Sedov, Denis; Yang, Zhirong

Word embedding based on low-rank doubly stochastic matrix decomposition

Word Embedding based on Low-Rank Doubly Stochastic Matrix Decomposition

Denis Sedov[1] and Zhirong Yang[1,2]

[1] Department of Compu"ter Science, Aalto University
[2] Department of Computer Science, Norwegian University of Science and Technology
denis.sedov@aalto.fi,zhirong.yang@ntnu.no

Abstract. Word embedding, which encodes words into vectors, is an important starting point in natural language processing and commonly used in many text-based machine learning tasks. However, in most current word embedding approaches, the similarity in embedding space is not optimized in the learning. In this paper we propose a novel neighbor embedding method which directly learns an embedding simplex where the similarities between the mapped words are optimal in terms of minimal discrepancy to the input neighborhoods. Our method is built upon two-step random walks between words via topics and thus able to better reveal the topics among the words. Experiment results indicate that our method, compared with another existing word embedding approach, is more favorable for various queries.

Keywords: Nonnegative matrix factorization, Word embedding, Cluster analysis, Doubly stochastic

1 Introduction

In recent years machine learning (ML) that involves text data has found many real-world applications [6, 11, 13]. Each data item in these applications is a sequence of words and other tokens. Originally each word is represented by its ID. However, this is not suitable for machine learning, where most common ML algorithms admit vectors as their input. One-hot encoding is inefficient when the vocabulary is large. Therefore word embedding which finds a low-dimensional vectorial representation of words is a fundamental starting point.

A good word embedding method should respect the relations among the words. It is commonly to learn an embedding vector space where the neighborhoods of the words are approximately preserved. Two typical approaches include *Word2Vec* [8] which maximizes the likelihood of each word given their neighbors (or in the reversed way) and *GloVe* which minimizes a weighted squared loss between the input and output pairwise relations. Some variants of Word2Vec and GloVe have been proposed subsequently [12, 5, 3].

However, embeddings learned by the above approaches may not provide optimal similarities between the words. After the word vectors are obtained, their pairwise similarities require external measures such as cosine similarity, which

can be suboptimal because the learning objective involves non-normalized word vectors. Moreover, the negative sampling trick in Word2Vec provides only an approximating surrogate. Theoretically it remains unknown whether the ad hoc choice of negative distribution guarantees that the original CBOW or Skip-Gram objectives are optimized or not.

In this paper we present a new nonnegative matrix factorization (NMF) method and apply it to learn vectorial representation of words. Our method factorizes the doubly stochastically constrained approximating matrix. In this way we directly optimize over the normalized word vectors and provide their optimal similarities in the embedding space in terms of least approximation discrepancy. Unlike Word2Vec, our method does not require extra stochastic approximation tricks or assumptions on negative distributions. We test our method on two popularly used text data sets and compare it with the Word2Vec results. Our results indicate that the proposed method is often more favorable for various k-nearest-neighbor queries.

The remaining of the paper is organized as follows. In Section 2 we review the word embedding problem and two existing embedding methods. Next we present our new NMF method and show how to apply it to learn probabilistic representation of words in Section 3. Our optimization algorithm is presented in Section 4. Experimental setting and results are presented in Section 5. Then in Section 6 we conclude the work and discuss some future directions.

2   Brief Review of Previous Word Embedding Methods

A text corpus can be treated as a sequence of words and some other tokens such as punctuations. Originally each word is represented by their id in the vocabulary. Because many modern machine learning methods admit vectors as input, conventionally the word ids are converted into their one-hot encodings. That is, the i-th word in the N-sized vocabulary is represented by an N-dimensional vector with the i-entry is 1 and the others are zeros. Obviously, such one-hot encoding is inefficient when N is large. A low-dimensional (r-dim with $r \ll N$) vector encoding, called word embedding, is needed for more efficient learning tasks.

Word embeddings should respect the proximity of words in the original sequence. A common requirement is that if two words often appear nearby, their mapped points in the embedding space should be close. On the other hand, if two words seldom co-occur in the same neighborhood, they should be placed distantly in the embedding.

