Multimedia Systems
https://doi.org/10.1007/s00530-022-00928-4

RESEARCH ARTICLE

PMIVec: a word embedding model guided by point-wise mutual information criterion

Minghong Yao[1,2] · Liansheng Zhuang[1] · Shafei Wang[2] · Houqiang Li[1]

Received: 31 March 2021 / Accepted: 21 February 2022

Abstract
Word embedding aims to represent each word with a dense vector which reveals the semantic similarity between words. Existing methods such as word2vec derive such representations by factorizing the word–context matrix into two parts, i.e., word vectors and context vectors. However, only one part is used to represent the word, which may damage the semantic similarity between words. To address this problem, this paper proposes a novel word embedding method based on point-wise mutual information criterion (PMIVec). Our method explicitly learns the context vector as the final word representation for each word, while discarding the word vector. To avoid the damage of semantic similarity between words, we normalize the word vector during the training process. Moreover, this paper uses point-wise mutual information to measure the semantic similarity between words, which is more consistent with human intuition on semantic similarity. Experiments on public data sets show that our PMIVec model can consistently outperform state-of-the-art models.

Keywords Natural language processing · Word embedding · Point-wise mutual information

1 Introduction

Word embedding is a widespread technique in boosting the performance of modern NLP systems by learning a vector for each word as its semantic feature. The general idea of word embedding is to assign each word with a dense vector. In a qualified word embedding model, the vector similarity tends to reflect the word semantic similarity. These vectors will be either directly used as feature representations or further fine-tuned with training data from downstream supervised tasks [21, 23]. Although contextualized word embedding models like Bert [4] and ELMo [20] have achieved great success, pre-trained word embedding models like fastText [1] remain to be vital in the scenario when the text are too short to be fed into Bert. For example, the state-of-the-art TextVQA models such as M4C [10] use fastText to generate embedding for the OCR tokens in images.

Previous studies such as word2vec [16] mainly focus on how to generate word embedding as a by-product of training a language model. Later, Levy derived that the skip-gram with negative-sampling (SGNS) training method in word-2vec is equivalent to implicitly factorizing a word–context co-occurrence matrix into "word" vectors and "context" vectors [12]. The cells in this word–context co-occurrence matrix are point-wise mutual information (PMI) between words and contexts. The PMI value and its variants have been considered as the best metric to evaluate the semantic similarity between words [11, 22] for a long time. As reported in Levy's paper, the exact factorization with SVD can achieve solutions that are at least as good as SGNS's solutions for word similarity tasks. After word2vec, several variants have been proposed in recent years [17, 18].

However, this paper argues that factorizing the word–context matrix into two different parts will create a gap between training and evaluation stage. During training, both the SGNS model and its variants use the inner product between

Communicated by B.-K. Bao.

✉ Liansheng Zhuang
lszhuang@ustc.edu.cn

Minghong Yao
yaominghong1@gmail.com

Shafei Wang
rockingsandstorm@163.com

Houqiang Li
lihq@ustc.edu.cn

[1] University of Science and Technology of China, Anhui, China

[2] Peng Cheng Laboratory, Shenzhen, China

Published online: 09 June 2022

"word" and "context" vectors to approximate the semantic similarity between words, while in the evaluation stage, people use only "word" vectors and discard the "context" vectors or vice versa. Such gap make the vector similarity between "word" ("context") vectors lack a clear and unambiguous definition, and damages the semantic similarity between words. For example, when the inner product between the "word" vector of "happy" and the "context" vector of "birthday" approximates the PMI value of "happy birthday", there is no reason to believe that the "word-word" inner product between "happy" and "birthday" also approximates the PMI value between them.

To close the above-mentioned gap and quantitatively define the information captured by vector similarity between words, this paper proposes a point-wise mutual information (PMI)-guided word embedding model. We proposed three improvements. First, we normalize the "word" vectors and scale up the "context" vectors in the training stage. This will force the PMI value to be encoded by the "context" vectors mainly. Next, we initialize all "word" and "context" vectors in the way that the angle between any pair of them is no bigger than 30°. Finally, we introduce a new objective function. This new objective function explicitly model the word pair's joint probability conditioned on different context in addition to the conventional uni-gram language models. In this way, our model can capture the word co-occurrence statistics better than current SG models. This paper shows that the resulting vector similarity between "context" vectors approximates the PMI value between words. Therefore, the vector similarity can better reveal the semantic relationship between words as expected.

In summary, the contributions of this paper are as follows:

- A new criterion called point-wise mutual information (PMI) is introduced to describe the human intuition on semantic similarity, which makes it possible to test if the word vectors' similarity can reflect the semantic similarity. To our best knowledge, this is the first attempt to explicitly define a quantitative criterion to measure the quality of a word embedding model.
- Guided by the PMI criterion, this paper develops a novel word embedding model called PMIVec model, which can significantly improve the performance of word embedding and better reveal the semantic relationship between words as expect. Moreover, experiments on public popular data sets also show that our model can outperform state-of-the-art models consistently across both word similarity tasks and sentence embedding tasks.

2 Related work

Mikolov introduces continuous bag of words (CBOW) and skip-gram algorithm to build a language model. In the skip-gram algorithm, they intend to estimate the probability of a context word appearing around the given center word. This is a conditional probability, and therefore, is asymmetric, i.e., $p(w_i|w_j) \neq p(w_j|w_i)$, where w_i, w_j denote different words. To estimate the two different conditional probability separately, each word has two vector representations. One is the "context" vector $\mathbf{O}_i$, and the other is "word" vector $\mathbf{I}_i$. At last, the skip-gram model proposes $p(w_i|w_k) \propto \langle \mathbf{O}_i, \mathbf{I}_k \rangle$ and $p(w_k|w_i) \propto \langle \mathbf{O}_k, \mathbf{I}_i \rangle$.

