An Automatic Dimensionality Selection for Static Word Embeddings Using Mixed Product Distance

Lingfeng Shen[1] **Ze Zhang**[2] **Ying Chen**[3*]

[1]College of Science, China Agricultural University, China
[2]Department of Computer Science and Technology, Tsinghua University, China
[3]College of Information and Electrical Engineering, China Agricultural University, China

Abstract

Although dimension is a key factor determining word embedding quality, dimension selection for word embeddings is less discussed. In order to efficiently select a proper dimension for static word embeddings used in various NLP tasks, this paper proposes an MPD-based dimensionality selection method that uses a novel metric (Mixed Product Distance, MPD) to select a proper dimensionality for static word embedding algorithms automatically. In the MPD, a scaling scheme is applied to alleviate the approximation error of the metric, and an adaptation design is made to utilize post-processing, an important process to improve the quality of word embeddings. Experiments on word intrinsic functionality tests and downstream tasks demonstrate the effectiveness and the efficiency-performance trade-off of our MPD-based dimensionality selection method.

Introduction

Word embedding has been widely used in numerous NLP tasks, such as recommendation systems (Zhang et al. 2019), text classification (Chung and Glass 2018), information retrieval (Palangi et al. 2016) and machine translation (Guo et al. 2019). Compared to the popularity of dynamic word embeddings like BERT (Devlin et al. 2018), static word embeddings, such as Word2Vec (Mikolov et al. 2013) and GloVe (Pennington, Socher, and Manning 2014), are still very useful, particularly in a resource-limited scenario. Thus, in this paper, we focus on the training of static word embeddings and examine their effectiveness on various NLP tasks.

Although there are many algorithms developed to train static word embeddings, the impact of dimensionality on word embeddings has not yet been fully investigated. The study of Yin (Yin and Shen 2018) has shown that a critical hyper-parameter, the choice of dimensionality for word vectors, has a significant impact on the performance of a model that builds on the word embedding. A low-dimensional word embedding is probably not expressive enough to represent the meanings of words, and in contrast, a high-dimensional word embedding is expressive but possibly leads to overfitting and high computational cost. For most NLP models building on word embeddings, dimensionality is selected ad hoc or by grid search, either of which can result in sub-optimal model performance. Therefore, it is necessary to establish a dimensionality selection procedure that selects a proper dimensionality for word embedding algorithms without training NLP models.

To our best knowledge, dimensionality selection for embeddings was first investigated by Yin (Yin and Shen 2018). They proposed a metric called **Pairwise Inner Product (PIP) loss** which attempts to reflect the relationship between the dimensionality and the quality of word embeddings, and then developed a PIP-based dimensionality selection method that selects a dimensionality with the minimal PIP loss. Then, Wang (Wang 2019) proposed a dimensionality selection method for static word embeddings based on Principal Components Analysis (PCA). However, both the PIP-based dimensionality selection and the PCA-based dimensionality selection suffer from the sensitivity of selecting a proper dimensionality. E.g., the effectiveness of the PIP-based dimensionality selection is affected much by the exponent parameter α. As α becomes smaller, the approximation computation for the PIP loss becomes more sensitive. As a result, it is more likely to select an improper dimensionality and generate a low-quality word embedding because of a large approximation error.

Furthermore, the studies of post-processing (Mu and Viswanath 2018; Liu, Ungar, and Sedoc 2019) have shown that post-processing that attempts to diminish the influence from word frequency in a word embedding is significant to improve the quality of the embedding. In a word embedding trained by popular static word embedding algorithms, frequent words often dominate some components (namely principal components, PCs), and these PCs hurt the representation capability of the embedding. In order to alleviate the impact of these PCs, a post-processing procedure, such as the All-but-the-top (ABTT) (Mu and Viswanath 2018), and Conceptor Negation (CN) (Liu, Ungar, and Sedoc 2019), is often applied to the trained embedding. However, post-processing has not been adopted in current dimensionality selection methods.

In order to overcome these problems, we propose MPD-based dimensionality selection, which minimizes a metric called **Mixed Product Distance (MPD)** to find a proper dimensionality. The MPD mainly follows the idea of the PIP

[*] the corresponding author

loss, and yet has the following two modifications. In order to enhance the robustness of selecting a proper dimensionality, the MPD alleviates the approximation error of the PIP loss through a scaling scheme. Then, in order to adopt post-processing, the MPD applies a post-process function to the PIP loss. Finally, besides word intrinsic functionality tests (e.g., word relatedness and word analogy) which are commonly used to test the effectiveness of dimensionality selection in previous work, we examine the MPD-based dimensionality selection on different downstream NLP tasks. Our intensive experiments demonstrate that the MPD-base dimensionality selection method can consistently achieve good performances.