One way to implement the above requirement is to maximize the likelihood of a language model. For example, the Word2Vec Skip-Gram method finds the vectors $\{w_t\}_{t=1}^N$ of the words which maximizes

$$\mathcal{L}(\{w_t\}_{t=1}^N) = \frac{1}{T} \sum_{t=1}^T \sum_{j \in \mathcal{N}(t)} \log P\left(\text{word}_j | \text{word}_t\right) , \tag{1}$$

where $\mathcal{N}(t)$ is the neighborhood of location t and the conditional likelihood is defined as

$$P\left(\text{word}_j|\text{word}_t\right) = \frac{\exp\left(w_j^T w_t\right)}{\sum_{i=1}^{N} \exp\left(w_i^T w_t\right)} \tag{2}$$

Another approach is to approximately preserve the probability that a word appears in the neighborhood of another word. For example, GloVe implements the approximation by minimizing a weighted squared loss [9], assuming log-normal noise in the observed neighboring frequencies.

3   Low-Rank Doubly Stochastic Matrix Decomposition

Although Word2Vec and GloVe are widely used, they do not provide a metric in the embedding space for retrieval. Cosine similarity as a conventional choice in natural language processing is often used to calculate, for example, k-nearest neighbors of a query in the embedding space. However, the cosine similarities between words are not optimized during the embedding learning. Therefore the retrieval based on such an external metric may not respect the original data distribution.

We observe that the mismatch arises mainly because the word vectors are not normalized in the learning objective, but they are normalized in the metric for retrieval. To overcome this problem we propose to use a new learning objective which explicitly involves the normalized word vectors. First, we employ the doubly stochasticity constraint to normalize the similarities in the embedding space, which enforces that each row or column of the output similarity matrix has unitary sum. This means each word in the embedding space has equal total similarity and denoises the imbalanced effect in the input space. Second, we find a low-rank nonnegative matrix which factorizes the doubly stochastic matrix, which significantly reduces the dimensionality of word vectors. Our optimization is based on multiplicative updates which are widely used in nonnegative matrix factorization (see Section 4).

Let W be the word embedding matrix (rows as word vector codes). It has been shown that the above factorization problem can be reformulated as follows (see Theorem 1 in [14]):

$$\underset{W \geq 0}{\text{minimize}} \ \ \mathcal{J}(W) = D(S||\widehat{S}) \tag{3}$$

$$\text{subject to} \ \ \widehat{S}_{ij} = \sum_{k=1}^{r} \frac{W_{ik}W_{jk}}{\sum_v W_{vk}}, \tag{4}$$

$$\sum_{k=1}^{r} W_{ik} = 1, \ i = 1\dots,N. \tag{5}$$

Here $D()$ is an information divergence measuring the discrepancy between between the input and output proximities S and $\widehat{S}$. We adopt the Kullback-Leibler

divergence

$$D(S||\widehat{S}) = \sum_{i=1}^{N} \sum_{j=1}^{N} \left(S_{ij} \log \frac{S_{ij}}{\widehat{S}_{ij}} - S_{ij} + \widehat{S}_{ij} \right) \tag{6}$$

because it accounts for Poisson noise and thus better for sparsity in S.

The doubly stochastic similarity matrix $\widehat{S}$ in embedding space provides a probabilistic interpretation. Let $W_{ik} = P(\text{topic}_k|\text{word}_i)$, the probability of assigning the ith data object to the kth topic. Without preference to any particular word, we impose a uniform prior $P(\text{word}_j) = 1/N$ over the words. With this prior, we can compute by the Bayes' formula

$$P(\text{word}_j|\text{topic}_k) = \frac{P(\text{topic}_k|\text{word}_j)P(\text{word}_j)}{\sum_{v=1}^{N} P(\text{topic}_k|\text{word}_v)P(\text{word}_v)} \tag{7}$$

$$= \frac{P(\text{topic}_k|\text{word}_j)}{\sum_{v=1}^{N} P(\text{topic}_k|\text{word}_v)}. \tag{8}$$

Then we can see that

$$\widehat{S}_{ij} = \sum_{k=1}^{r} \frac{W_{ik}W_{jk}}{\sum_{v=1}^{N} W_{vk}} \tag{9}$$

$$= \sum_{k=1}^{r} \frac{P(\text{topic}_k|\text{word}_j)}{\sum_{v=1}^{N} P(\text{topic}_k|\text{word}_v)} P(\text{topic}_k|\text{word}_i) \tag{10}$$

$$= \sum_{k=1}^{r} P(\text{word}_j|\text{topic}_k)P(\text{topic}_k|\text{word}_i) \tag{11}$$

$$= P(\text{word}_j|\text{word}_i). \tag{12}$$

That is, if we define a bipartite graph with the words and topics as graph nodes, $\widehat{S}_{ij}$ is the probability that the ith word node reaches the jth word node via a topic node (see Figure 1). It is easy to verify that $\widehat{S}_{ij} = \widehat{S}_{ji}$. Therefore the output similarity matrix $\widehat{S}$ is doubly stochastic.

