2024 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)

Efficient Feature Selection for Word Embedding Dimension Reduction

Jintang Xue, Yun-Cheng Wang, Chengwei Wei and C.-C. Jay Kuo
University of Southern California, Los Angeles, CA, USA

Abstract—**Word embeddings transform each word into a vector space representation, making it possible for computers to process and analyze the human language. However, storing and processing these word vectors can become resource-intensive as the vector space dimension increases and the vocabulary expands. It poses a challenge, especially for mobile edge-device applications. This paper explores dimension reduction for word embedding to balance computational costs and performance. We propose an efficient weakly-supervised feature selection method called WordFS, which has two variants utilizing novel criteria for feature selection. Our experiments on various tasks, such as word similarity and binary and multi-class classification, show that WordFS outperforms other dimension reduction methods while requiring lower computational costs.**

I. Introduction

In recent years, large language models (LLMs) have made a breakthrough in natural language processing (NLP). These models have revolutionized the understanding and generation of human language tasks such as question-answering, machine translation, and text summarization [1], [2], etc. As a fundamental component in language models, word embedding represents words as vectors in a continuous, high-dimensional space [3]. Word vectors capture semantic and syntactic meanings of words, providing word relationships for various downstream tasks. Static word embedding methods assign a fixed vector to each word. They convert input words to corresponding vectors for further processing and model training. Contextual word embedding methods leverage deep-learning models to generate word vectors based on the context. A word can have a different contextual word embedding in a different sentence, allowing the model to capture subtle differences in the meaning of the same word in various contexts.

A common challenge for static and contextual word embedding methods is that the high dimension of a word vector leads to an enormous model size. Typically, a word vector has hundreds to thousands of dimensions. For instance, loading a 300-dimensional word embedding matrix with 2.5 million tokens would require up to 6 GB of memory on a 64-bit system [4]. On the one hand, high-dimensional word vectors provide a good representation of complex human language, which is crucial for performing downstream tasks. On the other hand, high-dimensional word vectors have higher demands on computational resources and memory requirements. Thus, dimension reduction is critical in the application of word vectors.

Existing dimension reduction methods mainly consist of traditional PCA-based and deep-learning-based models. Traditional unsupervised PCA-based methods are known for their efficiency and interpretability. In contrast, supervised deep-learning-based methods are inefficient and lack interpretability. Semi-supervised feature selection methods can also be used for dimension reduction. Since word vectors can be viewed as extracted features for each word, feature selection methods have the potential to provide a straightforward yet effective way to reduce the word dimension. It could help balance unsupervised PCA-based methods and resource-intensive deep-learning-based methods.

This paper investigates using a small subset of labeled word similarity pairs to develop a weakly supervised dimension reduction method called WordFS (i.e., word dimension reduction with **F**eature **S**election). We demonstrate that one can achieve dimension reduction effectively by supervising a limited number of word similarity pairs. Note that word similarity is typically used in evaluation tasks but is rare in word embedding dimension reduction, a key novelty of this work. The proposed WordFS method consists of three stages: 1) post-processing, 2) feature extraction, and 3) weakly-supervised feature selection. We apply WordFS to other downstream tasks to show its generalizability. Experimental results show that WordFS outperforms existing methods in word similarity and various tasks while achieving much lower computational costs.

This work has the following significant contributions.

- We propose a novel, effective, and efficient dimension reduction method called WordFS for word embeddings from the perspective of feature selection based on weakly-supervised learning.
- We demonstrate the potential of combining feature selection methods and word similarity for word embedding dimension reduction.
- We show the effectiveness and efficiency of our approach on various downstream tasks, including binary and multi-class classification tasks. Our method generally outperforms the existing techniques while being more straightforward.

The rest of the paper is organized as follows. Related work is reviewed in Sec. II. The proposed WordFS method is described in Sec. III. Experimental results are shown in Sec. IV. Finally, concluding remarks and future extensions are given in Sec. V.

II. Related work

Word embedding compression is an essential topic for storing and processing word embeddings, especially on computationally limited devices. Existing dimension reduction methods mainly consist of traditional and deep learning-based models.

