A Unified Learning Framework of Skip-Grams and Global Vectors

Jun Suzuki and **Masaaki Nagata**
NTT Communication Science Laboratories, NTT Corporation
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237 Japan
{suzuki.jun, nagata.masaaki}@lab.ntt.co.jp

Abstract

Log-bilinear language models such as SkipGram and GloVe have been proven to capture high quality syntactic and semantic relationships between words in a vector space. We revisit the relationship between SkipGram and GloVe models from a machine learning viewpoint, and show that these two methods are easily merged into a unified form. Then, by using the unified form, we extract the factors of the configurations that they use differently. We also empirically investigate which factor is responsible for the performance difference often observed in widely examined word similarity and analogy tasks.

1 Introduction

Neural-network-inspired word embedding methods such as Skip-Gram (**SkipGram**) have been proven to capture high quality syntactic and semantic relationships between words in a vector space (Mikolov et al., 2013a). A similar embedding method, called 'Global Vector (**GloVe**)', was recently proposed. It has demonstrated significant improvements over SkipGram on the widely used 'Word Analogy' and 'Word Similarity' benchmark datasets (Pennington et al., 2014). Unfortunately, a later deep re-evaluation has revealed that GloVe does not consistently outperform SkipGram (Levy et al., 2015); both methods provided basically the same level of performance, and SkipGram even seems 'more robust (not yielding very poor results)' than GloVe. Moreover, some other papers, *i.e.*, (Shi and Liu, 2014), and some researchers in the community have discussed a relationship, and/or which is superior, SkipGram or GloVe.

From this background, we revisit the relationship between SkipGram and GloVe from a machine learning viewpoint. We show that it is natural to think that these two methods are essentially identical, with the chief difference being their learning configurations.

$\mathcal{V}$	: set of vocabulary (set of words)
$\|\mathcal{V}\|$	: vocabulary size, or number of words in $\mathcal{V}$
i	: index of the input vector, where $i \in \{1, \dots, \|\mathcal{V}\|\}$
j	: index of the output vector, where $j \in \{1, \dots, \|\mathcal{V}\|\}$
$\mathbf{e}_i$	: input vector of the i-th word in $\mathcal{V}$
$\mathbf{o}_j$	: output vector of the j-th word in $\mathcal{V}$ If $i = j$, then $\mathbf{e}_i$ and $\mathbf{o}_j$ are the input and output vectors of the same word in $\mathcal{V}$, respectively.
D	: number of dimensions in input and output vectors
$m_{i,j}$	: (i, j)-factor of matrix M
$s_{i,j}$	: dot product of input and output vectors, $s_{i,j} = \mathbf{e}_i \cdot \mathbf{o}_j$
$\mathcal{D}$	: training data, $\mathcal{D} = \{(i_n, j_n)\}_{n=1}^N$
$\Psi(\cdot)$	: objective function
$\sigma(\cdot)$	: sigmoid function, $\sigma(x) = \frac{1}{1+\exp(-x)}$
$c_{i,j}$	: co-occurrence of the i-th and j-th words in $\mathcal{D}$
$\mathcal{D}'$	: (virtual) negative sampling data
$c'_{i,j}$	: co-occurrence of the i-th and j-th words in $\mathcal{D}'$
k	: hyper-parameter of the negative sampling
$\beta(\cdot)$	: 'weighting factor' of loss function
$\Phi(\cdot)$	: loss function

Table 1: List of notations used in this paper.

The final goal of this paper is to provide a unified learning framework that encompasses the configurations used in SkipGram and GloVe to gain a deeper understanding of the behavior of these embedding methods. We also empirically investigate which learning configuration most clearly elucidates the performance difference often observed in word similarity and analogy tasks.

2 SkipGram and GloVe

Table 1 shows the notations used in this paper.

2.1 Matrix factorization view of SkipGram

SkipGram can be categorized as one of the simplest neural language models (Mnih and Kavukcuoglu, 2013). It generally assigns two distinct D-dimensional vectors to each word in vocabulary $\mathcal{V}$; one is 'input vector', and the other is 'output vector'[1].