4   Optimization

We implement the optimization in Eqs. 3 to 5 by multiplicative updates. To minimize an objective $\mathcal{J}$ over a nonnegative matrix W, we first calculate the gradient and separate it into two nonnegative parts ($\nabla_{ik}^+ \geq 0$ and $\nabla_{ik}^- \geq 0$):

$$\nabla_{ik} \stackrel{\text{def}}{=} \frac{\partial \mathcal{J}}{\partial W_{ik}} = \nabla_{ik}^+ - \nabla_{ik}^-. \tag{13}$$

Usually the separation can easily be identified from the gradient. Then the algorithm iteratively applies a multiplicative update rule

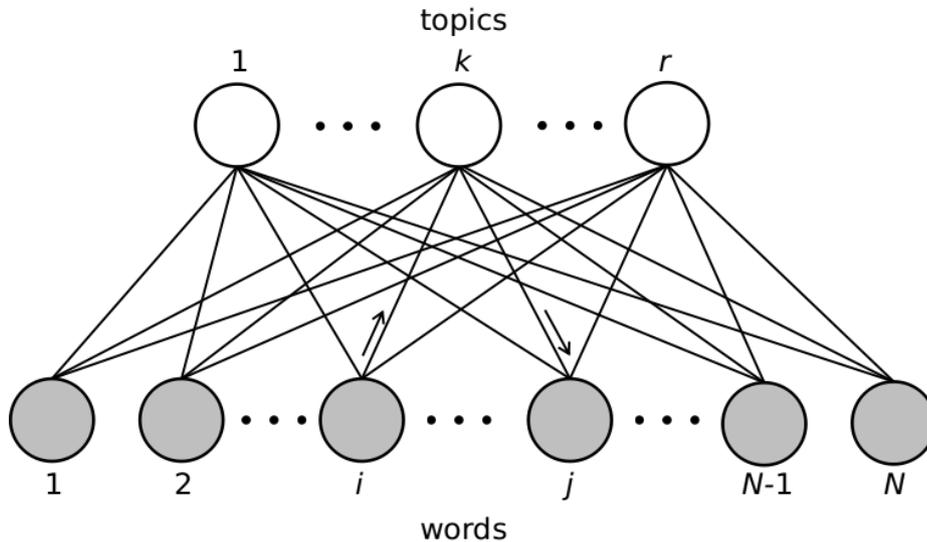$$W_{ik} \leftarrow W_{ik} \frac{\nabla_{ik}^-}{\nabla_{ik}^+} \tag{14}$$

topics

Fig. 1. Word-Topic bipartite graph for N words and r topics ($r < N$). The arrows show a Word-Topic-Word random walk path, which starts at the ith word node and ends at the jth word node via the kth topic node.

until convergence. Such algorithms have several attractive properties, as they naturally maintain the positivity of W and do not require extra effort to tune learning step size. For a variety of NMF problems, such multiplicative updates monotonically decrease $\mathcal{J}$ after each iteration and therefore W can converge to a stationary point [15].

We cannot directly apply the above multiplicative fixed-point algorithm to the proposed learning objective because there are probability constraints on the W rows. Projecting the W rows to the probability simplex after each iteration would often lead to poor local minima in practice.

Instead, we employ a relaxing strategy [18] to handle the probability constraint. We first introduce Lagrangian multipliers $\{\lambda_i\}_{i=1}^N$ for the constraints:

$$\mathcal{L}(W, \lambda) = \mathcal{J}(W) + \sum_i \lambda_i \left(\sum_{k=1}^r W_{ik} - 1 \right).$$ (15)

This suggests a preliminary multiplicative update rule for W:

$$W'_{ik} = W_{ik} \frac{\nabla_{ik}^- - \lambda_i}{\nabla_{ik}^+},$$ (16)

where

$$\frac{\partial \mathcal{J}}{\partial W} = \underbrace{\left[\left(W^T Z W\right)_{kk} s_k^{-2}\right]}_{\nabla_{ik}^+} - \underbrace{\left[2\left(Z W\right)_{ik} s_k^{-1}\right]}_{\nabla_{ik}^-}, \tag{17}$$

with $Z_{ij} = S_{ij}/\widehat{S}_{ij}$ and $s_k = \sum_{v=1}^N W_{vk}$. Imposing $\sum_k W'_{ik} = 1$ and isolating λ_i, we obtain

$$\lambda_i = \frac{b_i - 1}{a_i}, \tag{18}$$

where

$$a_i = \sum_{l=1}^r \frac{W_{il}}{\nabla_{il}^+}, \text{ and }, b_i = \sum_{l=1}^r W_{il} \frac{\nabla_{il}^-}{\nabla_{il}^+}. \tag{19}$$