The GloVe model learns the word vectors by aggregating global word–word co-occurrence matrix from a corpus [19]. Specifically, in the training stage, Glove forces the inner product $\langle \mathbf{O}_i, \mathbf{I}_j \rangle$ to fit w_i and w_j's co-occurrence count C_{ij}. However, Glove will only use $\langle \mathbf{O}_i, \mathbf{O}_j \rangle$ in the downstream tasks and throw $\langle \mathbf{I}_i, \mathbf{I}_j \rangle$ away or vice versa. The co-occurrence count information C_{ij} is not explicitly used in the downstream tasks because $\langle \mathbf{O}_i, \mathbf{I}_j \rangle$ is never used. PMIVec explicitly avoids this problem by forcing $\langle \mathbf{O}_i, \mathbf{O}_j \rangle$ to capture the PMI, which is determined by the global word–word co-occurrence matrix, between w_i and w_j.

The FastText model is proposed to enrich word vectors with subword information, that is, it represents each word with a bag of character n-grams, and the SG model will learn vector representation not only for the whole word but also for each character n-gram [1]. Therefore, this model can compute word vectors for out-of-vocabulary (OOV) words by summing their character n-grams' representation. The FastText model inherits most of the SG's problems.

Xing points out that there is an inconsistency among the SG model's objective function used to learn the word vectors (maximum likelihood based on inner product) and the distance measurement for word vectors (cosine distance). Therefore, they developed a normalization model [24]. By normalizing each word vector during training, the inner product will be equivalent to the cosine distance. However, we argue that this model use only one vector to represent each word, and therefore, it cannot estimate the asymmetrical conditional probability well fundamentally.

In recent years, the contextualized word representation learning [20] and the pre-trained [4] language models achieve the state-of-the-art results in multiple NLP tasks. In these models, one word will have different vector representations based on its context, and therefore,

can handle the polysemy better than the word embedding models. However, the contextualized models cannot perform well for context-free tasks. For example, BERT's score in WordSim353 task is 0.477 while skip-gram's score is 0.711 according to [15]. What is more, we argue that our work can shade light on these models. To be specific, BERT will estimate a masked language model during training time, and any language model will rely on a softmax layer to make prediction. The weight for each word in this softmax layer corresponds to the "context" vector in our model. Therefore, with further analysis, one can extend our model's conclusion to these models and explore the relationship between the softmax layer's weights and the contextualized word representations. This topic goes beyond this paper's scope.

3 Our model

We will introduce the PMI's definition and why it is important in the first subsection. Then, we will review the skip-gram model briefly to reveal why its energy function is problematic in the second subsection. In the third subsection, we will introduce our model and show why our model can solve this problem. At last, we will describe the algorithm of our model in the fourth subsection.

3.1 Point-wise mutual information

PMI is an estimation of how much one word w_i tells about the other word w_k [3]. People developed the notion of word window to help define when two words "co-occur", i.e., w_i and w_k co-occur, only when they appear in the same word window [16]. Let L denote the word window's width. By moving the word window across the whole corpus, we can construct a co-occurrence matrix $[C_{ki}]_{V*V}$, where V is the vocabulary's length. Each entry C_{ki} represents that the word w_i appears in w_k's word window C_{ki} times totally in the whole corpus. Based on these counts, one can calculate how many times the word w_k appears in the corpus totally. Let C_k denote this quantity, then

$$C_k = 1/L \sum_{i=1}^{V} C_{ki}.$$

What is more, let T denote the total number of tokens in this corpus, then one can also calculate $p(w_k)$ and $p(w_i|w_k)$ by

$$p(w_k) = \frac{C_k}{T};\ p(w_i|w_k) = \frac{C_{ki}}{LC_k}. \tag{1}$$

At last, the PMI's definition is

$$\mathrm{PMI}(w_i, w_k) = \log\left(\frac{p(w_i|w_k)}{p(w_i)}\right). \tag{2}$$

The PMI evaluated from co-occurrence counts has a strong linear relationship with human semantic similarity judgments from survey data [8]. Therefore, it is reasonable to associate word embedding with the PMI.

3.2 Skip-gram model revisit

There is a sequence of training words $w_{v_1}, \ldots, w_{v_T}$, in which each word w_{v_k} is chosen from a fixed vocabulary. The vocabulary's size is V. The original objective of SG model is to maximize the average log likelihood

$$\frac{1}{T}\sum_{k=1}^{T}\sum_{i=k-L/2}^{k+L/2} \log\left(p\left(w_{v_i}|w_{v_k}\right)\right),\ i \neq k. \tag{3}$$

For the sake of simplification, let w_i denote w_{v_i}. Then, $p(w_i|w_k)$ is modeled by the Gibbs distribution with $\langle \mathbf{O}_i, \mathbf{I}_k \rangle$ as its energy function

$$p(w_i|w_k) = \frac{\exp(\langle \mathbf{O}_i, \mathbf{I}_k \rangle)}{\sum_{j=1}^{V} \exp(\langle \mathbf{O}_j, \mathbf{I}_k \rangle)}, \tag{4}$$

where $\mathbf{O}_i$ is the "context" vector of word w_i, and $\mathbf{I}_k$ is the "word" vector of word w_k. The conditional probability $p(w_i|w_k)$ describes what are the chances that word w_i appears around word w_k, and is called uni-gram language model.