[1] These two vectors are generally referred to as 'word (or target) vector' and 'context vector'. We use the terms 'in-

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics
and the 7th International Joint Conference on Natural Language Processing (Short Papers), pages 186–191,
Beijing, China, July 26-31, 2015. ©2015 Association for Computational Linguistics

Roughly speaking, SkipGram models word-to-word co-occurrences, which are extracted within the predefined context window size, by the input and output vectors. Recently, SkipGram has been interpreted as implicitly factorizing the matrix, where the factors are calculated from co-occurrence information (Levy and Goldberg, 2014). Let $m_{i,j}$ be the (i,j)-factor of matrix $\mathbf{M}$ to be 'implicitly' factorized by SkipGram. SkipGram approximates each $m_{i,j}$ by the inner product of the corresponding input and output vectors, that is:

$$m_{i,j} \approx \mathbf{e}_i \cdot \mathbf{o}_j, \tag{1}$$

2.1.1 SkipGram with negative sampling

The primitive training sample for SkipGram is a pair of a target word and its corresponding context word. Thus, we can represent the training data of SkipGram as a list of input and output index pairs, that is, $\mathcal{D} = \{(i_n, j_n)\}_{n=1}^N$. Thus the estimation problem of 'SkipGram with negative sampling (**SGNS**)' is defined as the minimization problem of objective function Ψ:

$$\begin{aligned}\Psi = &- \sum_{(i_n,j_n)\in\mathcal{D}} \log\big(\sigma(\mathbf{e}_{i_n}\cdot\mathbf{o}_{j_n})\big)\\ &- \sum_{(i_n,j_n)\in\mathcal{D}'} \log\big(1-\sigma(\mathbf{e}_{i_n}\cdot\mathbf{o}_{j_n})\big),\end{aligned} \tag{2}$$

where the optimization parameters are $\mathbf{e}_i$ and $\mathbf{o}_j$ for all i and j. Note that we explicitly represent the negative sampling data $\mathcal{D}'$ (Goldberg and Levy, 2014).

Let us assume that, in a preliminary step, we count all co-occurrences in $\mathcal{D}$. Then, the SGNS objective in Eq. 2 can be rewritten as follows by a simple reformulation:

$$\begin{aligned}\Psi = -\sum_i\sum_j \Big(&c_{i,j}\log\big(\sigma(\mathbf{e}_i\cdot\mathbf{o}_j)\big)\\ &+c'_{i,j}\log\big(1-\sigma(\mathbf{e}_i\cdot\mathbf{o}_j)\big)\Big).\end{aligned} \tag{3}$$

Here, let us substitute $\mathbf{e}_i \cdot \mathbf{o}_j$ in Eq. 3 for $s_{i,j}$, and then assume that all $s_{i,j}$ are free parameters. Namely, we can freely select the value of $s_{i,j}$ independent from any other $s_{i',j'}$, where $i \neq i'$ and $j \neq j'$, respectively. The partial derivatives of Ψ with respect to $s_{i,j}$ take the following form:

$$\partial_{s_{i,j}}\Psi = -\Big(c_{i,j}\big(1-\sigma(s_{i,j})\big) - c'_{i,j}\sigma(s_{i,j})\Big). \tag{4}$$

The minimizer can be obtained when $\partial_{s_{i,j}}\Psi = 0$ for all $s_{i,j}$. By using this relation, we can obtain the following closed form solution:

$$s_{i,j} = \log\left(\frac{c_{i,j}}{c'_{i,j}}\right). \tag{5}$$

Overall, SGNS approximates the log of the co-occurrence ratio between 'real' training data $\mathcal{D}$ and 'virtual' negative sampling data $\mathcal{D}'$ by the inner product of the corresponding input and output vectors in terms of minimizing the SGNS objective written in Eq. 2, and Eq. 3 as well. Therefore, we can obtain the following relation for SGNS:

$$m_{i,j} = \log\left(\frac{c_{i,j}}{c'_{i,j}}\right) \approx \mathbf{e}_i \cdot \mathbf{o}_j. \tag{6}$$

Note that the expectation of $c'_{i,j}$ is $\frac{kc_ic_j}{|\mathcal{D}|}$ if the negative sampling is assumed to follow unigram probability $\frac{c_j}{|\mathcal{D}|}$, and the negative sampling data is k-times larger than the training data $\mathcal{D}$, where $c_i = \sum_j c_{i,j}$ and $c_j = \sum_i c_{i,j}$[2]. The above matches 'shifted PMI' as described in (Levy and Goldberg, 2014) when we substitute $c'_{i,j}$ for $\frac{kc_ic_j}{|\mathcal{D}|}$ in Eq. 6,