A. Matrix Decomposition Techniques

Matrix decomposition techniques, such as singular value decomposition (SVD), principal components analysis (PCA), non-negative matrix factorization (NMF), and factor analysis (FA), have been applied to dimension reduction of word embeddings. While simple and efficient, these methods are not highly effective.

Post-processing methods help enhance word embedding in dimension reduction. One can achieve lower dimensional word embeddings by combining post-processing algorithms (PPAs) to reduce the anisotropy [5] with PCA [4]. The dimension-reduced vectors can perform similarly or even better than the original pre-trained word embeddings, outperforming most unsupervised methods.

B. Deep Learning Methods

Deep learning-based models have recently become quite popular, and efforts have been made to explore their potential in reducing word embedding dimensions. EmbedTextNet [6] achieves better performance in low-dimensional embedding sizes by utilizing a VAE model with a correlation penalty added to the weighted reconstruction loss. However, deep learning models typically require more computational resources and longer training time, contradicting the original goal of dimension reduction.

C. Feature Selection Methods

Feature selection is another simple and intuitive method for removing redundant features. Feature selection methods can be applied to bag-of-words representations, where each unique word is treated as a distinct feature, for emotion recognition [7] and sentiment classification [8] tasks to reduce the number of terms and save storage and memory space. Also, feature selection methods can be applied for text classification to select the most representative word [9]. However, the role of feature selection for word embedding dimension reduction is under-explored. This paper shows that a straightforward and intuitive feature selection method can effectively reduce dimension. We believe applying the feature selection method for word embedding dimension reduction is novel.

III. Proposed Method

A. System Overview

Our WordFS method consists of three stages: 1) post-processing, 2) feature extraction, and 3) feature selection. We adopt word similarity datasets as the source of supervision to guide the feature selection in our model. As a widely-used benchmark of word embeddings, the word similarity task evaluates the quality of word embeddings in representing semantic and syntactic meanings by measuring how closely the representation of word vectors matches human perception of similarity [10], making them good supervision resources for word embedding dimension reduction. Also, since word similarity captures fundamental characteristics of word embeddings, which are crucial for many NLP tasks, its potential for generalizing to downstream tasks is noteworthy.

In our method, We begin by optionally applying a post-processing method to our pre-trained word embeddings since the anisotropy may not always be harmful [11]. We then extract pair-wise features for each word pair by conducting element-wise production with normalization. Then, we adopt two different criteria for feature selection to evaluate each dimension and identify essential dimensions. The first one is based on a supervised feature selection method called RFT [12]. The second is based on Spearman's rank correlation coefficient between each feature dimension and word similarity scores. Finally, the selected dimensions of the word embeddings are kept, and other dimensions are discarded according to the requirements for dimension reduction. The details of each stage are elaborated below.

B. Post-processing

To improve the quality of word embedding in downstream applications, performing post-processing techniques on pre-trained word embeddings can be crucial. The post-processing algorithm (PPA) [5] is an effective post-processing method to improve the isotropy of word representations by removing the top principal components of all words.

The existing effective PCA-based method [4] applies the PPA before and after dimension reduction to enhance both the original and the dimension-reduced word vectors. In our method, we only apply PPA before dimension reduction, which makes our method simpler. Applying PCA-based post-processing to the selected subset after feature selection may disrupt the previous results because PCA and the feature selection criteria can have different objectives. Specifically, PCA transforms features to maximize variance, while feature selection methods usually select features based on their correlation with labels. Also, the selected feature subset may lose crucial information for PCA to find valuable principal components. As an enhancement of original word embeddings before dimension reduction, we also make PPA an optional choice. The reason is that anisotropy may not always be harmful [11] and using PPA depends on the pre-trained word embeddings and specific application scenarios. Specifically, word embeddings trained on less data may be vulnerable to noise. Applying PPA helps mitigate some of the noise and offers better performance. On the other hand, using PPA may result in the loss of crucial and intricate information related to word representation, which can harm performance. Word embeddings trained on relatively small data benefit from uniformly distributed word embeddings, while complex contextual tasks may require anisotropy to capture detailed information.

