Liang. 2018. Generalizing Word Embeddings using Bag of Subwords. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 601–606.