Putting this λ back in Eq. 16, we obtain

$$W_{ik} \leftarrow W_{ik} \frac{\nabla_{ik}^- a_i + 1 - b_i}{\nabla_{ik}^+ a_i}. \tag{20}$$

To maintain the positivity of W, we add b_i to both the numerator and denominator, which does not change the fixed point and gives the ultimate update rule:

$$W_{ik} \leftarrow W_{ik} \frac{\nabla_{ik}^- a_i + 1}{\nabla_{ik}^+ a_i + b_i}. \tag{21}$$

The above calculation steps are summarized in Algorithm 1. In implementation, one does not need to construct the whole matrix $\widehat{S}$. The ratio $Z_{ij} = S_{ij}/\widehat{S}_{ij}$ only requires calculation on the non-zero entries of S.

The above algorithm obeys a monotonicity guarantee provided by the following theorem.

Theorem 1 *Denote W^{new} the updated matrix after each iteration of Algorithm 1. It holds that $\mathcal{L}(W^{new}, \lambda) \leq \mathcal{L}(W, \lambda)$ with $\lambda_i = (b_i - 1)/a_i$.*

The proof follows the Majorization-Minimization procedure [4, 15, 16] and is a direct corollary of Theorem 2 in [14]. The theorem shows that Algorithm 1 jointly minimizes the approximation error and drives the rows of W towards the probability simplex. The Lagrangian multipliers are adaptively and automatically selected by the algorithm, without extra human tuning effort. The quantities b_i are the row sums of the unconstrained multiplicative learning result, while the quantities a_i balance between the gradient learning force and the probability simplex attraction. Besides convenience, we find that this relaxation strategy works more robustly than the brute-force projection after each iteration.

Algorithm 1 Optimization algorithm of our method

Input: input similarity matrix S, number of topics r, positive initial guess of W.
Output: word embedding matrix W (rows as word vectors).
repeat

$$\widehat{S}_{ij} = \sum_{k=1}^{r} \frac{W_{ik} W_{jk}}{\sum_v W_{vk}}$$

$$Z_{ij} = S_{ij} / \widehat{S}_{ij}$$

$$s_k = \sum_{v=1}^{N} W_{vk}$$

$$\nabla_{ik}^{-} = 2 \left(ZW \right)_{ik} s_k^{-1}$$

$$\nabla_{ik}^{+} = \left(W^T ZW \right)_{kk} s_k^{-2}$$

$$a_i = \sum_{l=1}^{r} \frac{W_{il}}{\nabla_{il}^{+}}, \quad b_i = \sum_{l=1}^{r} W_{il} \frac{\nabla_{il}^{-}}{\nabla_{il}^{+}}$$

$$W_{ik} \leftarrow W_{ik} \frac{\nabla_{ik}^{-} a_i + 1}{\nabla_{ik}^{+} a_i + b_i}$$

until W converges under the given tolerance

---

5 Experiments

To compare the performance of our method with Word2Vec, we train word embeddings on two publicly available datasets and then construct k-nearest neighbor tables for specific word queries. Finally, we perform qualitative analysis on these tables and show that our method is better at capturing semantic relation between the words. The codes used in the experiments are available online[1].

Both training datasets used during the experiments represent a collection of English Wikipedia articles. The first dataset is `WikiText-2` [7], which consists of 2.5M tokens with 33K words in the vocabulary. We also include `text8` dataset[2], which is almost 7 times larger than `WikiText-2` and consists of 17M tokens and 254K words in the vocabulary. During the experiments, we used only top 20K most frequent words for both datasets.

For fair comparison, we followed the same default setting in the original version of Word2Vec[3]. Both methods are trained on the same set of vocabulary words, the word embedding dimension is set to 200 and the size of word neighborhood equals to 8.