C. Feature Extraction

In this stage, we extract suitable features for feature selection based on the dataset that provides supervision. In this paper, we leverage word similarity data as the weak supervision of our model. Since the word similarity is based on the correlation (i.e., word similarity score) between every two words, we extract pair-wise features for each word similarity pair.

The pair-wise features are constructed in each dimension based on cosine similarity, an effective measure for evaluating the similarity between word embeddings. Cosine similarity between two-word vectors is derived by dividing their dot product by the product of their magnitudes:

$$\begin{aligned}\text{Sim}(a, b) &= \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|} \\ &= \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \cdot \sqrt{\sum_{i=1}^{n} b_i^2}},\end{aligned} \tag{1}$$

where $\mathbf{a}$ and $\mathbf{b}$ are embeddings of the two words. The cosine similarity score can be viewed as the sum of the normalized results of element-wise products. Each pair of elements (i.e., each dimension) has a linear impact on the cosine similarity score of that pair of word vectors. Therefore, the normalized results of element-wise products can be viewed as features to predict the similarity score. The feature we use for each dimension is:

$$f_i = \frac{a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \cdot \sqrt{\sum_{i=1}^{n} b_i^2}}, \tag{2}$$

where f_i is the feature of the i^{th} dimension extracted from the two word embeddings.

D. Feature Selection

The method for the feature selection process depends on the objective of downstream tasks. Specifically, we employ two feature selection methods for prediction and similarity tasks. These two feature selection methods are developed into our model's variants, WordFS-S and WordFS-P. We focus on filter feature selection techniques since they are simple and computationally efficient. Although different dimensions of distributed word embeddings may capture nuanced and interdependent semantic relationships, they allow us to identify dimensions that contribute most significantly to model performance. Besides, filter feature selection techniques select features based on their correlation with the labels and are independent of the classifier. As a result, the selected features are generalizable.

Prediction tasks: For prediction tasks, we leverage the methods from Discriminant Feature Test (DFT) and Relevant Feature Test (RFT) [12] to build our WordFS-P model. DFT and RFT are a pair of filter feature selection techniques proposed recently for classification and regression tasks, respectively. They have been widely used in green learning architectures [13] for feature dimension reduction to reach a smaller model size. In our experiments, RFT is utilized for feature selection, and we adopt the word similarity scores as the labels. RFT partitions each feature dimension into two subintervals and calculates the overall mean square error (MSE) as the loss function. A smaller loss function means a better feature dimension. Specifically, given a feature of the i^{th} dimension f_i, the feature space is partitioned into $B = 2^k, k = 1, 2, ...$ uniform segments and the optimal partition threshold is searched among $B - 1$ candidates in the range $[min(f_i), max(f_i)]$:

$$f_b^i = min(f_i) + \frac{b}{B}\left[max(f_i) - min(f_i)\right], \tag{3}$$

where $b = 1, 2, ...B - 1$. The threshold t partitions the i^{th} feature space into the left subset $S_{L,t}^i$ and the right subset $S_{R,t}^i$. The MSEs for regression $R_{L,t}^i$ and $R_{R,t}^i$ are calculated separately in the two subsets. Then, the RFT loss is defined as the weighted sum of the two MSEs:

$$R_t^i = \frac{N_{L,t}^i R_{L,t}^i + N_{R,t}^i R_{R,t}^i}{N}, \tag{4}$$

where $N_{L,t}^i$ and $N_{R,t}^i$ denote the number of samples in each subset and N is the total number of samples. The RFT loss of each feature dimension is defined as the minimal RFT loss among all the candidate thresholds:

$$R_i = \min_{t \in T} R_t^i. \tag{5}$$

Finally, all feature dimensions' RFT loss is ranked in ascending order, and the top K dimensions are selected, where K is the dimension of the word vectors after our dimension reduction approach.