Above all, the main contributions of the paper can be outlined as follows:

- We design a novel metric, called Mixed Product Distance (MPD), which uses a scaling scheme and a post-process function to overcome the two problems of the PIP loss (the approximation error and the post-processing adoption), respectively.
- We propose an MPD-based dimensionality selection method that can work well with various static word embedding algorithms to generate high-quality word embeddings.
- We examine the MPD-based dimensionality selection on different tasks, including word intrinsic functionality tests and downstream NLP tasks. Both tests demonstrate the effectiveness and the efficiency-performance tradeoff of the method.

Related Work

Static Word Embedding

Most existing static word embedding algorithms can be formulated as matrix factorization as follows, either explicitly or implicitly. Assuming M be a **signal** matrix (e.g., the PMI matrix) and $M = UDV^T$ be its SVD decomposition, through truncating the matrix U at dimension k, and multiplying it by a power of the truncated diagonal matrix D, an embedding X with k dimensionality is obtained, $X = U_{1:k}D^{\alpha}_{1:k,1:k}$, where the dimensionality k and the exponent parameter $\alpha \in [0, 1]$ are given (Levy, Goldberg, and Dagan 2015).

Some static word embedding algorithms use **explicit** matrix factorization, including Latent Semantic Analysis (LSA). LSA attempts to directly truncate the SVD of the input signal matrix M to obtain its word embedding, and the signal matrix M is generated by utilizing co-occurrence statistics, such as the Pointwise Mutual Information (PMI) matrix, positive PMI matrix, and Shifted PPMI matrix (Levy and Goldberg 2014). In the other hand, there are also a wide range of word embedding algorithms using **implicit** matrix factorization which learns word embeddings by optimizing objective functions with gradient descent methods. For example, Word2Vec and GloVe perform implicit factorization on symmetric matrix (i.e., $\alpha = 0.5$). Moreover, the input signal matrix M is the shifted PPMI matrix for Word2Vec and the log-count matrix for GloVe.

Although intensive studies have been done to learn word embeddings using matrix factorization, selecting a proper dimensionality for word embedding algorithms is less investigated. Yin (Yin and Shen 2018) proposed a PIP-based dimensionality selection method. Specifically, for every possible dimensionality, its PIP loss is obtained by approximated computation, and then the dimensionality with the minimal PIP loss is selected. However, the approximation computation is very sensitive when exponent parameter α is small and results in a large approximation error. Such an error-prone PIP loss often leads to an improper dimensionality selected by the PIP-based dimensionality selection and a low-quality embedding trained by word embedding algorithms. In order to alleviate the approximation error for the PIP loss, in this paper, we propose Mixed Product Distance (MPD), where a scaling scheme is applied to the PIP loss to limit the approximation error.

Post-processing for Word Embedding

In order to further improve the quality of a word embedding trained by a static word embedding algorithm, post-processing which reinforces linguistic constraints on the vectors of the embedding, is often used. According to the studies on post-processing, (Mu and Viswanath 2018; Liu, Ungar, and Sedoc 2019), one of the important linguistic constraints is to diminish the influence of word frequency. Specifically, in a word embedding, principal components (PCs), mainly encoded by frequent words, are shared by all words. As a result, the vectors of less frequent words have a higher variance than those of frequent words, and such a high variance hurts the quality of the embedding because word frequencies are unrelated to lexical semantics. Moreover, two popular post-processing methods, the All-but-the-top (ABTT) (Mu and Viswanath 2018) and Conceptor Negation (CN) (Liu, Ungar, and Sedoc 2019), are often used to remove or suppress PCs in word embeddings. In order to use post-processing during dimensionality selection, a post-processing function is applied to the PIP loss in our MPD.

Methodology

Overview

Given a signal matrix M for a vocab of size n, a dimensionality selection is used to find the dimensionality k. Then, a static word embedding algorithm is chosen to train an embedding matrix $X \in \mathbb{R}^{n\times k}$, where X is composed of n vectors, and word w_i is represented by vector $v_i \in \mathbb{R}^k$, $i \in [n]$, and k is the dimension of a vector. In the following section, we focus on the explanation of our MPD-based dimensionality selection.

Formally, for a static word embedding algorithm with two parameters (the dimensionality k and the exponent parameter α), an **oracle embedding** X is derived for an ideal signal matrix M with the form $X = f_{\alpha,k_1}(M) \triangleq U_{\cdot,1:k_1}D^{\alpha}_{1:k_1,1:k_1}$, where $M = UDV^T$ is its SVD obtained by matrix factorization (Levy, Goldberg, and Dagan 2015). Notice, the ideal signal matrix M refers to the one without noise (e.g. PMI matrix). Nevertheless, in practice, only an empirical signal matrix $\tilde{M}$, which is estimated from training data, is available, where $\tilde{M} = M + Z$ is perturbed by noise Z. Through factorizing the noisy matrix $\tilde{M}$, a **trained embedding** $\hat{X} = f_{\alpha,k_2}(\tilde{M})$ is obtained. We aim at minimizing the distance between X and $\hat{X}$.