In addition, the word2vec implementation uses a smoothing factor α to reduce the selection of high-occurrence-frequency words during the negative sampling. The expectation of $c'_{i,j}$ can then be written as: $kc_i\frac{(c_j)^\alpha}{\sum_{j'}(c_{j'})^\alpha}$. We refer to $\log\left(c_{i,j}\frac{\sum_{j'}(c_{j'})^\alpha}{kc_i(c_j)^\alpha}\right)$ as 'α-parameterized shifted PMI (SPMI$_{k,\alpha}$)'.

2.2 Matrix factorization view of GloVe

The GloVe objective is defined in the following form (Pennington et al., 2014):

$$\Psi = \sum_i\sum_j \beta(c_{i,j})\Big(\mathbf{e}_i\cdot\mathbf{o}_j - \log(c_{i,j})\Big)^2, \tag{7}$$

where $\beta(\cdot)$ represent a 'weighting function'. In particular, $\beta(\cdot)$ satisfies the relations $0 \leq \beta(x) < \infty$, and $\beta(x) = 0$ if $x = 0$. For example, the following weighting function has been introduced in (Pennington et al., 2014):

$$\beta(x) = \min\Big(1, \big(x/x_{\max}\big)^\gamma\Big). \tag{8}$$

This is worth noting here that the original GloVe introduces two bias terms, b_i and b_j, and defines

put' and 'output' to reduce the ambiguity since 'word' and 'context' are exchangeable by the definition of model (*i.e.*, SkipGram or CBoW).

[2] Every input of the i-th word samples k words. Therefore, the negative sampling number is kc_i. Finally, the expectation can be obtained by multiplying count kc_i by probability $\frac{c_j}{|\mathcal{D}|}$.

configuration	SGNS	GloVe
training unit	sample-wise	co-occurrence
loss function	logistic (Eq. 11)	squared (Eq. 12)
neg. sampling	explicit	no sampling
weight. func. $\beta(\cdot)$	fixed to 1	Eq. 8
fitting function	SPMI$_{k,\alpha}$	$\log(c_{i,j})$
bias	none	b_i and b_j

Table 2: Comparison of the different configurations used in SGNS and GloVe.

$\mathbf{e}_i \cdot \mathbf{o}_j + b_i + b_j$ instead of just $\mathbf{e}_i \cdot \mathbf{o}_j$ in Eq. 7. For simplicity and ease of discussion, we do not explicitly introduce bias terms in this paper. This is because, without loss of generality, we can embed the effect of the bias terms in the input and output vectors by introducing two additional dimensions for all $\mathbf{e}_i$ and $\mathbf{o}_j$, and fixing parameters $e_{i,D+1} = 1$ and $o_{j,D+2} = 1$.

According to Eq. 7, GloVe can also be viewed as a matrix factorization method. Different from SGNS, GloVe approximates the log of co-occurrences:

$$m_{i,j} = \log(c_{i,j}) \approx \mathbf{e}_i \cdot \mathbf{o}_j, \tag{9}$$

3 Unified Form of SkipGram and GloVe

An examination of the differences between Eqs. 6 and 9 finds that Eq. 6 matches Eq. 9 if $c'_{i,j} = 1$. Recall that $c'_{i,j}$ is the number of co-occurrences of (i, j) in negative sampling data $\mathcal{D}'$. Therefore, what GloVe approximates is SGNS when the negative sampling data $\mathcal{D}'$ is constructed as 1 for all co-occurrences. From the viewpoint of matrix factorization, GloVe can be seen as a special case of SGNS, in that it utilizes a sort of uniform negative sampling method.