However, optimizing function (3) is impractical because the denominator of Eq. (4), a summation over V terms (usually 10^5–10^7 terms), is difficult to calculate and optimize [16]. To solve this problem, SG model develops the famous negative-sampling technique by simplifying the noise contrastive estimation (NCE) method [7]. The following theorem shows that based on the energy function used in (4), the negative-sampling method will lead to a problematic solution.

Theorem 1 *Assuming that, for any conditional probability matrix* $\left[p(w_i|w_k)\right]_{V\times V}$, *there exist* $\{\mathbf{O}_i \in \mathcal{R}^d\}_{i=1,\ldots,V}$ *and* $\{\mathbf{I}_k \in \mathcal{R}^d\}_{k=1,\ldots,V}$, *such that* $\forall i, k$, *Eq.* (4) *holds. Then, the optimal solution to SG model's negative-sampling loss will have the following property, that is*

$$\langle \mathbf{O}_i, \mathbf{I}_k \rangle = PMI\left(w_i, w_k\right) - \log K, \tag{5}$$

where the constant K is the number of negative samples per positive sample.

The detailed definition of negative-sampling loss, and Theorem 1's proof can be found in the appendices. Although Levy derived Eq. (5) in [12] as well, they need to assume

that both $\mathbf{O}_i, \mathbf{I}_k$ have infinite dimension, which is obviously unrealistic. Theorem 1 does not require such an assumption.

Although the inner product between w_i's context vector and w_k's word vector is meaningful, what people really use in practice is $\langle \mathbf{O}_i, \mathbf{O}_k \rangle$ or $\langle \mathbf{I}_i, \mathbf{I}_k \rangle$. We argue that this gap will limit the performance of word embedding.

3.3 PMIVec model

We propose to use the inner product between the "context" vectors of two words to approximate the PMI value between them. For any pair of words w_i and w_j, their "context" vector representations, $\mathbf{O}_i$ and $\mathbf{O}_j$, should satisfy the following equation:

$$2 * \langle \mathbf{O}_i, \mathbf{O}_j \rangle = \mathrm{PMI}(w_i, w_j) = \log\left(\frac{p(w_i, w_j)}{p(w_i)p(w_j)}\right). \tag{6}$$

Noticing that both sides of the Eq. (6) are symmetrical about w_i, w_j, this resolves the dilemma described in the Introduction section. What is more, Eq. (6) also quantitatively defines what statistical information between a pair of words has been captured by their vector representations. To achieve Eq. (6), one key observation is that its left hand side is determined by three vectors' norm, i.e.,

$$\mathbf{LHS} = \left\|\mathbf{O}_i + \mathbf{O}_j\right\|_2^2 - \|\mathbf{O}_i\|_2^2 - \left\|\mathbf{O}_j\right\|_2^2,$$

and its right hand side is also determined by three terms, i.e.,

$$\mathbf{RHS} = \log\left(p(w_i, w_j)\right) - \log\left(p(w_i)\right) - \log\left(p(w_j)\right).$$

Therefore, if one can design a Gibbs distribution and its energy function, such that its optimal solution satisfying $\|\mathbf{O}_i\|_2^2 \propto \log(p(w_i))$, then Eq. (6) can be achieved.

This paper proposes to use the following Gibbs distribution to fit $p(w_i|w_k)$:

$$p(w_i|w_k) = \frac{\exp\left(\|\mathbf{O}_i\|_2 \left\langle \mathbf{O}_i, \frac{\mathbf{I}_k}{\|\mathbf{I}_k\|_2} \right\rangle\right)}{\sum_{j=1}^{V} \exp\left(\|\mathbf{O}_j\|_2 \left\langle \mathbf{O}_j, \frac{\mathbf{I}_k}{\|\mathbf{I}_k\|_2} \right\rangle\right)}. \tag{7}$$

The sketch proof of why (7) works better than (4) is that its optimal solutions satisfy the following equation:

$$p(w_i|w_k) * Z_k = \exp\left(\left\|\mathbf{O}_i\right\|_2^2 * \cos\left(\theta_{ik}\right)\right), \tag{8}$$

where Z_k is the denominator in (7), and θ_{ik} is the angle between $\mathbf{O}_i$ and $\mathbf{I}_k$. Then, the expectation of both sides of (8) with respect to $p(w_k)$ is

$$\sum_{k=1}^{V} p(w_i, w_k) Z_k = \sum_{k=1}^{V} \exp\left(\|\mathbf{O}_i\|_2^2 \cos(\theta_{ik})\right) p(w_k). \tag{9}$$

If $\forall i, k$, $\theta_{ik} \leq 30°$, then no matter what is the distribution of $p(w_k)$, the RHS of (9) will approximate to $\exp\left(\|\mathbf{O}_i\|_2^2\right)$, because $\cos(\theta_{ik}) \approx 1$. This assumption can be assured by properly initializing and regularizing the angle between $\mathbf{O}_i$ and $\mathbf{I}_k$. The LHS of (9) is approximated by $p(w_i)\hat{Z}$, where $\hat{Z}$ is the mean value of all Z_k. One important phenomenon about the partition functions, Z_k, is that they tend to concentrate around the mean value. The rigorous proof can be found in the chapter 7 of Ma's Ph.D. thesis [14]. Finally, the log value of (9)'s both sides are

$$\log\left(p(w_i)\right) + \log(\hat{Z}) \approx \left\|\mathbf{O}_i\right\|_2^2. \tag{10}$$