In order to measure the distance between the oracle embedding X and the training embedding $\hat{X}$, in this paper, a distance metric, MPD$(X, \hat{X})$, is defined, which is formulated as the geometric mean of two terms (primitive relative distance d_r and post-relative distance d_p), as shown in Eq.(1). In the following section, the two distances are presented, and a detailed algorithm implementation is given.

$$MPD(X, \hat{X}) = \sqrt{d_r(X, \hat{X}) \cdot d_p(X, \hat{X})} \qquad (1)$$

Primitive Relative Distance

In order to overcome the approximation error problem of the PIP loss (Yin and Shen 2018), we define the primitive relative distance d_r (see Definition 1), which applies a scaling scheme to the PIP loss. Moreover, unless specifically stated, the matrix norms used in the paper are **Frobenius** norms.

Definition 1. *The primitive relative distance d_r between the oracle embedding $X \in \mathbb{R}^{n\times k_1}$ and the training embedding $\hat{X} \in \mathbb{R}^{n\times k_2}$ is defined as follows:*

$$d_r\left(X, \hat{X}\right) = \frac{k_1 k_2}{k_1^2 + k_2^2} \frac{\left\|XX^T - \hat{X}\hat{X}^T\right\|^2}{\|XX^T\| \left\|\hat{X}\hat{X}^T\right\|} \qquad (2)$$

In Eq.(2), a constant term $\frac{k_1 k_2}{(k_1^2+k_2^2)}$ is used [1] to scale the value of d_r into $[0, 1]$, the numerator $||XX^T - \hat{X}\hat{X}^T||^2$ is the PIP loss which measures the vectors' relative position shifts between X and $\hat{X}$, and the denominator $||XX^T||||\hat{X}\hat{X}^T||$ is a scaling term.

Discussion In order to quantify the quality of a word embedding, we show two perspectives of the primitive relative distance: the approximation error of d_r, and the relationship between d_r and the unitary-invariance of the trained embedding $\hat{X}$ and the oracle embedding X (see Definition 2). Moreover, due to the limited space, the proofs of Theorem 1 and Theorem 2 are given in supplementary materials.

- The approximation error: according to Theorem 1, the value of d_r is limited in $[0, 1]$, and the limitation results in a smaller approximation error for d_r than the PIP loss.
- The relationship between d_r and the unitary-invariance: according to Theorem 2, if d_r is close to 0, the two embeddings (X and $\hat{X}$) are unitary-invariant. In other words, if d_r is close to 0, the trained embedding $\hat{X}$ has the same property as the oracle embedding X (Smith et al. 2017; Artetxe, Labaka, and Agirre 2016), such as word analogy and word relatedness (Baroni, Dinu, and Kruszewski 2014; Schnabel et al. 2015). Therefore, the optimal dimensionality for $\hat{X}$ is the one which miminizes d_r.

Definition 2. *Unitary-invariance of word embeddings refers that two embeddings are essentially identical if one can be transformed from the other by performing a unitary operation (e.g., a simple rotation). A unitary operation on a vector v in an embedding corresponds to multiplying the vector by a unitary matrix T, i.e., $v' = vT$, where $T^T T = TT^T = Id$. Such a unitary operation maintains the relative geometry of the vector (and the embedding). Therefore, unitary-invariance also defines the equivalence class of word embeddings.*

Theorem 1. *Under Arora's spatial isotropy assumption (Arora et al. 2016), d_r owns an asymptotic bound $[0, 1]$.*

Theorem 2. *Suppose $\|d_r(A, B)\| \approx 0$, then A and B are unitary-invariant (i.e., existing an unitary matrix T, $B \approx AT$).*

Post Relative Distance

In order to overcome the post-processing adoption problem of the PIP loss, the post-relative distance d_p is defined, as shown in Definition 3. Notice, the post-processing function in d_p can be any post-processing method used in word embedding generation because post-processing can be considered as a matrix transformation function.

Definition 3. *Given a post-processing function F, the post-relative distance d_p between the oracle embedding $X \in \mathbb{R}^{n\times k_1}$ and the trained embedding $\hat{X} \in \mathbb{R}^{n\times k_2}$ is defined as follows:*

$$d_r\left(X, \hat{X}\right) = \frac{k_1 k_2}{k_1^2 + k_2^2} \frac{\left\|XX^T - \hat{Y}\hat{Y}^T\right\|^2}{\|XX^T\| \left\|\hat{Y}\hat{Y}^T\right\|} \qquad (3)$$

where $\hat{Y} = F(\hat{X})$.