Similarity tasks: Similarity tasks are usually done by calculating the cosine similarity between target vectors, which differs from prediction tasks. The cost function of DFT/RFT may not match well with the evaluation criteria of similarity tasks. Inspired by the evaluation of word similarity, we use Spearman's rank correlation coefficient to develop our WordFS-S model. It involves finding the correlation between the ranks generated by features extracted from each dimension and the target labels. First, the ranked values of each feature dimension and the labels are calculated. Second, Spearman's rank correlation coefficient for each feature dimension is calculated by the Pearson correlation coefficient between the ranked values of each dimension and the labels:

$$r_i = \rho_{R(f_i), R(y)} = \frac{\text{cov}(R(f_i), R(y))}{\sigma_{R(f_i)} \sigma_{R(y)}}. \tag{6}$$

The higher the correlation coefficient, the better the feature dimension is. Finally, the Spearman's rank correlation coefficients of all the feature dimensions are then ranked in descending order, and the top K dimensions are selected.

IV. EXPERIMENTS

We evaluate our method by applying it to pre-trained word embeddings. The word similarity datasets provide weak supervision to guide our feature selection module. The dimension-reduced word embeddings are used for word similarity tasks

TABLE I
Comparison of Spearman's rank correlation coefficient across multiple word similarity datasets, where **boldface** indicates the best value in each column, and underline indicates the second best value in each column.

Dataset	MC-30	Men-TR-3k	M Turk-287	M Turk-771	RG-65	RW Stanford	Sim Lex-999	VE RB-143	WS-353-ALL	WS-353-REL	WS-353-SIM	YP-130	Avg
Glove-300D [14]	64.03	73.71	62.44	64.60	72.97	41.16	36.97	29.97	60.27	55.44	64.74	55.15	56.79
Algo-150D [4]	67.97	75.21	62.47	64.11	72.94	43.45	37.81	**38.06**	66.71	60.55	70.74	**58.09**	59.84
EmbedTextNet-150D [6]	**79.59**	75.43	58.36	62.91	**80.00**	41.31	37.49	27.33	63.06	55.35	67.50	55.80	58.68
WordFS-P-woP-150D (ours)	66.79	74.94	60.68	63.44	69.76	36.16	37.93	30.30	62.49	53.69	66.78	52.27	56.27
WordFS-P-wP-150D (ours)	72.94	75.38	64.38	64.24	70.46	45.01	42.47	34.88	**68.53**	**60.58**	70.25	50.67	59.98
WordFS-S-woP-150D (ours)	67.79	75.78	62.62	62.95	70.26	36.07	44.72	36.61	64.72	55.36	69.42	45.76	57.67
WordFS-S-wP-150D (ours)	72.27	**75.96**	**65.57**	**64.35**	70.69	**45.22**	**45.08**	36.03	67.16	59.18	**71.00**	52.90	**60.45**

TABLE II
Performance comparison of Spearman's rank correlation coefficients on the aggregated word similarity dataset.

Dimension	300	150	100	50
Glove [14]	45.74	–	38.45	36.24
Algo [4]	–	53.48	49.35	42.61
EmbedTextNet [6]	–	52.71	43.88	41.00
WordFS-P-woP (ours)	–	48.23	48.39	48.22
WordFS-P-wP (ours)	–	54.69	52.60	47.29
WordFS-S-woP (ours)	–	55.85	54.45	49.32
WordFS-S-wP (ours)	–	**56.76**	**54.73**	**50.11**

and downstream tasks. Then, we compare the results of our method with the original pre-trained word embeddings, the dimension-reduced word embeddings from the PCA-based method called Algo [4], and the results from the deep learning-based method called EmbedTextNet [6].

A. Pre-trained Word Embeddings

In our experiments, we use the Glove word embeddings [14] trained on Wikipedia 2014 and Gigaword 5 corpus (6B tokens, 400K vocabulary). They are available in multiple dimensions, precisely 50, 100, 200, and 300.

B. Word Similarity Datasets

We use the widely-used word similarity datasets [15] to evaluate our method. The word similarity datasets contain word pairs and their corresponding scores from human annotators based on perceived relatedness or similarity. The cosine similarity of each pair of words calculates the similarity score from the word embeddings. We use Spearman's rank correlation coefficient as our evaluation metric. It measures how closely the ranking derived from the cosine similarities of given word vectors matches those based on human judgments. A higher value of this metric indicates a better match to human-labeled similarity rankings.

