2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC) | 978-1-7281-8028-1/20/$31.00 ©2021 IEEE | DOI: 10.1109/IAEAC50856.2021.9390956

A Brief Survey of Word Embedding and Its Recent Development

Qilu Jiao[1], Shunyao Zhang[1]*
1. Xiamen Institute of Software Technology Xiamen, China
jql@xmu.edu.cn,16097349@qq.com
* Corresponding author: Shunyao Zhang

***Abstract*—Learning effective representations of text words has long been a research focus in natural language processing and other machine learning tasks. In many early tasks, a text word is often represented by a one-hot vector in a discrete manner. Such a solution is not only restricted by the dimension curse, but also unable to reflect the semantic relationships between words. Recent developments focus on the learning of low-dimension and continuous vector representations of text words, known as *word embedding*, which can be easily applied to downstream tasks such as machine translation, natural language inference, semantic analysis and so on. In this paper, we will introduce the development of word embedding, describe the representative methods, and report its recent research trend. This paper can provide a quick guide for understanding the principle of word embedding and its development.**

***Index Terms*—word embedding, natural language processing, word2vec, BERT**

I. Introduction

How to obtain the effective representations of text words has long been a research focus in natural language processing and many other machine learning tasks [3], [15], [16], [19], [25]. Unlike image and audio, which can be represented by analog or digital signals, text words can not be directly processed or understood by the computer system. In this case, we often need a method to represent text words in a mathematical way, *e.g.,* converting each word into a binary or real-value vector. Such a study can be attributed to the *Vector Space Model* [22] originally used for information retrieval.

A simple yet common solution is to represent each text word with a one-hot vector [14]. Specifically, given a text corpus, we first create a vocabulary that contains all or the most important words of this corpus. Afterwards, each word can be represented by a binary vector, of which dimension is equal to the size of this vocabulary. The difference between these vectors are the word indexes, as shown in Fig.1. Since the construction is simple, this method has been widely used in some NLP tasks like document classification [24], accompanied with other techniques like *tf-idf* and *bag of words*.

However, the one-hot vector representation has obvious shortcomings. Above all, this method is often suffer from the *dimension curse* [3]. For instance, on the task of language translation [2], the number of involved text words can be millions. As a result, the corresponding one-hot vector will become extremely large and sparse, leading to the inefficiency of language modeling. In addition, these discrete vectors are unable to reflect the semantic relationships between words.

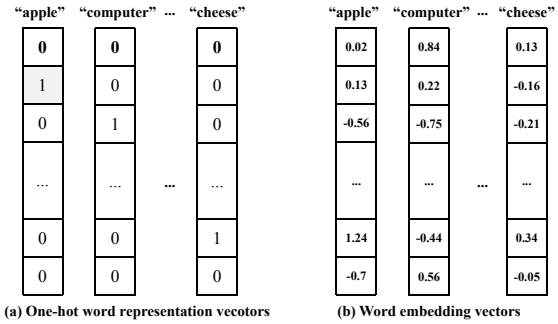

Fig. 1. Illustrations of word representations. The values of one-hot vector are binary, while the ones of word embeddings are continuous.

In this case, more research work focuses on learning low-dimension and continuous vector representations, also known as *word embedding*. The concept of *word embedding* can be contributed to the work of Bengio *et al.* [3], and its illustration is given in Fig.1.b. As shown in this figure, each word is represented by a real-value vector, and two similar text words will be also more close in the embedding space. The basic principle behind word embedding is based on the *distributed hypothesis* [9], [11], which states that words occurring in similar contexts tend to have similar semantics.

According to the methodologies used, the research of word embedding can be roughly divided into two main categories. The first one is global matrix factorization methods [4], [5], [7], [13], [29], such as latent semantic analysis (LSA) [5], which use low-rank approximation to decompose the large word co-occurrence matrices. These methods are also termed as *counted-based* or *syntagmatic* models [27]. They can well exploit the statistical information of a text corpus, but are also much inferior on the word analogy task. Meanwhile, their scalability to the large-scale corpus is also very limited.