The analysis above briefly shows why the energy function $\|\mathbf{O}_i\|_2 \left\langle \mathbf{O}_i, \frac{\mathbf{I}_k}{\|\mathbf{I}_k\|_2} \right\rangle$, and the proper initialization are necessary for property (6). The analysis about $\|\mathbf{O}_i + \mathbf{O}_j\|_2^2$ is similar. To assure the property (6), one also needs to fit $p(w_i, w_j|w_k)$ using the following Gibbs distribution

$$p(w_i, w_j|w_k) = \frac{\exp\left(\|\mathbf{O}_i + \mathbf{O}_j\|_2^2 \cos(\theta_{(ij)k})\right)}{Z'_k}, \tag{11}$$

where $\theta_{(ij)k}$ is the angle between $\mathbf{O}_i + \mathbf{O}_j$ and $\mathbf{I}_k$, and Z'_k's definition is

$$Z'_k = \sum_{(m,n)}^{C_V^2+V} \exp\left(\left\|\mathbf{O}_m + \mathbf{O}_n\right\|_2^2 \cos\left(\theta_{(mn)k}\right)\right). \tag{12}$$

The following theorem shows that, based on the 3 modification proposed by this paper, there exist solutions such that they have property (6).

Theorem 2 *Assuming that, for any conditional probability matrix* $[p(w_i|w_k)]_{V\times V}$ *and* $[p(w_i, w_j|w_k)]_{(C_V^2+V)\times V}$, *there exist* $\{\mathbf{O}_i \in \mathcal{R}^d\}_{i=1,\ldots,V}$ *and* $\{\mathbf{I}_k \in \mathcal{R}^d\}_{k=1,\ldots,V}$, *such that*

- $\forall i, j, k$, *the angles* $\theta_{ik}, \theta_{jk}, \theta_{(ij)k} \leq 30°$,
- $\forall i, j, k$, *the Eq.* (7), *and* (11) *both hold*,

then Eq. (6) *holds for any pair of words* w_i, w_j.

The detailed proof of theorem 2 can be found in the appendices.

3.4 PMIVec algorithm

To fit $p(w_i|w_k)$ and $p(w_i, w_j|w_k)$, PMIVec algorithm adopts the NCE method [7] instead of the negative sampling technique. The resulted loss function consists of two parts.

The first part of loss is to fit $p(w_i|w_k)$

$$\log \sigma(s(i;k)) + \sum_{t}^{K} \log \sigma(-s(t;k)), \tag{13}$$

where $s(i;k) = \frac{\|\mathbf{O}_i\|_2}{\|\mathbf{I}_k\|_2} \langle \mathbf{O}_i, \mathbf{I}_k \rangle - \log Z_k - \log K p_n(w_i)$. The Z_k is treated as a constant to be optimized, and $p_n(w_i)$ is an arbitrary noise distribution. The first part of loss is denoted as L_1.

The second part of loss is to fit $p(w_i, w_j|w_k)$

$$\log \sigma(s(i,j;k)) + \sum_{m,n}^{K^2} \log \sigma(-s(m,n;k)), \tag{14}$$

where

$$s(i,j;k) = \frac{\|\mathbf{O}_i+\mathbf{O}_j\|_2}{\|\mathbf{I}_k\|_2} \left\langle \mathbf{O}_i + \mathbf{O}_j, \mathbf{I}_k \right\rangle - \log Z'_k - \log K^2 p_n(w_i) p_n(w_j).$$

The second part of loss is denoted as L_2.

Algorithm 1 PMIVec $(K, \mathbf{O}_l, \mathbf{I}_l, l = 1, \cdots, |V|)$

```
1: for epoch in epochs do
2:   for w_k in corpus do
3:     I_k ← {O_l}_{l=1,...,|V|};
4:     for w_i in Window(w_k) do
5:       O_i ← {O_l}_{l=1,...,|V|};
6:       O_j ← {O_l}_{l=1,...,|V|};
7:       L1+ = L1(s(i;k));
8:       L2+ = L2(s(i,j;k));
9:       Gradient update of O_i, O_j.
10:      for t in range(K) do
11:        w_t ~ p_n(w_t);
12:        O_t ← {O_l}_{l=1,...,|V|};
13:        L1+ = L1(s(t;k));
14:        Gradient update of O_t.
15:      for m in range(K) do
16:        for n in range(K) do
17:          w_m ~ p_n(w_m), w_n ~ p_n(w_n);
18:          O_m ← {O_l}_{l=1,...,|V|};
19:          O_n ← {O_l}_{l=1,...,|V|};
20:          L2+ = L2(s(m,n;k));
21:          Gradient update of O_m, O_n.
22:    Riemann gradient update of I_k.
```

As shown in algorithm 1, "←" means we are querying a dictionary, "~" represents sampling from a distribution. The definition of Riemann gradient update in line 22 can be found in [15].

Inline 3 of algorithm 1, PMIVec retrieves word w_k's "word" vector $\mathbf{I}_k$. PMIVec also does the retrieval operation for word w_i, w_j, w_t, w_m, w_n's "context" vectors $\mathbf{O}_\cdot$ respectively inline 5, 6, 12, 18, and 19.

From line 4 to line 9, PMIVec samples the context words pair (w_i, w_j) from w_k's word window and does the corresponding gradient update. More specifically, PMIVec samples w_i iteratively for $L1$ firstly, and then it samples another context word w_j randomly from w_k's word window.

From lines 10 to 14, PMIVec samples K negative samples w_t from the whole vocabulary for $L1$ and updates the gradient. Inline 11 of algorithm 1, w_t is sampled from the negative distribution $p_n(w_t)$.