Our assessment of the original GloVe paper suggests that the name "Global Vector" mainly stands for the architecture of the two stage learning framework. Namely, it first counts all the co-occurrences in $\mathcal{D}$, and then, it leverages the gathered co-occurrence information for estimating (possibly better) parameters. In contrast, the name "SkipGram" stands mainly for the model type; how it counts the co-occurrences in $\mathcal{D}$. The key points of these two methods seems different and do not conflict. Therefore, it is not surprising to treat these two similar methods as one method; for example, SkipGram model with two-stage global vector learning. The following objective function is a generalized form that subsumes Eqs. 3 and 7:

$$\Psi = \sum_i \sum_j \beta(c_{i,j}) \Phi(\mathbf{e}_i, \mathbf{o}_j, c_{i,j}, c'_{i,j}). \tag{10}$$

hyper-parameter	selected value	
	word2vec	glove
context window (W)	10	
sub (Levy et al., 2015)	dirty, $t = 10^{-5}$	–
del (Levy et al., 2015)	use 400,000 most frequent words	
cds (Levy et al., 2015)	$\alpha = 3/4$	–
w+c (Levy et al., 2015)	$\mathbf{e} + \mathbf{o}$	
weight. func. ($\gamma, x_{\max}$)	–	3/4, 100
initial learning rate (η)	0.025	0.05
# of neg. sampling (k)	5	–
# of iterations (T)	5	20
# of threads	56	
# of dimensions (D)	300	

Table 3: Hyper-parameters in our experiments.

In particular, the original SGNS uses $\beta(c_{i,j}) = 1$ for all (i, j), and logistic loss function:

$$\begin{aligned}\Phi(\mathbf{e}_i, \mathbf{o}_j, c_{i,j}, c'_{i,j}) = {} & c_{i,j} \log\big(\sigma(\mathbf{e}_i \cdot \mathbf{o}_j)\big) \\ & + c'_{i,j} \log\big(1 - \sigma(\mathbf{e}_i \cdot \mathbf{o}_j)\big).\end{aligned} \tag{11}$$

In contrast, GloVe uses a least squared loss function:

$$\Phi(\mathbf{e}_i, \mathbf{o}_j, c_{i,j}, c'_{i,j}) = \Big(\mathbf{e}_i \cdot \mathbf{o}_j - \log\Big(\frac{c_{i,j}}{c'_{i,j}}\Big)\Big)^2. \tag{12}$$

Table 2 lists the factors of each configuration used differently in SGNS and GloVe.

Note that this unified form also includes SkipGram with noise contrastive estimation (SGNCE) (Mnih and Kavukcuoglu, 2013), which approximates $m_{i,j} = \log(\frac{c_{i,j}}{kc_j})$ in matrix factorization view. This paper omits a detailed discussion of SGNCE for space restrictions.

4 Experiments

Following the series of neural word embedding papers, our training data is taken from a Wikipedia dump (Aug. 2014). We tokenized and lowercased the data yielding about 1.8B tokens.

For the hyper-parameter selection, we mostly followed the suggestion made in (Levy et al., 2015). Table 3 summarizes the default values of hyper-parameters used consistently in all our experiments unless otherwise noted.

4.1 Benchmark datasets for evaluation

We prepared eight word similarity benchmark datasets (**WSimilarity**), namely, R&G (Rubenstein and Goodenough, 1965), M&C (Miller and Charles, 1991), WSimS (Agirre et al., 2009), WSimR (Agirre et al., 2009), MEM (Bruni et al., 2014), MTurk (Radinsky et al., 2011), SCWS (Huang et al., 2012), and RARE (Luong

method	time	WSimilarity	WAnalogy
SGNS (original)	8856	**65.4** (65.2, 65.7)	63.0 (62.2, 63.8)
GloVe (original)	8243	57.6 (57.5, 57.9)	64.8 (64.6, 65.0)
w/o bias terms	8027	57.6 (57.5, 57.7)	64.8 (64.5, 65.0)
fitting=SPMI$_{k,\alpha}$	8332	57.5 (57.2, 57.8)	**65.0** (64.8. 65.1)

Table 4: Results: the micro averages of Spearman's rho (WSimilarity) and accuracy (WAnalogy) for all benchmark datasets.

et al., 2013). Moreover, we also prepared three analogy benchmark datasets (**WAnalogy**), that is, GSEM (Mikolov et al., 2013a), GSYN (Mikolov et al., 2013a), and MSYN (Mikolov et al., 2013b).