The other is the sliding-window based methods, *e.g.,* Skip-Gram and CBOW [15], which uses shallow neural networks to perform text word prediction within local context window [3], [16], [17], [27]. In these methods, sentences are randomly sampled form the training corpus, and each word in these sentences are used to predict its context neighbors within a sliding window. These methods are also named *prediction-*

based or *paradigmatic* models [27]. Depending on the context-based prediction, these methods can capture the syntactic regularities in the text corpus, and performs much better on the word analogy task. Besides, methods like Skip-Gram or CBOW can be easily adapt to the large-scale training corpus [15]. However, these sliding-window based models are still criticized for the lack of global statistical information [19], [27]. To this end, there are also some method proposed to combine merits of both the matrix-factorization based and sliding-window based methods, such as GloVe [19].

The recent trend in word embedding is to learn the contextualized word representations [1], [6], [20], [31]. In these methods, each word is assigned by a representation that is a function of the entire input sentence, which can make word embeddings different in different linguistic contexts and address the issue of polysemy. Notably, in these methods, word embedding learning is integrated into a part of the optimization of the language model, which are trained on the large-scale unlabeled training data. Thus, the trained language model can not only output effective word representations in different linguistic contexts, but also can be used in the downstream tasks, *e.g.,* machine translation, natural language inference, semantic analysis and so on.

In this paper, we first introduce some representative methods of word embedding in Sec.II and its recent developments in Sec.III. Lastly, we draw a conclusion of this paper in Sec.IV.

II. Representative Methods

A. Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a classical method for obtaining distributed word representations proposed by Deerwester *et al.* [5]. LSA is proposed to release the issues of matrix sparsity of *Vector Space Model* (VSM), which can also reflects the latent semantics of the documents.

Specifically, given a text corpus of n documents and a vocabulary of m words, VSM will first construct a *word-doc* matrix denoted as $\mathbf{X} \in \mathbb{R}^{m \times n}$. Each row of $\mathbf{X}$ reflects the statics of the corresponding word on the text corpus, of which element can be binary value or *tf-idf* scores. The problem of $\mathbf{X}$ is that when the number of documents and the size of vocabulary are large, the matrix will become large and sparse as well, which leads to the computation inefficiency.

In this case, LSA apply *Singular Value Decomposition* (SVD) to decompose $\mathbf{X}$ into three small matrices. Concretely, given the *doc-doc* matrix, denoted as $\mathbf{B} = \mathbf{X}^T\mathbf{X}$, and the *word-word* matrix, denoted as $\mathbf{C} = \mathbf{X}\mathbf{X}^T$, the SVD process can be defined as:

$$\mathbf{X} = \mathbf{S}\Sigma\mathbf{U}^T, \tag{1}$$

where $\mathbf{S}$ and $\mathbf{U}$ are the matrices of the eigenvectors of $\mathbf{B}$ and $\mathbf{C}$, and Σ is the diagonal matrix of the singular values obtained as square root of the eigenvalues of $\mathbf{B}$. Since some of the singular values are "too small to neglect", LSA keeps k singular values in Σ, and reduces the dimensions of $\Sigma, \mathbf{S}$ and $\mathbf{U}$ correspondingly. Then, the truncated SVD is defined as:

$$\mathbf{X} \approx \mathbf{S}_k\Sigma_k\mathbf{U}_k^T, \tag{2}$$

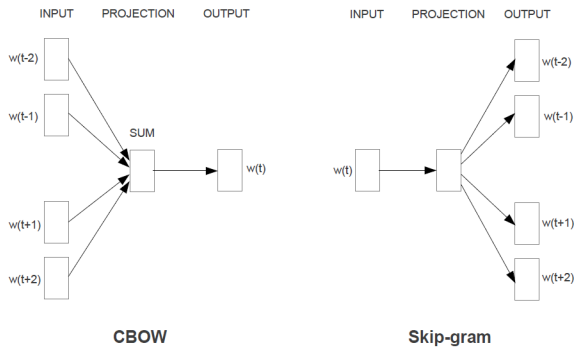

Fig. 2. Illustrations of CBOW and Skip-Gram [16].

where the dimensions of $\mathbf{S}_k, \Sigma_k, \mathbf{U}_k^T$ are $n \times k$, $k \times k$ and $m \times k$, and normally, $k \ll m, n$.

After the approximation, $\mathbf{U}_k$ can be used to represent text words on the given training corpus, and its dimension is much smaller than that of $\mathbf{X}$ and the sparsity is also greatly reduced. However, LSA is still not applicable to the large-scale text corpus due to the expensive computation of SVD.