PMIVec also does the negative-sampling operation inline 17. From lines 15 to 21, PMIVec samples K^2 negative sample pairs (w_m, w_n) from the whole vocabulary for $L2$ and updates the gradient.

4 Experiments

In this section, we will validate the effectiveness of our word embedding model on context-free tasks, and compare with state-of-the-art models, including the SG model, the FastText model,[1] the GloVe model,[2] and the JoSE model.[3] Another SG model's embedding with extra training epochs is also reported for the sake of fairness. We will also include BERT's results reported in JoSE just for comparison. Our model is denoted as **PMIVec**.

For all the models, we set the word window width to be 10, the negative samples to be 5, the epochs to be 5, and the word vector's dimension to be 100. Since our model would update 3 positive examples and 30 negative examples each time, we will train an extra SG model's embedding with 15 epochs. What is more, the negative samples for this SG model are also set to be 30. The other hyper-parameters are set to be their default values.

The data set used in our experiments is the MBTA (Massachusetts Bay Transportation Authority) corpus, which is crowed from the web[4] and is a subset of the GloWbe corpus.[5] The MBTA corpus is well cleaned and contains about 700 million tokens. The minimum vocabulary count is set to be 100, i.e., we discard the words appears less than 100 times in the corpus. Note here that, we did not evaluate our model on the famous Wiki training corpus[6] as some previous work [15, 16]. This is because the script provided to transform its format from XML to text is problematic. By looking at the resulting vocabulary of the Wiki corpus carefully, one can find that many non-English characters remains to exist. We believe that these characters would introduce a lot of noises to the samples of $p(w_k), p(w_i|w_k)$, and $p(w_i, w_j|w_k)$, and are harmful to the evaluation of word embedding models.

In Sect. 4.1, to directly check whether our model's embedding fits the PMI between words better than the

[1] https://github.com/facebookresearch/fastText.

[2] https://github.com/stanfordnlp/GloVe.

[3] https://github.com/yumeng5/Spherical-Text-Embedding.

[4] https://mbta.com.

[5] https://www.english-corpora.org/glowbe/.

[6] https://dumps.wikimedia.org/enwiki/.

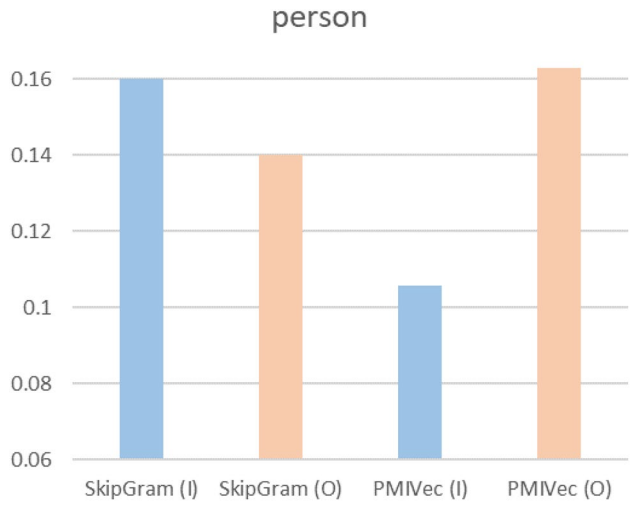

Fig. 1 Pearson correlation coefficient between the vector similarity and the PMI value in large vocabulary

Table 1 Pearson correlation coefficient rank on word similarity evaluation

Models	MEN	WS353	SimLex	RW	RG65
SG(O)	0.622	0.579	0.286	0.397	0.622
SG(I)	0.734	0.608	0.319	0.382	0.714
SGE(O)	0.653	0.568	0.279	0.390	0.567
SGE(I)	0.741	0.576	0.304	0.369	0.708
GloVe(I)	0.624	0.483	0.282	0.285	0.610
Fast(I)	0.726	0.574	0.286	0.379	0.709
JoSE(I)	0.727	**0.681**	0.301	0.325	0.645
BERT(I)	0.594	0.477	0.287	–	–
PMIVec(O)	0.712	0.649	0.329	**0.414**	0.678
PMIVec(I)	**0.743**	0.596	**0.342**	0.295	**0.723**

The bold values mean the best performance

original word2vec model, we calculate the Pearson correlation coefficient between the word–word PMI and word–word vector similarities. Section 4.2 presents the performance of all embedding models in the word similarity tasks. Section 4.3 shows the performance of all embedding models in the sentence embedding tasks.

4.1 Quantitative model evaluation

In this subsection, we propose to judge the word similarity between words according to the PMI between them, and this is a quantity that has a specific definition and can be calculated for any collection of corpus. We implement a script to estimate the PMI between words from the raw corpus according to the definition in Eq. (2). Some words are so rare, that they may never appear around another word, i.e., $p(i|k)$ may be zero. For these cases, we simply ignore these word pairs. At last, we can calculate the Pearson correlation coefficient between the vector similarities and the word pair's PMI value.

The results are presented in Fig. 1. We can observe that the PMIVec model's context vectors have the highest Pearson value, which means that they have captured the PMI information between words better than the rest vectors.

What is more, in the original SG model, the word vectors are more consistent with the PMI values than the context counterparts. However, the situation are opposite in the PMIVec model. The context vectors are the more consistent one. This is not surprising because we have normalized the word vectors so that the context vectors can capture more information about the co-occurrence information between words. Despite the seemingly contradictory results between Table 1 and Fig. 1, we argue that this is because of the vocabulary size difference between them. To be specific, the maximum vocabulary size in Table 1 is 1000 pairs of words, while Fig. 1's vocabulary size is above 10,000. Therefore, with more noise in the vocabulary, the Pearson correlation coefficient would decrease dramatically, and appears to be contradict to some local results as presented in Table 1. In fact, in the sentence embedding tasks, the context vectors are better than the word vectors constantly in the PMIVec model.