Discussion The post-relative distance is essentially the same as the primitive relative distance except that the post-processing function F is applied to the trained embedding $\hat{X}$. Similar to the discussion of d_r, the value of d_p is limited in $[0, 1]$, and possesses unitary-invariance.

Moreover, post-processing can effectively eliminate the principal components contained in a word embedding, and however, it also inevitably eliminates some crucial information in the word embedding. In order to reduce the excessive impact of post-processing to a word embedding, we choose to combine d_r and d_p with a geometric mean (see Eq. 1).

Algorithm Implementation

This part summarizes the whole procedure of the MPD dimension selection in a practical scenario. Formally, for a corpus that is used for word embedding, we first divide it into two eqaul parts, namely C_1 and C_2. Based on C_1 and C_2, we obtain the co-occurrence matrices A_1 and A_2, and then obtain the signal matrices M_1 and M_2, respectively. Next, the oracle embedding X_1 is constructed on M_1, the trained embedding X_2 is constructed on M_2, and a post-processing F can launch on X_2 to obtain $\hat{X}$. Then, the numerator in d_r can be computed through approximation

[1] In practical scenes, $k_1 = k_2$, we distinguish them for without generality.

in Yin (Yin and Shen 2018), while the denominator can be directly computed. Finally, the MPD can be calculated by $\sqrt{d_r(X_1, X_2) \cdot d_r(X_1, \hat{X})}$. Moreover, the grid search to find the dimension k that minimizes MPD loss can be done without evaluating on specific NLP tasks.

Experiment

In this section, we carry out four sets of experiments to comprehensively analyze our MPD-based dimensionality selection. The first two sets of experiments compare the effectiveness of the PIP-based dimensionality selection and our MPD-based dimensionality selection on two word intrinsic functionality tests and six downstream tasks, respectively. The third set of experiments compare the efficiency-performance trade-off among different dimensionality selection methods (PIP-based, PCA-based and MPD-based) and grid search. The fourth set of experiments examine the approximation errors of the PIP loss and the MPD. Moreover, according to the results in the third set of experiments, the PCA-based dimensionality selection and the PIP-based dimensionality selection have similar performances. Thus, the effectiveness comparison in the first two sets of experiments focuses only on the PIP-based selection and the MPD-based selection.

Specifically, we choose three static word embedding algorithms (LSA, Word2Vec[2] and GloVe) and the two commonly-used post-processing methods (CN and ABTT). For each experiment, a **theoretically optimal dimensionality** k^* and an **empirically optimal dimensionality** k^+ are selected as follows.

- Theoretically optimal dimensionality k^*: it is the dimensionality that minimizes the focused metric (the MPD or the PIP loss). Given the dimensionality selection method (e.g. MPD-based), for every possible dimensionality k, its metric value (e.g., MPD) is estimated. Then, the one with the minimal metric value is selected as the theoretically optimal dimensionality k^*.
- Empirically optimal dimensionality k^+ (Optimal): it is the dimensionality that achieves the highest performance on the test dataset using grid search. Given a word embedding algorithm, word embeddings are trained from dimensionality 50 to 1000 with an increment step of 1. Then, each word embedding is evaluated on the test dataset, and the one achieving the highest performance is selected as the empirically optimal dimensionality k^+.

Moreover, regarding to the estimation of the PIP loss and the MPD, the approximation computation used by Yin (Yin and Shen 2018) is applied in our experiments. To make fair comparison to the PIP loss, Text8 corpus[3] is used to generate the empirical signal matrix $\tilde{M}$ for Word2Vec and GloVe.

Word Intrinsic Functionality Tests

Experiment Settings In order to evaluate word intrinsic functionality, we conduct experiments of the word relatedness test, one of the two word intrinsic functionality tests done by Yin (Yin and Shen 2018). The quality of each embedding is evaluated on two test datasets, WordSim353 (Finkelstein et al. 2001) and MTurk777 (Halawi et al. 2012), and the performance is measured by the correlation between vector cosine similarity and human labels.

Data	WS353			MT771		
α	MPD	PIP	Optimal	MPD	PIP	Optimal
0	0.70	0.67	0.71	0.54	0.52	0.55
0.25	0.68	0.65	0.70	0.52	0.52	0.55
0.5	0.67	0.65	0.70	0.54	0.51	0.56
0.75	0.67	0.64	0.69	0.55	0.55	0.57
1	0.65	0.63	0.67	0.54	0.52	0.55

Table 1: The word relatedness performance using LSA. Column 'MPD' and 'PIP' represent the theoretically optimal dimensionalities k^*'s selected by the MPD-based and PIP-based dimensionality selection, respectively; Column 'Optimal' represents the empirically optimal dimensionalities k^+'s; Column "WS353" and "MT771" represent the two test datasets, WS353 and MT771, respectively.