B. Skip-Gram and CBOW

Skip-Gram and CBOW are two representative methods of the prediction-based word embeddings proposed by Mikolov *et al.* [15]. Their formulations are quite distinct from LSA, where word embeddings are built based on the word co-occurrence statistic. In contrast, these two methods optimize word embedding vectors via performing predictions of words in a sampled sequence.

Notably, the objective functions of Skip-Gram and CBOW are also different, as shown in Fig.2. The objective of CBOW is similar to Feedforward Neural Network Language Model (NNLM) [3], which is to predict the target word based on the context words within a sliding window. And the problem is formulated as:

$$p\left(w_t | \left\{w_0, w_1, ..., w_n\right\}\right), \tag{3}$$

where $\mathbf{w} = \{w_0, w_1, ..., w_n\}$ are the context words of w_t within a window of size n. In [15], the order of words is neglected during the prediction, so given a sequence of training words $\mathbf{w} = \{w_0, w_1, ..., w_T\}$, the loss function of CBOW is defined as:

$$\begin{aligned} E &= -\log p\left(w_t | \left\{w_0, w_1, ..., w_n\right\}\right) \\ &= -v_{w_t}'^T h + \log \sum_{j=1}^{V} \exp\left(v_{wj}'^T h_{w_t}\right), \end{aligned} \tag{4}$$

where V denotes the word vocabulary, v_w' denotes the word vector in the word embedding matrix, and h_w is the output of the hidden layer of the neural network.

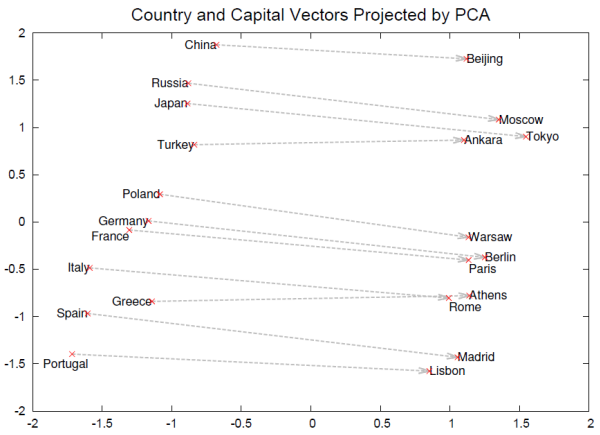

Fig. 3. Visualizations of the word vectors by Skip-Gram [15].

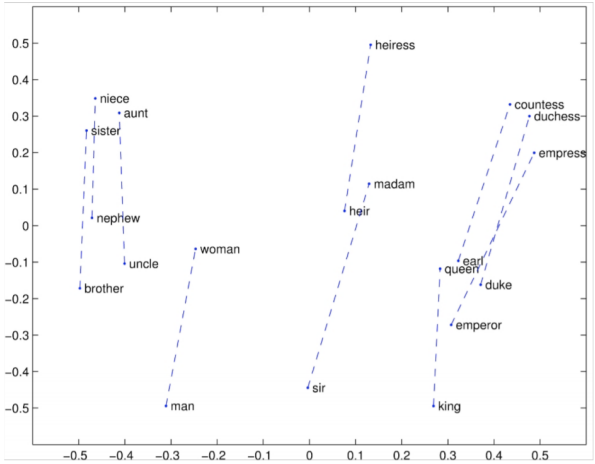

Fig. 4. Visualizations of word embeddings by GloVe [19].

Instead, the principle of Skip-Gram is to predict the context words based on the target word, as shown in Fig.2. In this case, the loss function of Skip-Gram is defined as:

$$\begin{aligned} E &= -\log p\left(\{w_0, w_1, ..., w_n\} | w_t\right) \\ &= -\log \prod_{n=1}^{N} \frac{\exp\left(v'^T_{wn} h_{wt}\right)}{\sum_{j=1}^{V} \exp\left(v'^T_{wj} h_{w_t}\right)}, \end{aligned} \tag{5}$$

where N denotes the number of context words.