4.2 Word similarity tasks

To compare the quality of different models' embedding, previous papers tend to exam whether the word vector similarities agree with human's judgements. For example, the test part of MEN data set [2] contains 1000 pairs of words together with human-assigned similarity judgments, such an example looks like "bird-insect-37.0", where "37.0" is the human-assigned similarity score. Then, the previous papers would calculate the vector similarity scores for these word pairs. At last, they would calculate the Pearson correlation coefficient between the vector similarity scores and the human-assigned similarity scores.

We also validate the performance of different models in a serious popular word similarity tasks, including the Word Similarity353 data set [6], the MEN data set [2], the SimLex999 data set [9], the RW data set [13], and the RG65 data set [13]. The different data sets are all human annotated but with different scales and coverage. Some words in these testing data sets do not appear in our training corpus, and this means we cannot calculate the inner product between vectors for those words. To provide comparable results, we simply remove these words, and calculate the Pearson correlation coefficient among the remaining words. What is more, for the BERT model, we simply adopt the results reported in

Table 2 Pearson correlation coefficient rank on sentence similarity evaluation

Models	STS 14	STS 15	STS 16	SICK-R
SG(O)	0.505	0.5451	0.5023	0.6348
SG(I)	0.5248	0.5697	0.5398	0.6352
SGE(O)	0.5218	0.5678	0.5251	0.6214
SGE(I)	0.5299	0.5753	0.5461	0.6387
GloVe(I)	0.4137	0.4744	0.4126	0.6297
Fast(I)	0.5375	0.5848	**0.5507**	0.6236
JoSE(I)	0.5182	0.5493	0.4854	0.6437
BERT(I)	0.4098	0.4715	0.4606	0.5227
PMIVec(O)	**0.5429**	**0.6016**	0.5455	**0.6543**
PMIVec(O)	0.5363	0.5747	0.5416	0.6366

The bold values mean the best performance

Table 3 Precision rank on sentence classification evaluation

Models	MR	CR	MPQA	SUBJ
SG(O)	70.83	72.53	85.6	86.88
SG(I)	71.66	72.9	85.51	87.21
SGE(O)	70.32	69.38	85.07	86.86
SGE(I)	70.46	70.73	85.24	87.59
GloVe(I)	70.55	72.03	84.43	86.57
Fast(I)	70.65	72.66	85.19	87.02
JoSE(I)	72.26	74.17	85.88	88.16
BERT(I)	71.53	73.25	84.66	87.37
PMIVec(O)	**73.11**	**75.02**	85.7	**88.48**
PMIVec(I)	71.77	71.66	**86.17**	87.6

The bold values mean the best performance

JoSE. Even though we used different training corpus, and our training corpus is smaller than the wiki corpus, these results can still validate our points here that the contextualized word representation learning model performs poorly in the context-free tasks.

We can observe that for most data sets, our model can outperform the SOTA models. For the sake of fairness, we have included an extra version of the SG model with more epochs and more negative-sampling numbers (see Tables 2 and 3).

4.3 Sentence embedding tasks

The purpose of this subsection is to explore whether our embedding can boost the performance of the downstream NLP tasks better than the other word embedding models. Since our embedding shows superiority in the word similarity tasks, it is reasonable to believe that we can extend this advantage to sentence embedding tasks with simple generating procedural. We choose the bag-of-words (bow) model to generate the sentence embedding from the word embedding with simple procedure like averaging the word embedding. More complex sentence embedding generation method goes beyond this paper's scope. Therefore, BERT or Elmo's results will not be reported here. There are two kinds of sentence embedding task. One is the sentence relatedness task, and the other one is the sentence classification task. For the relatedness tasks, we evaluate how the cosine distance between two sentences correlates with a human-labeled similarity score. For the classification tasks, we use a multi-layer perceptron which feeds on the sentence embedding.

The test data sets of relatedness tasks include the STS 14, 15, 16, and the SICK-Relatedness. Similar to the word similarity tasks, there is a premise sentence and a hypothesis sentence in these data sets. Take STS 14 for example, one such premise sentence is "Liquid ammonia leak kills 15 in Shanghai". The corresponding hypothesis sentence is "Liquid ammonia leak kills at least 15 in Shanghai". The label for this pair of sentence is 4.6. The minimum score is 0, and the maximum score is 5.

For the classification tasks, we use the MR, CR, MPQA, and SUBJ data sets to test our model's performance. The MR data set is about sentiment classifications over movies' reviews. For example, the sentence, "Too slow for a younger crowd, too shallow for an older one.", has a negative label. The rest test data sets are similar, while they are collected from different domains.

As we can see, our model can outperform the rest models in most of the data sets by a large margin except for the STS 16 data set. In the sentence embedding tasks, the context vector of our model is always better than the word vector counterpart except in the MPQA data set. We argue that it is because each sample in this data set is rather a phrase than a sentence. For example, a positive record would look like "liberation and independence", and the negative one would be like "suffering from some intoxication".

5 Conclusion

In this paper, we introduce PMI as a new criterion to describe the human intuition on semantic similarity, which makes it possible to test if the word vectors' similarity can reflect the semantic similarity. Guided by the PMI criterion, this paper develops a novel word embedding model called PMIVec model. Different from previous work, our model explicitly models the word pair's joint probability conditioned on different context in addition to the conventional uni-gram language model. Our model can capture the word co-occurrence statistics better than current SG models. Besides, the resulting vector similarity between any two

words is more consistent with human's expectation. Experiments show that our model can improve the performance of word embedding on the word similarity tasks, sentence embedding tasks.