4.2 SGNS and GloVe Results

Table 4 shows the training time and performance results gained from our benchmark data. The column 'time' indicates average elapsed time (second) for model learning. All the results are the **average performance of ten runs**. This is because the comparison methods have some randomized factors, such as initial value (since they are non-convex optimization problems) and (probabilistic) sampling method, which significantly impact the results.

At first, we compared the original SGNS as implemented in the `word2vec` package[3] and the original GloVe as implemented in the `glove` package[4]. These results are shown in the first and second rows in Table 4. In our experiments, SGNS significantly outperformed GloVe in WSimilarity while GloVe significantly outperformed SGNS in WAnalogy. As we explained, these two methods can be easily merged into a unified form. Thus, there must be some differences in their configurations that yields such a large difference in the results. Next, we tried to determine the clues as the differences.

4.3 Impact of incorporating bias terms

The third row (w/o bias terms) in Table 4 shows the results of the configuration without using the bias terms in the `glove` package. A comparison with the results of the second row, finds no meaningful benefit to using the bias terms. In contrast, obviously, the elapsed time for model learning is consistently shorter since we can discard the bias term update.

(a) WSimilarity

method	W=2	3	5	10	20
SGNS (original)	64.9	65.1	**65.4**	**65.4**	64.9
GloVe (original)	53.6	55.7	57.0	57.6	**57.8**
w/o harmonic func.	54.6	56.9	57.8	**58.2**	57.9

(b) WAnalogy

method	W=2	3	5	10	20
SGNS (original)	62.8	63.5	**63.9**	63.0	61.3
GloVe (original)	51.7	58.4	62.3	64.8	**66.1**
w/o harmonic func.	52.6	58.0	60.5	**61.6**	60.7

Table 5: Impact of the context window size, and harmonic function.

	W=2	3	5	10	20
(1) $0 < c_{i,j} < 1$	104M	213M	377M	649M	914M
(2) $1 \leq c_{i,j}$	167M	184M	207M	234M	251M
non-zero $c_{i,j}$	271M	398M	584M	883M	1165M
ratio of (1)	38.5%	53.6%	64.5%	73.5%	78.4%

Table 6: The ratio of entries less than one in co-occurrence matrix.

4.4 Impact of fitting function

The fourth row (fitting=SPMI$_{k,\alpha}$) in Table 4 shows the performance when we substituted the fitting function of GloVe, namely, $\log(c_{i,j})$, for SPMI$_{k=5,\alpha=3/4}$ used in SGNS. Clearly, the performance becomes nearly identical to the original GloVe. Accordingly, the selection of fitting function has only a small impact.

4.5 Impact of context window size and harmonic function

Table 5 shows the impact of context window size W. The results of SGNS seem more stable against W than those of GloVe.

Additionally, we investigated the impact of the 'harmonic function' used in GloVe. The 'harmonic function' uses the inverse of context distance, *i.e.*, $1/a$ if the context word is a-word away from the target word, instead of just count 1 regardless of the distance when calculating the co-occurrences. Clearly, GloVe without using the harmonic function shown in the third row of Table 5 yielded significantly degraded performance on WAnalogy, and slight improvement on WSimilarity. This fact may imply that the higher WAnalogy performance of GloVe was derived by the effect of this configuration.

4.6 Link between harmonic function and negative sampling

This section further discusses a benefit of harmonic function.

Recall that GloVe does not explicitly consider 'negative samples'. It fixes $c'_{i,j} = 1$ for all (i, j) as shown in Eq. 7. However, the co-occurrence

[3] https://code.google.com/p/word2vec/

[4] http://nlp.stanford.edu/projects/glove/

count given by using the harmonic function can take values less than 1, *i.e.,* $c_{i,j} = 2/3$, if the i-th word and the j-th word co-occurred twice with distance 3. As a result, the value of the fitting function of GloVe becomes $\log(2/3)$. Interestingly, this is essentially equivalent to co-occur 3 times in the negative sampling data and 2 times in the real data since the fitting function of the unified form shown in Eq. 12 is $\log(c_{i,j}/c'_{i,j}) = \log(2/3)$ when $c_{i,j} = 2$ and $c'_{i,j} = 3$. It is not surprising that rare co-occurrence words that occur only in long range contexts may have almost no correlation between them. Thus treating them as negative samples will not create a problem in most cases. Therefore, the harmonic function seems to 'unexpectedly' mimic a kind of a negative sampling method; it is interpreted as 'implicitly' generating negative data.