The results for `WikiText-2` dataset are presented in Table 1. We can see that sometimes word neighbors produced by Word2Vec are not close semantically to the query words, whereas our method was able to produce much better results. For example, for the word "asteroid" the proposed method produces words like "planets", "orbit" and "spacecraft", which are all related to cosmos, whereas

---

[1] `https://users.aalto.fi/~sedovd1/Matrix_decomp_WE/`
[2] `http://mattmahoney.net/dc/textdata.html`
[3] `https://github.com/tmikolov/word2vec`

Table 1. Seven nearest neighbors for `WikiText-2` dataset using: (top) our method and (bottom) Word2Vec.

word	neighbors
camera	footage, shots, shooting, screen, shoot, setting, showing
zoo	gorillas, bars, spiders, exhibit, Pattycake, sharing, lowland
literature	literary, poets, languages, language, writings, tradition, references
moon	observations, observation, Venus, solar, transit, measurements, atmosphere
coin	coins, dollar, dollars, Mint, purchase, fund, costs
leather	silk, cloth, wrapped, manufactured, mud, synthetic, mills
cold	warm, heat, exposed, hot, winter, falling, dry
spring	winter, summer, kept, fall, brief, Over, arrival
queen	ruler, mentions, kings, supreme, throne, kingdom, succession
asteroid	planets, probe, orbit, spacecraft, NASA, Solar, orbits

word	neighbors
camera	reggae, synthesizers, backup, retro, boots, carriage, bouncing
zoo	griffin, Avis, Reader, headpiece, earthworks, Sleat, Owl
literature	Kannada, writings, Fu, tradition, poets, Vaishnava, historical
moon	skeletal, reactivity, equilibrium, sodium, spectral, infrared, triangular
coin	dollar, convention, solution, price, potential, program, annular
leather	twigs, pipes, longitudinal, bags, triangular, tapered, gum
cold	curved, snow, reaches, beak, rough, bars, loop
spring	autumn, 1900s, 1940s, 1880s, 1870s, 2000s, 1800s
queen	Blanche, Wentworth, diplomat, bodyguard, mathematician, relates, prince
asteroid	molecule, insect, orientation, isotope, triangle, undirected, flash

Word2Vec yields words like "molecule", "insect" and "orientation" that share very little in common. Moreover, it shows that the performance of Word2Vec can be quite poor for small datasets.

Table 2 shows the results for `text8` dataset. We can see that the increase in text corpus size helps to obtain more meaningful embeddings. However, Word2Vec tends to produce rather rare and specific neighbors for the query words, whereas our method produces more common words. For example in case of Word2Vec, the closest neighbors to the word "dracula" are "stoker" and "bram", which constitute the name of the author, who wrote the corresponding novel, as well as "lugosi" and "bela", which are related to the name of the actor portraying Dracula. On the other hand, by using our method, the close neighbors consist of the words "frankenstein" and "godzilla". These three words constitute the group of the iconic horror movie monsters and are well associated with each other.

6   Conclusion

We have proposed a new word embedding method which is based on low-rank decomposition of doubly stochastic similarity matrix in the embedding space. Unlike previous approaches, our method provides not only the low-dimensional word vectors but also their pairwise similarity metric for subsequent applications

Table 2. Seven nearest neighbors for `text8` dataset using: (top) our method and (bottom) Word2Vec.

word	neighbors
green	blue, red, white, yellow, black, color, brown
airport	railway, rail, downtown, airlines, traffic, train, metropolitan
celebrity	interviews, credits, kids, favorite, talent, joy, charity
microsoft	windows, operating, apple, mac, os, dos, macintosh
ancient	greek, middle, historical, pre, latin, medieval, tradition
byzantine	emperors, dynasty, ottoman, conquered, constantinople, conquest, rulers
roman	empire, church, catholic, holy, eastern, christian, ancient
dinosaur	dinosaurs, prehistoric, fossils, habitat, insect, specimen, elephants
dracula	frankenstein, vampire, noir, adaptations, godzilla, cyberpunk, horror
godzilla	sequel, monsters, monster, adventure, anime, horror, robot

word	neighbors
green	shade, lantern, purple, onion, violet, herring, panther
airport	heathrow, ferry, destinations, monorail, airline, flights, hub
celebrity	britney, quiz, vh, listings, portrayals, futurama, syndicated
microsoft	novell, xp, excel, hypercard, borland, netscape, macromedia
ancient	hellenistic, etruscan, sumerian, vedic, mycenaean, phoenician, hellenic
byzantine	achaemenid, seleucid, assyrian, justinian, hittite, heraclius, frankish
roman	byzantine, frankish, aztec, claudian, gaius, aurelius, seleucid
dinosaur	mammal, reptiles, lizard, dodo, zebra, bipedal, skeleton
dracula	stoker, bram, vampire, lugosi, bela, poirot, remake
godzilla	lugosi, toho, miniseries, bela, remake, akira, highlander