Appendix

Proof of Theorem 1

There are two steps to prove Theorem 1. The first step is to formulate the loss function of SG model by applying the noise contrastive estimation (NCE) method [7] to equation (3) and (4). The second step is to reveal that the simplification made by SG's negative sampling technique will lead to property (5).

Step 1

To find the assumed optimal solutions $\{\mathbf{O}_i, \mathbf{I}_k\}_{i,k=1,\ldots,V}$, and yet to avoid directly computing the denominator of Eq. (4), which will be denoted as Z_k, NCE proposes to treat it as a constant number to be estimated, and one can estimate $\{\mathbf{O}_i, \mathbf{I}_k, Z_k\}_{i,k=1,\ldots,V}$ via solving a supervised learning task.

One can draw T_d real samples from $p(\cdot|w_k)$, and $K * T_d$ noise samples from the known noise distribution $p_n(\cdot)$, where $p_n(\cdot)$ can be chosen to be any distribution and K is a constant like 5. Each sample will have a label y, and $y = 1$ stands for real sample; $y = 0$ stands for noise sample. Then, one can mix these samples together and pick one of them, asking whether it is a real sample or not.

One can train a logistic regression model, whose parameters are $\{\mathbf{O}_i, \mathbf{I}_k, Z_k\}_{i,k=1,\ldots,V}$, to discriminate the real samples from the noise samples. For a real sample w_i, the logistic regression model's prediction on it being true is

$$p(y=1|w_i;w_k) = \sigma\big[\log(\exp(\langle \mathbf{O}_i, \mathbf{I}_k\rangle)) - \log(Z_k) - \log(Kp_n(w_i))\big].$$

According to NCE's conclusion, if one treats Z_k as an additional scalar parameter that can be optimized, then the following optimization problems (minimizing the cross entropy) will have the same solution as to (3)

$$-\mathbf{E}_{i\sim p_d} \log p(1|w_i;w_k) - K\mathbf{E}_{t\sim p_n} \log p(0|w_t;w_k) \tag{15}$$

where $\forall k = 1, \ldots, V$,

$$p(1|w_i;w_k) = \sigma\big[\langle \mathbf{O}_i, \mathbf{I}_k\rangle - \log Z_k - \log Kp_n(w_i)\big],$$

and p_d represents the true distribution of $p(\cdot|w_k)$. One can sample w_i from p_d with word window's help.

Step 2

The following part will show how the negative-sampling technique adopted by the skip-gram model can lead to Eq. (5).

The SG model simplifies the calculation of $p(1|w_i;w_k)$ by omitting $\log Z_k$ and $\log Kp_n(w_i)$. This means that the logistic model assumes $Kp_n(w_i) = 1$ [5], i.e.,

$$\begin{aligned} p\big(y=1|w_i;w_k\big) &= \sigma(\langle \mathbf{O}_i, \mathbf{I}_k\rangle) = \tfrac{\exp(\langle \mathbf{O}_i, \mathbf{I}_k\rangle)}{\exp(\langle \mathbf{O}_i, \mathbf{I}_k\rangle)+1}, \\ \hat{p}_d\big(y=1|w_i;w_k\big) &= \tfrac{p_d(w_i|w_k)}{p_d(w_i|w_k)+Kp_n(w_i)}. \end{aligned}$$

where $\hat{p}_d(1|w_i;w_k)$ represents the true probability of sample w_i being true, and it can be derived by applying the Bayesian formula to it. It is easy to show that (15) is equivalent to

$$\sum_i \rho * KL\big\{\hat{p}_d\big(y|w_i;w_k\big)\|p\big(y|w_i;w_k\big)\big\},$$

where $\rho = p_d(w_i;w_k) + Kp_n(w_j)$. Obviously, the above objective will equal to 0 iff

$$\hat{p}_d\big(y|w_i;w_k\big) = p\big(y|w_i;w_k\big),\ y = 1, 0. \tag{16}$$

Noticing that SG chooses $p(w_k)$ as the noise distribution, and with some simple rearrangement, the Eq. (16) implies

$$\langle \mathbf{O}_i, \mathbf{I}_k\rangle = \mathrm{PMI}(w_i, w_k) - \log K.$$

Proof of Theorem 2

There are two steps to prove Theorem 2. The first step is to show the following equations hold for the assumed solutions $\{\mathbf{O}_i, \mathbf{I}_k\}_{i,k=1,\ldots,V}$

$$\log\big(p(w_i)\big) + \log\big(\hat{Z}\big) = \bar{\alpha}_i * \|\mathbf{O}_i\|_2^2, \tag{17}$$

$$\log\big(p(w_j)\big) + \log\big(\hat{Z}\big) = \bar{\alpha}_j * \|\mathbf{O}_j\|_2^2, \tag{18}$$

$$\log\big(p(w_i, w_j)\big) + \log\big(\hat{Z}'\big) = \bar{\alpha}_{ij} * \|\mathbf{O}_i + \mathbf{O}_j\|_2^2, \tag{19}$$

where $\hat{Z}$'s definition can be found in the Eq. (10), $\hat{Z}'$ is the mean value of all $\hat{Z}'_k$, and $\hat{Z}'_k$ is defined in the Eq. (12). $\bar{\alpha}_i$, $\bar{\alpha}_j$, and $\bar{\alpha}_{ij}$ are both constant number. The second step is to show that

$$\begin{aligned} \hat{Z}' &\approx \hat{Z}^2, \\ \bar{\alpha}_i &\approx \bar{\alpha}_j \approx \bar{\alpha}_{ij}, \end{aligned}$$

and therefore, (19), (18), and (17) will prove this theorem.