From Eq.4 and Eq.5 we can see that, to accomplish the prediction, both two models need to compute the probability of every word in the vocabulary. This computation is very time consuming especially when the vocabulary size is large. In this case, Mikolov *et al.* apply two methods to optimize the *exp* terms in two models, which are *Hierarchical Softmax* [18] and *Noise Contrastive Estimation* (NCE) [10].

The structures of CBOW and Skip-Gram are similar to that of NNLM [3], which all use forward network to perform word predictions. The difference is that in CBOW and Skip-Gram, the network is more shallow, where the non-linear hidden layer is removed and the projection layer is shared for all words. The training complexity is also reduced. CBOW and Skip-Gram can well exploit the linguistic patterns as linear relationships between word vectors. As shown in Fig.3, the word vectors learned by Skip-Gram can accurately reflects their empirical semantics on the vector space, and performs better on the word analogy task, *e.g.,* the word of Beijing will be more close to the one of China than Japan.

C. GloVe

GloVe is a word embedding method proposed by Pennington *et al.* [19], which aims to combine the merits of the matrix factorization and the prediction-based methods. Specially, GloVe is to learn word vectors based on the information of word co-occurrence and local context.

In practice, GloVe first construct a matrix based on the word-word co-occurrence counts, denoted as $\mathbf{X}$, where its element X_{ij} denotes the count of word w_j appeared in w_i. With $\mathbf{X}$, we can obtain the probability of w_j appeared in the context of w_i, denoted as $P_{ij} = P\left(j|i\right) = X_{ij} / \sum_k X_{ik}$.

To include the word co-occurrence information into the local context prediction, GloVe introduce the *vector difference* to the embedding process. Then, a general embedding function F is defined as:

$$F\left(w_i - w_j, \hat{w}_k\right) = F\left(\left(w_i - w_j\right)^T, \hat{w}_k\right) = \frac{P_{ik}}{P_{jk}}, \tag{6}$$

where $w \in \mathbb{R}^d$ are word vectors and $\hat{w} \in \mathbb{R}^d$ are separate context word vectors. Meanwhile, F is required to be a homomorphism between the groups $(\mathbb{R}, +)$ and $(\mathbb{R}_{>0}, \times)$, *i.e.,*

$$\begin{aligned} F\left(\left(w_i - w_j\right)^T, \hat{w}_k\right) &= \frac{F\left(w_i^T, \hat{w}_k\right)}{F\left(w_j^T, \hat{w}_k\right)}, \\ where \quad F\left(w_i^T, \hat{w}_k\right) &= P_{ik} = \frac{X_{ik}}{X_i}. \end{aligned} \tag{7}$$

The solution is Eq.7 is $F = \exp$, or,

$$w_i^T \hat{w}_k = \log(P_{ik}) = log(X_{ik}) - log(X_i), \tag{8}$$

which is further simplified to:

$$w_i^T \hat{w}_k + b_i + \hat{b}_k = \log(X_{ik}). \tag{9}$$

Here, b_i and $\hat{b}_k$ are two biases to restore the symmetry. Lastly, the objective function of GloVe is defined as:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \hat{w}_j + b_i + \hat{b}_j - \log(X_{ij}))^2, \tag{10}$$

where f is a weighting function defined as:

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{if } x < x_{max} \\ 1 & \text{otherwise.} \end{cases} \tag{11}$$

Conclusively, GloVe combines the merits of LSA and CBOW, which is also training efficient and scalable to huge corpora. It can also achieve good performance with small corpus and small vectors. The visualization of word vectors by GloVe is shown in Fig.4.

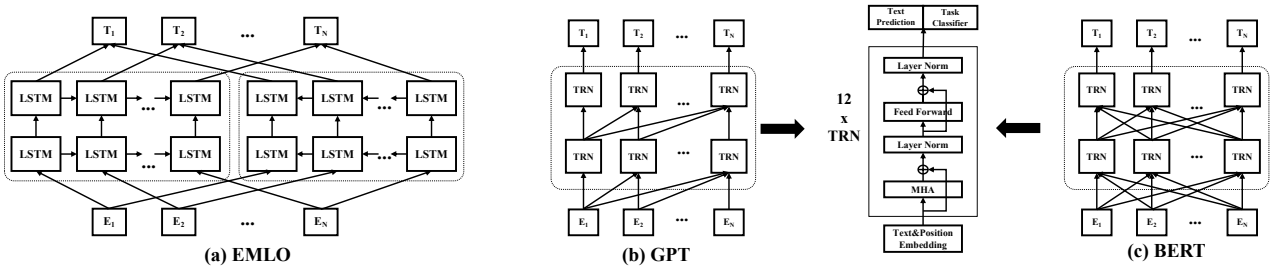

Fig. 5. Illustrations of Embedding from languae model (EMLO), Generative Pre-Training (GPT) and BERT. Here, TRN denotes Transformer layers,and MHA denotes multi-head attention.