We first evaluate our methods on twelve word similarity datasets. Since we choose word similarity as the guidance for feature selection, we apply 5-fold cross-validation to each word similarity dataset to train and test our method. The reported results are the average Spearman's rank correlation coefficients from the five folds. We evaluate the pre-trained word embeddings and all the methods on the same cross-validation fold and take the average for a fair comparison. All the results we report average over five different cross-validation trials in all the experiments. We set the number of bins in RFT to 4, the default and recommended value from the RFT model, and the threshold in the PPA to 7, the same as in the PCA-based model.

Table I shows the results across twelve word similarity datasets of different dimension reduction methods reduced from 300-dimensional Glove word embeddings to 150 dimensions. In the table, WordFS-P and WordFS-S represent our proposed methods based on RFT and similarity feature selection methods, respectively. P represents PPA, and wP and woP represent PPA and without PPA. The last column is the average of the previous columns. For the dimension-reduced vectors, our methods outperform the original word embeddings and other methods in most datasets. Our WordFS-S-wP method achieves the highest average score. Both feature selection-based approaches perform better than the existing methods. As expected, the WordFS-S-wP method performs better than the WordFS-P-wP method in this task. Our WordFS-S-wP method outperforms the best performance from existing methods with an average improvement of 0.61 in terms of Spearman's rank correlation coefficients when reducing to 150 dimensions. Our method can perform better than the original 300-dimensional word embeddings when reduced to 150 dimensions.

Then, we aggregated the twelve word similarity datasets. We first scaled them into the same range [0, 1] and then averaged the similarity scores of overlapped word pairs. There are two reasons to aggregate the datasets: 1) to provide an overall evaluation for the word similarity tasks and 2) to construct a comprehensive dataset to train our feature selection methods for various downstream tasks. We refer to our model as a weakly-supervised method for downstream tasks because there are only 7,705 human-labeled similarity scores for different word pairs in the aggregated dataset, which is much less than the total number of possible word pairs in the corpus to pre-train the word embeddings. For example, there are 400K word vectors in the pre-trained Glove word embeddings, which can generate approximately 80 billion unique word pairs. Compared with the possible word pairs, the guidance provided by the aggregated dataset is limited. We experimented

TABLE III
COMPARISON OF THE RESULTS OF DOWNSTREAM PREDICTION TASKS. THE LAST COLUMN IS THE AVERAGE OF THE PREVIOUS COLUMNS.

Task	MR	CR	MPQA	SUBJ	STS-B	SST -FG	TREC	MRPC	SICK-E	Avg
Glove-300D [14]	74.99	75.81	86.75	91.39	78.20	40.77	66.6	75.28	77.19	73.81
Algo-150D [4]	73.70	75.32	85.31	89.59	**76.66**	40.95	60.7	**71.68**	76.49	72.27
EmbedTextNet-150D [6]	72.67	74.09	84.95	**90.08**	73.75	40.50	64.8	71.54	74.43	71.87
WordFS-P-woP-150D (ours)	73.09	74.86	**86.24**	89.86	76.22	39.55	**65.4**	71.01	**76.62**	72.54
WordFS-P-wP-150D (ours)	**74.03**	**76.00**	**86.24**	89.20	76.50	**41.27**	65.2	70.38	76.46	**72.81**
Algo-100D [4]	70.61	72.82	82.71	88.16	74.63	38.28	56.2	**71.83**	75.36	70.07
EmbedTextNet-100D [6]	71.55	72.87	84.44	**88.88**	70.18	38.46	62.2	71.48	75.14	70.58
WordFS-P-woP-100D (ours)	72.24	73.96	85.26	88.62	73.97	38.37	62.0	71.42	**76.60**	71.38
WordFS-P-wP-100D (ours)	**72.68**	**75.42**	**85.50**	88.25	**76.94**	**40.50**	**64.4**	71.59	74.87	**72.24**
Algo-50D [4]	66.24	70.23	76.35	84.32	69.36	35.88	48.1	71.39	72.98	66.09
EmbedTextNet-50D [6]	69.36	**72.34**	82.79	**87.65**	71.66	**38.64**	**60.2**	68.29	**74.20**	**69.45**
WordFS-P-woP-50D (ours)	68.54	70.54	81.35	85.51	69.85	35.61	56.2	**71.65**	73.19	68.05
WordFS-P-wP-50D (ours)	**69.96**	71.05	**83.29**	84.25	**72.10**	37.69	56.6	71.30	72.82	68.78