such as retrieval. The resulting similarities are explicitly optimized in terms of least Kullback-Leibler divergence to the input similarity matrix. We have proposed an optimization algorithm based on multiplicative updates for minimizing the presented cost function. Experiment results have shown that our method works better for two selected text corpora compared to the state-of-the-art word embedding method in terms of providing more meaningful k-nearest neighbors in the embedding space.

There are several future directions. In this work we have used a batch-mode optimization algorithm, which could be replaced by using distributed and stochastic learning techniques, for example co-distillation [1], towards a more scalable and more efficient method. We could also incorporate Bayesian treatment of the embedding vectors, for example, using Dirichlet priors [10, 14] and automatic rank determination [17]. Moreover, the discrepancy between input and output proximity matrices could be replaced by other learnable information divergence [2]. In addition, our methods is ready to be applied in other domains, for example, finding the embedding vectors of k-mers in DNA sequences.

Acknowledgment

References

1. Anil, R., Pereyra, G., Passos, A., Ormandi, R., Dahl, G.E., Hinton, G.E.: Large Scale Distributed Neural Network Training through Online Distillation. In: International Conference on Learning Representations (2018)
2. Dikmen, O., Yang, Z., Oja, E.: Learning the Information Divergence. IEEE Transactions on Pattern Analysis and Machine Intelligence 37(7), 1442–1454 (2015)
3. Dingwall, N., Potts, C.: Mittens: an Extension of Glove for Learning Domain-Specialized Representations. In: NAACL-HLT. pp. 212–217 (2018)
4. Hunter, D., Lange, K.: A Tutorial on MM Algorithms. The American Statistician 58(1), 30–37 (2004)
5. Ling, W., Dyer, C., Black, A., Trancoso, I.: Two/Too Simple Adaptations of Word2Vec for Syntax Problems. In: NAACL-HLT. pp. 1299–1304 (2015)
6. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
7. Merity, S., Xiong, C., Bradbury, J., Socher, R.: Pointer Sentinel Mixture Models. In: International Conference on Learning Representations (2017)
8. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed Representations of Words and Phrases and their Compositionality. In: Advances in Neural Information Processing Systems. pp. 3111–3119 (2013)
9. Pennington, J., Socher, R., Manning, C.: Glove: Global Vectors for Word Representation. In: Empirical Methods in Natural Language Processing. pp. 1532–1543 (2014)
10. Sinkkonen, J., Aukia, J., Kaski, S.: Component Models for Large Networks. CoRR abs/0803.1628 (2008)
11. Stamatatos, E., Kokkinakis, G., Fakotakis, N.: Automatic Text Categorization in Terms of Genre and Author. Comput. Linguist. 26(4), 471–495 (2000)
12. Stergiou, S., Straznickas, Z., Wu, R., Tsioutsiouliklis, K.: Distributed Negative Sampling for Word Embeddings. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp. 2569–2575 (2017)
13. Turian, J., Ratinov, L., Bengio, Y.: Word Representations: A Simple and General Method for Semi-supervised Learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. pp. 384–394 (2010)
14. Yang, Z., Corander, J., Oja, E.: Low-Rank Doubly Stochastic Matrix Decomposition for Cluster Analysis. Journal of Machine Learning Research 17(187), 1–25 (2016)
15. Yang, Z., Oja, E.: Unified Development of Multiplicative Algorithms for Linear and Quadratic Nonnegative Matrix Factorization. IEEE transactions on Neural Networks 22(12), 1878–1891 (2011)
16. Yang, Z., Peltonen, J., Kaski, S.: Majorization-Minimization for Manifold Embedding. In: International Conference on Artificial Intelligence and Statistics. pp. 1088–1097 (2015)
17. Yang, Z., Zhu, Z., Oja, E.: Automatic Rank Determination in Projective Nonnegative Matrix Factorization. In: International Conference on Latent Variable Analysis and Signal Separation. pp. 514–521 (2010)
18. Zhu, Z., Yang, Z., Oja, E.: Multiplicative Updates for Learning with Stochastic Matrices. In: Proceedings of the 18th Scandinavian Conference on Image Analysis. pp. 143–152 (2013)