Data	WS353			MT771		
	MPD	PIP	Optimal	MPD	PIP	Optimal
GloVe	0.66	0.63	0.69	0.56	0.53	0.58
Wd2Vec	0.65	0.63	0.68	0.55	0.53	0.58

Table 2: The word relatedness performance using GloVe and Word2Vec. 'Wd2Vec' represents Word2Vec. Note that α in GloVe and Word2Vec is 0.5 due to their symmetric properties.

Results The performances of selected dimensionalities are listed in Table 1 and 2, respectively. Notice, for LSA, α can be set to different values, and for Word2Vec and GloVe, α is 0.5 due to their symmetric property.

As shown in Table 1 and 2, our MPD-based dimensionality selection performs better than the PIP-based dimensionality selection in the word relatedness tests. In Table 1, the theoretically optimal dimensionality based on MPD (i.e., MPD k^*'s) achieves better results than PIP and is very close to the empirically optimal dimensionality k^+'s (i.e., optimal k^+'s) on both benchmarks, except two conditions: $\alpha = 0.25, 0.75$ on MT771, MPD performs the same as PIP. Similarly, in Table 2, MPD k^*'s performance also gets closer to optimal k^+'s and better than PIP k^*'s. Generally, MPD outperforms PIP in the intrinsic evaluation due to the incorporation of post-processing, which enables the neglect of the redundant components in the word embedding to better capture word semantics.

Downstream Task Evaluation

Experiment Settings Because it is often a case that the success on the word intrinsic functionality tests cannot be

[2] We choose the CBOW version.

[3] http://mattmahoney.net/dc/text.html

Embedding	Word2Vec							GloVe						
	—	ABTT			CN			—	ABTT			CN		
	PIP	PIP	MPD	Opti	PIP	MPD	Opti	PIP	PIP	MPD	Opti	PIP	MPD	Opti
MR	63.6	68.8	**70.6**	71.0	69.6	**70.9**	71.4	65.0	69.2	**71.3**	72.0	70.1	**72.0**	72.5
SST-2	80.0	80.5	**81.6**	81.9	80.2	**81.6**	81.9	88.5	88.7	**90.0**	90.2	88.4	**89.8**	90.2
SST-5	32.4	36.8	**38.0**	38.9	35.6	**38.2**	38.7	34.4	38.0	**39.4**	39.6	38.0	**39.2**	39.7
CoLA	4.3	4.7	**5.6**	5.7	4.8	**5.5**	5.6	14.4	14.9	**15.6**	15.7	14.5	**15.6**	15.8
WOZ	72.5	75.9	**83.9**	85.1	78.1	**83.4**	85.6	74.6	80.0	**86.5**	89.5	81.2	**90.2**	92.4
MRPC	79.4	79.7	**80.6**	80.8	80.2	**80.9**	80.9	80.0	80.7	**81.2**	81.4	80.1	**81.3**	81.5
QQP	53.9	55.8	**56.1**	56.4	55.6	**56.2**	56.6	60.4	61.0	**62.4**	62.6	61.0	**62.7**	62.9
STS-B	56.6	57.8	**58.6**	58.7	56.6	**58.8**	59.0	58.0	59.2	**60.0**	60.2	59.4	**60.2**	60.4
MNLI	54.4	55.8	**57.0**	57.4	55.6	**57.7**	58.0	67.9	69.0	**71.5**	71.6	69.0	**71.4**	71.6

Table 3: The accuracy performance of downstream NLP tasks using Word2Vec. Rows (e.g., 'MR', 'SST-5' and 'WOZ') represent the used datasets; Column '—', 'ABTT' and 'CN' represent the used post-processing methods, where '—' refers to the case that no post-processing is used; Column 'PIP', 'MPD' and 'Opti' represent PIP k^*, MPD k^* and the empirically optimal dimensions k^+, respectively.

well transferred to downstream NLP tasks (Schnabel et al. 2015), and the performances on downstream tasks are more crucial in real-world scenarios, we conducted experiments that directly compare the effectiveness of word embeddings on six downstream NLP tasks, including language inference, text classification, linguistic acceptability, sentence paraphrase, semantic textual similarity, and dialogue state tracking which recommends restaurants according to the constraints provided by users (Liu, Ungar, and Sedoc 2019). Moreover, the tasks can be divided into the following two kinds:

- **Single-sentence tasks**: (1) Text classification: the movie review (MR) dataset (Pang and Lee 2005); SST-5 (Socher et al. 2013) and SST-2 dataset (Socher et al. 2013); (2) Dialogue state tracking: Wizard-of-Oz383 (WOZ) dataset (Wen et al. 2017); (3) Linguistic acceptability: CoLA (Wang et al. 2018) dataset
- **Sentence-pair tasks**: (1) Sentence paraphrase: MRPC (Wang et al. 2018) and QQP (Wang et al. 2018) dataset; (2) Semantic textual similarity: STS-B dataset (Wang et al. 2018); (3) Language inference: MNLI dataset (Wang et al. 2018)