TABLE IV
COMPLEXITY COMPARISON OF THE ALGO METHOD, THE EMBEDTEXTNET METHOD, AND OUR METHODS WHEN REDUCING THE WORD EMBEDDING DIMENSIONS FROM 300 TO 150 IN HARDWARE CONFIGURATION AND TRAINING TIME.

Method	Hardware Configuration	Training Time (s)
EmbedTextNet [6]	GPU	2149.04 (1535.03X)
Algo [4]	CPU	21.50 (53.75X)
WordFS-S-wP (ours)		17.63 (44.08X)
WordFS-P-wP (ours)		11.84 (29.60X)
WordFS-S-woP (ours)		4.05 (10.13X)
WordFS-P-woP (ours)		0.40 (1X)

with the aggregated dataset using the same settings as before.

Table II shows the results of various methods for dimension reduction on the aggregated dataset. Our WordFS-S methods perform best among all the experiments in different dimensions. And there is a large gap between our methods and the existing methods. Compared to other methods, our approaches significantly benefit this task, particularly in lower dimensions where the gap widens. It is worth noting that even when reduced to 50 dimensions, our WordFS-S-wP method achieves better performance compared to the original 300-dimensional word embeddings.

C. Downstream Tasks

It is insufficient to evaluate word embeddings only based on word similarity tasks. To demonstrate the power of our method, generalizability on downstream tasks is crucial. We utilize the SentEval toolkit [16] to evaluate the dimension-reduced vectors on various downstream tasks. We conducted experiments on nine prediction tasks, including binary and multi-class classification tasks (MR, CR, MPQA, SUBJ, STS-B, STS-FG, and TREC), paraphrase detection task (MRPC), and entailment and semantic relatedness task (SICK-E). These tasks directly take the word embeddings as input. We applied our WordFS-P method with the RFT feature selection based on weak guidance from the aggregated word similarity dataset to the pre-trained word embeddings in all the experiments to reduce the dimension. Then, we directly input the dimension-reduced embeddings to the downstream tasks and compare the test accuracy of different methods for each task.

Table III shows the results of various methods on the nine prediction tasks. Our experiments show that our method outperforms the PCA-based method on most prediction tasks. Also, our method achieves higher average scores than the PCA-based method in all the settings. Our method performs better than the EmbedTextNet model, although it performs slightly worse when reduced to 50 dimensions. However, as shown in the next section, our model is much more efficient and takes less time to train than the EmbedTextNet model. Our method retains more helpful information for most settings while reducing dimensions, achieving a much closer prediction performance to the original 300-dimensional word embeddings. It indicates the effectiveness of our method. Also, it proves that our method can generalize well on various prediction downstream tasks in a zero-shot manner.

D. Model Efficiency

We compare the model efficiency of the Algo method, the EmbedTextNet method, and our method when reducing the word embedding dimensions from 300 to 150 in Table IV. Different hardware is used for other methods because of their algorithmic nature. All the experiments were conducted on the same server, equipped with two AMD EPYC 7543 32-core CPUs and seven NVIDIA RTX A6000 GPUs. The deep learning-based model is trained using the GPUs, while the Algo model and our methods are trained using only the CPUs. The EmbedTextNet model takes approximately 30 minutes to train on the GloVe word embeddings. In contrast, the training time for our WordFS-P-wP and WordFS-S-wP models is less than 19 seconds and 13 seconds, respectively. Our methods without post-processing require even less time — the training of the WordFS-P-woP model is completed within one second, making it thousands of times faster than the deep learning-based method.

Additionally, our methods are more efficient than the PCA-based method because they employ PPA before and after the PCA, while we only apply it once before feature selection. Our model, without the post-processing, can achieve even less time. The reason is the post-processing and PCA must be done on the entire vocabulary. However, our feature selection method only focuses on a subset of words that appear in the aggregated word similarity dataset, making the procedure very fast. In conclusion, Our methods take only a fraction of the time compared to deep learning methods and less than half the time compared to existing PCA-based methods. The experimental results demonstrate that our methods are much more efficient than existing methods.