Step 1

Similar to the analysis in the PMIVec section's sketch proof, the assumed solution $\{\mathbf{O}_i, \mathbf{I}_k\}_{i,k=1,\ldots,V}$ satisfies

$$\begin{aligned}\sum_{k=1}^{V} p(w_i, w_k) Z_k &= \sum_{k=1}^{V} \exp\left(\|\mathbf{O}_i\|_2^2 \cos(\theta_{ik})\right) p(w_k),\\ \sum_{k=1}^{V} p(w_i, w_j, w_k) Z'_k &= \\ &\sum_{k=1}^{V} \exp\left(\left\|\mathbf{O}_i + \mathbf{O}_j\right\|_2^2 \cos(\theta_{(ij)k})\right) p(w_k).\end{aligned}$$

Then the α_i is the mean value of $\cos(\theta_{ik})$, α_j is the mean value of $\cos(\theta_{jk})$, and α_{ij} is the mean value of $\cos(\theta_{(ij)k})$.

The log value of both sides of these equations leads to Eqs. (17), (18), and (19).

Step 2

It is obvious that $\bar{\alpha}_i \approx \bar{\alpha}_j \approx \bar{\alpha}_{ij}$ because of the assumption about θ_{ik}, θ_{jk}, and $\theta_{(ij)k}$.

Noticing that each term of Z'_k is bigger than each term of Z_k^2, and meanwhile that the total number of Z'_k is smaller than the total number of Z_k^2. Therefore, $Z'_k \approx Z_k^2$. This leads to the conclusion that $\hat{Z}' \approx \hat{Z}^2$.

Acknowledgements This work was supported in part to Dr. Liansheng Zhuang by NSFC under contract No.U20B2070 and No.61976199, and in part to Dr. Houqiang Li by NSFC under contract No.61836011.

References

1. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017)
2. Bruni, E., Tran, N.-K., Baroni, M.: Multimodal distributional semantics. J. Artif. Intell. Res. **49**, 1–47 (2014)
3. Church, K.W., Hanks, P.: Word association norms, mutual information, and lexicography. Comput. Linguist. **16**(1), 22–29 (1990)
4. Devlin, J., Chang, M.-W., Lee, K., Kristina, T.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805, (2018)
5. Dyer, C.: Notes on noise contrastive estimation and negative sampling. arXiv:1410.8251, (2014)
6. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., Ruppin, E.: Placing search in context: the concept revisited. ACM Trans. Inf. Syst. **20**(1), 116–131 (2002)
7. Gutmann, M.U., Hyvärinen, A.: Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. J. Mach. Learn. Res. **13**, 307–361 (2012)
8. Hashimoto, T.B., Alvarez-Melis, D., Jaakkola, T.S.: Word embeddings as metric recovery in semantic spaces. Trans. Assoc. Comput. Linguist. **4**, 273–286 (2016)
9. Hill, F., Reichart, R., Korhonen, A.: Simlex-999: evaluating semantic models with (genuine) similarity estimation. Comput. Linguist. **41**(4), 665–695 (2015)
10. Hu, R., Singh, A., Darrell, T., Rohrbach, M.: Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9992–10002, (2020)
11. Kolesnikova, O.: Survey of word co-occurrence measures for collocation detection. Comput. Sist. **20**(3), 327–344 (2016)
12. Levy, O., Goldberg, Y.: Neural word embedding as implicit matrix factorization. In: Ghahramani, I.Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in neural information processing systems. Curran Associates Inc. **27**, 2177–2185 (2014)
13. Luong, M.-T., Socher, R., Manning, C.D.: Better word representations with recursive neural networks for morphology. In: Proceedings of the Seventeenth Conference on Computational Natural Language Learning, pp. 104–113, (2013)
14. Ma, T.: Non-convex optimization for machine learning: design, analysis, and understanding. PhD thesis, Princeton University, (2017)
15. Meng, Y., Huang, J., Wang, G., Zhang, C., Zhuang, H., Kaplan, L., Han, J.: Spherical text embedding. In: Wallach, H., Larochelle, H., Beygelzimer, A., Alche, F., Fox, E., and Garnett, R (eds.) Advances in neural information processing systems. Curran Associates, Inc., **32**, 8206–8215 (2019)
16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in neural information processing systems. Curran Associates, Inc., **26**, 3111–3119 (2013)
17. Mnih, A., Kavukcuoglu, K.: Learning word embeddings efficiently with noise-contrastive estimation. In: Burges, C.J., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K.Q. (eds.) Advances in neural information processing systems. Curran Associates, Inc., **26**, 2265–2273 (2013)
18. Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient non-parametric estimation of multiple embeddings per word in vector space. arXiv:1504.06654, (2015)
19. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In : Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, (2014)
20. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv:1802.05365, (2018)
21. Socher, R., Bauer, J., Manning, C.D., et al.: Parsing with compositional vector grammars. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 455–465, (2013)
22. Terra, E.L., Clarke, C.L.A.: Frequency estimates for statistical word similarity measures. In: Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, pp. 244–251, (2003)
23. 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. Association for Computational Linguistics, (2010)
24. Xing, C., Wang, D., Liu, C., Lin, Y.: Normalized word embedding and orthogonal transform for bilingual word translation. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1006–1011, (2015)