Specifically, for the two single-sentence tasks (text classification and linguistic acceptability), we encode the sentence with static word embedding and pass the resulting vector to a Bi-LSTM classifier. For another single-sentence task, dialogue state tracking, a deep-neural-network-based model, Neural Belief Tracker4 (NBT) (Mrksic and Vulic 2018) is chosen. For the three sentence-pair tasks, we encode sentences independently to produce vectors u, v, and pass $[u; v; |u-v|; u \cdot v]$ to a classifier for sentence-pair tasks. The classifier is an MLP with a 512-dimension hidden layer.

In each experiment, a word embedding is generated, and then a downstream NLP model (e.g., a Bi-LSTM classifier) is built on the embedding and the dataset (e.g., SST-2). Different word embedding generation frameworks are used, where either Word2Ve or GloVe is used during the static word embedding learning, and CN or ABTT is used as the post-processing. The performances on downstream tasks can reflect whether the selected dimension is proper.

Results The experimental results of the six downstream NLP tasks using Word2Vec and GloVe are reported in Table 3.

Firstly, in Table 3, the selection with post-processing (i.e., Column 'ABTT') significantly outperform the ones without post-processing (i.e., Column '—'). This shows the necessity of introducing post-processing to the word embedding generation process. Moreover, when post-processing is applied, MPD-based dimensionality selection consistently outperforms the PIP-based dimensionality selection, and achieves competitive performance with the optimal ones. For example, for NBT models (Row 'WOZ'), compared to PIP+Word2Vec+ABTT, the accuracy score based on MPD +Word2Vec+ABTT increases from 75.9% to 83.9%, which is closer to the optimal performance of 85.1%. This means that MPD-based dimensionality selection works effectively for downstream tasks.

Secondly, as illustrated in Table 3, the performances change as the used post-processing method changes, and the dimensions selected by MPD+CN generally achieves better performances than those using MPD+ABTT. For example, for NBT models, compared to MPD+GloVe+ABTT, the accuracy score based on MPD+GloVe+CN increases from 86.5% to 90.2%. The significant performance improvement indicates that MPD-based dimensionality selection can work better if an appropriate post-processing method F is chosen.

Finally, the performances on the downstream tasks also depend on the sensitivity of NLP tasks to dimension. In Table 3, the performance improvements of the NBT models used in dialogue state tracking are greater than those of Bi-LSTM models used in text classification. For example, compared to PIP+GloVe+CN, the performance of MPD+GloVe+CN rises by 9.1% on WOZ and increases only 0.9% on MR. Furthermore, the performances of Bi-LSTM models consistently improve on MR and SST-5, but the gains do not vary much. For example, compared to PIP+GloVe+CN, the performance of MPD+GloVe+CN rises by 1.3% and 2.6% on MR and SST-5, respectively.

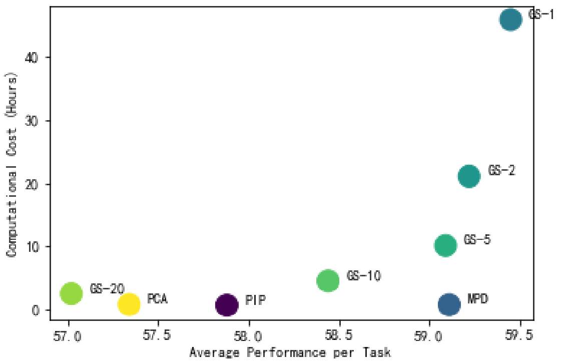
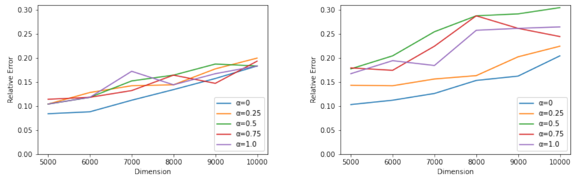


Figure 1: Comparison of efficiency-performance trade-off. Ideally, we hope a method could achieve a better performance and remain computationally efficient.

Efficiency-performance Trade-off

In this subsection, we compare efficiency-performance trade-off among grid search and three dimensionality selection methods (PIP-based, PCA-based, and MPD-based) through using Word2Vec on Text8, where efficiency is measured by computational cost. Specifically, for each selection method (or each grid search), the computation time of each run is recorded, and then the average of the computation time over five random runs is chosen as computational cost. Moreover, the efficiency-performance evaluation of grid search is dependent on grid granularity because a coarser grid search could save time at the cost of performance loss. Thus, in this experiment, we choose the dimensionality bound as $[50, 1000]$ and use five grid searches with five increments (including 1,2,5,10 and 20), respectively. The five grid searches are denoted as 'GS-1', 'GS-2','GS-5','GS-10' and 'GS-20', respectively. The efficiency-performance results are illuminated in Fig. 1.