Table 6 shows the ratio of the entries $c_{i,j}$ whose value is less than one in matrix $\mathbf{M}$. Remember that vocabulary size was 400,000 in our experiments. Thus, we had a total of 400K×400K=160B elements in $\mathbf{M}$, and most were 0. Here, we consider only non-zero entries. It is clear that longer context window sizes generated many more entries categorized in $0 < c_{i,j} < 1$ by the harmonic function. One important observation is that the ratio of $0 < c_{i,j} < 1$ is gradually increasing, which offers a similar effect to increasing the number of negative samples. This can be a reason why GloVe demonstrated consistent improvements in WAnalogy performance as context window increased since larger negative sampling size often improves performance (Levy et al., 2015). Note also that the number of $0 < c_{i,j} < 1$ always becomes 0 in the configuration without the harmonic function. This is equivalent to using uniform negative sampling $c'_{i,j} = 1$ as described in Sec. 3. This fact also indicates the importance of the negative sampling method.

4.7 Impact of weighting function

Table 7 shows the impact of weighting function used in GloVe, namely, Eq 8. Note that '$\beta(\cdot)$=1' column shows the results when we fixed 1 for all non-zero entries[5]. This is also clear that the weighting function Eq 8 with appropriate parameters significantly improved the performance of both WSimilarity and WAnalogy tasks. However unfortunately, the best parameter values for WSimilarity and WAnalogy tasks looks different.

(a) WSimilarity

hyper param.	$\beta(\cdot)$=1	$x_{\max}$ = 1	10	100	10000
$\gamma = 0.75$	59.4	60.1	**60.9**	57.7	49.5
w/o harmonic func.	58.2	58.0	**60.7**	58.2	56.0
$\gamma = 1.0$	(59.4)	**60.1**	59.4	55.9	36.1
w/o harmonic func.	(58.2)	58.3	**60.7**	57.7	46.7

(b) WAnalogy

hyper param.	$\beta(\cdot)$=1	$x_{\max}$ = 1	10	100	10000
$\gamma = 0.75$	55.7	61.1	64.3	**64.8**	28.4
w/o harmonic func.	53.4	52.6	60.3	**61.6**	42.5
$\gamma = 1.0$	(55.7)	61.0	**63.8**	59.1	7.5
w/o harmonic func.	(53.4)	54.1	**60.8**	60.1	20.3

Table 7: Impact of the weighting function.

We emphasize that harmonic function discussed in the previous sub-section was still a necessary condition to obtain the best performance, and better performance in the case of '$\beta(\cdot)$=1' as well.

5 Conclusion

This paper reconsidered the relationship between SkipGram and GloVe models in machine learning viewpoint. We showed that SGNS and GloVe can be easily merged into a unified form. We also extracted the factors of the configurations that are used differently. We empirically investigated which learning configuration is responsible for the performance difference often observed in widely examined word similarity and analogy tasks. Finally, we found that at least two configurations, namely, the weighting function and harmonic function, had significant impacts on the performance. Additionally, we revealed a relationship between harmonic function and negative sampling. We hope that our theoretical and empirical analyses will offer a deeper understanding of these neural word embedding methods[6].

Acknowledgment

We thank three anonymous reviewers for their helpful comments.

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 19–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

[5] This is equivalent to set 0 to `-x-max` option in `glove` implementation.

[6] The modified codes for our experiments will be available in author's homepage

Elia Bruni, Nam Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *J. Artif. Int. Res.*, 49(1):1–47, January.

Yoav Goldberg and Omer Levy. 2014. word2vec Explained: Deriving Mikolov et al.'s Negative-sampling Word-embedding Method. *CoRR*, abs/1402.3722.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pages 873–882. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics*, 3.

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, Sofia, Bulgaria, August. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. 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*, pages 746–751, Atlanta, Georgia, June. Association for Computational Linguistics.

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

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2265–2273. Curran Associates, Inc.

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)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 337–346, New York, NY, USA. ACM.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633, October.

Tianze Shi and Zhiyuan Liu. 2014. Linking GloVe with word2vec. *CoRR*, abs/1411.5595.