V. Conclusion and Future Work

This paper proposes a general dimension reduction method called WordFS for pre-trained word embeddings with the post-processing algorithm (PPA) and simple yet effective feature selection methods based on weak supervision provided by a limited number of word similarity pairs. Our method is more straightforward, efficient, intuitive, and effective in most cases than the existing methods. Empirical results show that our method outperforms existing methods in word similarity tasks and generalizes well to various downstream tasks with much lower computational costs. In the future, we would like to explore the compression of word embeddings further. Since feature selection methods may result in information loss, there is still room for improvement in the performance of prediction tasks between the original word embeddings and the reduced word vectors. Additionally, suitable feature selection methods can be developed to compress pre-trained word vectors in specific domains.

References

[1] J. Xue, Y.-C. Wang, C. Wei, X. Liu, J. Woo, and C.-C. J. Kuo, "Bias and fairness in chatbots: An overview," *APSIPA Transactions on Signal and Information Processing*, vol. 13, no. 2, 2024. DOI: 10.1561/116.00000064. [Online]. Available: http://dx.doi.org/10.1561/116.00000064.

[2] H. Naveed, A. U. Khan, S. Qiu, *et al.*, "A comprehensive overview of large language models," *arXiv preprint arXiv:2307.06435*, 2023.

[3] C. Wei, Y.-C. Wang, B. Wang, and C.-C. J. Kuo, "An overview of language models: Recent developments and outlook," *APSIPA Transactions on Signal and Information Processing*, vol. 13, no. 2, 2024. DOI: 10.1561/116.00000010. [Online]. Available: http://dx.doi.org/10.1561/116.00000010.

[4] V. Raunak, V. Gupta, and F. Metze, "Effective dimensionality reduction for word embeddings," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, 2019, pp. 235–243.

[5] J. Mu and P. Viswanath, "All-but-the-top: Simple and effective postprocessing for word representations," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=HkuGJ3kCb.

[6] D. Y. Hwang, B. Taha, and Y. Nechaev, "Embedtextnet: Dimension reduction with weighted reconstruction and correlation losses for efficient text embedding," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 9863–9879.

[7] Z. Erenel, O. R. Adegboye, and H. Kusetogullari, "A new feature selection scheme for emotion recognition from text," *Applied Sciences*, vol. 10, no. 15, p. 5351, 2020.

[8] A. K. Uysal and Y. L. Murphey, "Sentiment classification: Feature selection based approaches versus deep learning," in *2017 IEEE International Conference on Computer and Information Technology (CIT)*, IEEE, 2017, pp. 23–30.

[9] W. Rui, J. Liu, and Y. Jia, "Unsupervised feature selection for text classification via word embedding," in *2016 IEEE International Conference on Big Data Analysis (ICBDA)*, IEEE, 2016, pp. 1–5.

[10] B. Wang, A. Wang, F. Chen, Y. Wang, and C.-C. J. Kuo, "Evaluating word embedding models: Methods and experimental results," *APSIPA transactions on signal and information processing*, vol. 8, e19, 2019.

[11] M. Ait-Saada and M. Nadif, "Is anisotropy truly harmful? a case study on text clustering," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2023, pp. 1194–1203.

[12] Y. Yang, W. Wang, H. Fu, and C.-C. J. Kuo, "On supervised feature selection from high dimensional feature spaces," *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 1, 2022.

[13] C.-C. J. Kuo and A. M. Madni, "Green learning: Introduction, examples and outlook," *Journal of Visual Communication and Image Representation*, vol. 90, p. 103 685, 2023.

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

[15] M. Faruqui and C. Dyer, "Improving vector space word representations using multilingual correlation," in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, 2014, pp. 462–471.

[16] A. Conneau and D. Kiela, "Senteval: An evaluation toolkit for universal sentence representations," *arXiv preprint arXiv:1803.05449*, 2018.