In Fig. 1, among the five grid searches, 'GS-5'performs most closely to the MPD-based method but takes 14.01x time, where '14.01x' represents a 14.01x speedup if our MPD-based method is used. Moreover, 'GS-1' and 'GS-2' achieve slightly better performances but takes 59.44x and 27.23x computational cost, which is a disaster efficiency-performance trade-off. On the other hand, from Fig. 1, we can find that the PCA-based method and the PIP-based method have similar performances in both efficiency and performance. Furthermore, compared to the PCA-based method, the MPD-based method achieves 1.77% performance improvement and yet costs only 95% computation time of the PCA-based method. Compared to the PIP-based method, the MPD-based method achieves a 1.22% performance increase and 1.23x computational cost. Thus, the efficiency-performance trade-off of the MPD-based method is affordable.



(a) MPD

(b) PIP

Figure 2: An approximation error comparison between the PIP loss and the MPD.

Approximation Error

Since either the PIP loss or the MPD involves approximation computation, we demonstrate the approximation errors in Fig. 2. As shown in Fig. 2, the MPD achieves a smaller relative error, which indicates the MPD is more fit to the estimation by spectral analysis method (Yin and Shen 2018).

Ablation

To further explore the effects of the two distances in MPD (the primitive relative distance d_r and the post-relative distance d_p), we perform an ablation study, which compares three dimensionality selection methods based the following three metrics, respectively: (1) MPD: $\sqrt{d_r \cdot d_p}$ (2) Prim-D: d_r which applies a scaling scheme to the PIP loss (3) Post-D: d_p which is an adaptation of d_r using a post-processing function. The ablation study is conducted on word intrinsic functionality and downstream tasks as follows.

Word Intrinsic Functionality Tests The experimental settings of the word intrinsic functionality test are used to compare the performances of word embeddings based on the three dimensionality selection: MPD-based, Prim-D-based, and Post-D-based. Moreover, LSA is used during word embedding learning because α can be set to different values. Tables 4 and 5 illustrate the theoretical optimal dimensionality k^*'s and the empirically optimal dimensionality k^+'s for different metrics.

As shown in Table 4 and 5, Prim-D performs slightly worse than PIP with an average 0.01 drop, which indicates that the scaling scheme may not be suitable in some cases of intrinsic evaluation. On the other hand, Post-D outperforms Prim-D and achieves comparable performance to PIP, demonstrating the effectiveness of incorporation with post-processing. Besides, MPD obtain better results than all other three methods, validating the loss design of the MPD which combines d_r and d_p through a geometric mean. Generally, the results indicate that (1) the scaling scheme results in a

Data	WS353					MT771				
α	MPD	PIP	Prim-D	Post-D	Optimal	MPD	PIP	Prim-D	Post-D	Optimal
0	**0.70**	0.67	0.67	0.66	0.71	**0.54**	0.52	0.52	0.52	0.55
0.25	**0.68**	0.65	0.62	0.65	0.70	**0.52**	0.52	0.54	0.52	0.55
0.5	**0.67**	0.65	0.64	0.66	0.70	**0.54**	0.54	0.54	0.52	0.56
0.75	**0.67**	0.64	0.64	0.62	0.69	**0.55**	0.55	0.52	0.52	0.57
1	**0.65**	0.63	0.64	0.63	0.67	**0.54**	0.52	0.50	0.55	0.55

Table 4: The ablation study in the intrinsic evaluation using LSA. ‘MPD’, ‘PIP’ indicate the dimension selection criterions.

Data	WS353					MT771				
	MPD	PIP	Prim-D	Post-D	Optimal	MPD	PIP	Prim-D	Post-D	Optimal
GloVe	**0.66**	0.63	0.62	0.65	0.69	**0.56**	0.53	0.55	0.55	0.58
Word2Vec	**0.65**	0.63	0.60	0.64	0.68	**0.55**	0.53	0.54	0.55	0.58

Table 5: The ablation study in the word relatedness performance using GloVe and Word2Vec.

Post-P	ABTT					CN				
Metric	PIP	MPD	Prim-D	Post-D	Optimal	PIP	MPD	Prim-D	Post-D	Optimal
MR	69.2	**71.3**	70.7	70.3	72.0	70.1	**72.0**	70.5	70.2	72.5
SST-2	88.7	**90.0**	89.2	89.1	90.2	88.4	**89.8**	89.1	88.9	90.2
SST-5	38.2	**39.4**	38.5	38.7	39.6	38.0	**39.2**	38.4	38.6	39.7
CoLA	14.9	**15.6**	15.0	15.3	15.7	14.5	**15.6**	14.9	15.1	15.8
WOZ	82.9	**86.1**	84.5	84.8	89.8	82.5	**90.2**	88.2	88.0	92.4
MRPC	80.7	**81.2**	80.6	80.5	81.4	80.1	**81.3**	80.3	80.5	81.5
QQP	61.0	**62.4**	61.5	61.2	62.6	61.0	**62.7**	62.0	61.9	62.9
STS-B	59.2	**60.0**	59.5	59.6	60.2	59.4	**60.2**	59.6	59.7	60.4
MNLI	69.0	**71.5**	69.3	69.6	71.6	69.0	**71.4**	69.6	69.9	71.6

Table 6: Ablation analysis on downstream NLP tasks using GloVe.

smaller approximation error but slightly decreases the performance in intrinsic evaluation. (2) the incorporation of post-processing is effective in intrinsic evaluation.

Downstream Task Evaluation The experimental settings in the downstream task evaluation are used to compare the performance on the three dimensionality selection, and GloVe is used in experiments. The experimental results are reported in Table 6. In Table 6, the dimension based on MPD achieves the best performance, which confirms the MPD's superiority for downstream NLP tasks. Furthermore, MPD achieves very close performance to the optimal performance by grid search with increment 1. On the other hand, Prim-D and Post-D consistently outperform the PIP loss. These results validate the effectiveness of the scaling scheme and post-processing term used in MPD. Besides, the different performance on word relatedness tests and downstream NLP tasks indicates that it is necessary to examine the effectiveness of the dimensionality selection methods on different NLP tasks.

Conclusion

In this paper, we propose MPD, a novel metric that applies a scaling scheme in the primitive relative distance d_r to alleviate approximation error of the PIP loss and uses the post relative distance d_p to incorporate the post-processing computation. Then, we develop an MPD-based dimensionality selection method that automatically selects the optimal dimensionality for static embedding algorithms. Extensive experiments on both word intrinsic functionality tests and downstream tasks, and the results show the effectiveness and the efficiency-performance trade-off of our MPD-based dimension selection.

References

Arora, S.; Li, Y.; Liang, Y.; Ma, T.; and Risteski, A. 2016. A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4: 385–399.

Artetxe, M.; Labaka, G.; and Agirre, E. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2289–2294.

Baroni, M.; Dinu, G.; and Kruszewski, G. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 238–247.

Chung, Y.-A.; and Glass, J. 2018. Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech. *arXiv preprint arXiv:1803.08976*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Finkelstein, L.; Gabrilovich, E.; Matias, Y.; Rivlin, E.; Solan, Z.; Wolfman, G.; and Ruppin, E. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, 406–414.

Guo, J.; Tan, X.; He, D.; Qin, T.; Xu, L.; and Liu, T.-Y. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3723–3730.

Halawi, G.; Dror, G.; Gabrilovich, E.; and Koren, Y. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1406–1414.

Levy, O.; and Goldberg, Y. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, 2177–2185.

Levy, O.; Goldberg, Y.; and Dagan, I. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3: 211–225.

Liu, T.; Ungar, L.; and Sedoc, J. 2019. Unsupervised post-processing of word vectors via conceptor negation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6778–6785.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Mrksic, N.; and Vulic, I. 2018. Fully Statistical Neural Belief Tracking. In *ACL (2)*.

Mu, J.; and Viswanath, P. 2018. All-but-the-Top: Simple and Effective Postprocessing for Word Representations. In *International Conference on Learning Representations*.

Palangi, H.; Deng, L.; Shen, Y.; Gao, J.; He, X.; Chen, J.; Song, X.; and Ward, R. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4): 694–707.

Pang, B.; and Lee, L. 2005. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, 115–124.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Schnabel, T.; Labutov, I.; Mimno, D.; and Joachims, T. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, 298–307.

Smith, S. L.; Turban, D. H.; Hamblin, S.; and Hammerla, N. Y. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, 926–934. Citeseer.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 353–355.

Wang, Y. 2019. Single Training Dimension Selection for Word Embedding with PCA. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3597–3602.

Wen, T.-H.; Vandyke, D.; Mrkšić, N.; Gasic, M.; Barahona, L. M. R.; Su, P.-H.; Ultes, S.; and Young, S. 2017. A Network-based End-to-End Trainable Task-oriented Dialogue System. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 438–449.

Yin, Z.; and Shen, Y. 2018. On the dimensionality of word embedding. In *Advances in Neural Information Processing Systems*, 887–898.

Zhang, S.; Yao, L.; Sun, A.; and Tay, Y. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1): 1–38.