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Fig. 6. Illustrations of polysemy of the word of "play" using GloVe and the context embeddings from EMLO [20]. Here, biLM dentoes the bi-directional LSTM network in EMLO.

III. Recent Developments

The word embeddings learned by the above methods are static, so the representation of each word is determined after training. Such a case leads to the issue of polysemy, *i.e.,* the semantic of a specific word may varies across different contexts. For instance, "*apple*" is known as a kind of fruit, but in technology reports it more often refers to a company.

In this case, the recent research trend is to include the learning of word embedding into the neural language models to obtain the contextualized word embedding. Then the representation vectors of words can be adjusted according to the input contexts, *e.g.,* sentences or documents. Three representative methods are illustrated in Fig.5.

A. EMLO

Embeddings from Language Model (EMLO) [20] is the first work to explore the deep contextualized word representations, of which structure is depicted in Fig.5-a. As shown in Fig.5-a, EMLO is consisted of a bi-directional LSTM [12], and the inputs are words of any sentence represented by static word embeddings, which can be randomly initialized or pre-trained [20]. To optimize the bi-directional LSTM network, EMLO adopts the target word predictions, *i.e.,* predicting words of the given sentences. After training, the hidden states of bi-directional LSTM can be used as the contextualized word representations.

In EMLO, the bi-directional LSTM network acts a role of extracting the contextualized word features, which can well address the issue of polysemy as shown in Fig.6. Meanwhile, since the model is optimized by the sentence word prediction, EMLO is applicable to the large-scale text corpus, which subsequently improve the ability of language understanding of the language model.

B. GPT

Generative Pre-training (GPT) [1] is a pre-training based language model, of which principle is very similar to EMLO. As shown in Fig.5, the main difference is that GPT use the stacked Transformer encoding layers [30] as the feature extractor, which is a very powerful attention-based neural network. Compared to the LSTM networks, Transformer is superior in capturing the long-term dependencies between input elements. Similar to EMLO, GPT is also uses the text word prediction to optimize the language network, which is also applicable to the large-scale unsupervised pretraining. And the outputs before the prediction layer can be used as the contextualized word representations.

C. BERT

BERT is another pre-training-oriented language model, of which structure is given in Fig.5-c. From the figure we can see that, the structure of BERT is very close to that of GPT, which also uses a 12-layer Transformer network as the language model. The difference is that BERT consider the bi-directional language modeling of context words, similar to that of EMLO. Besides, BERT also includes some new technical designs such as *position embedding*, which is combined with the static word embeddings as the model input. And the optimization tasks used are *Masked Language Modeling*, *i.e.,* masking a certain words in the sentence, and *Next Sentence Prediction*, *i.e.,* predicting whether two sentences are together in the document. Similar to GPT, the outputs of the Transformer network can be used as the contextualized word representations.

Notably, the contributions of these models are not limited to the deep contextualized word embeddings, and they also greatly improve the language understanding ability of networks via large-scale unsupervised pre-training. In practice, they are fine-tuned on downstream tasks and achieve new SOTA performance on various NLP tasks, such as *question answering* [21], *named entity extraction* [23], *document classification* [8], [8], [28], *semantic analysis* [26] and machine translation [2] and so on.

IV. Conclusion

In this paper, we give a brief introduction to word embedding and its development. The learning of the low-dimension and distributed word embeddings is a key step in the development of natural language processing, which can facilitate the study of various language understanding tasks. We describe the definition of three static word embedding methods in detail, which can help people understand the principle behind word embedding. Meanwhile, we also introduce three recent methods of word embedding. Conclusively, this paper can provide a quick guide for understanding word embedding.

Acknowledgement

This paper is supported by Social Science Research Funded Project of The Education Department of Fujian Province, China (JAT191605).

References

[1] R. A. improving language understanding by generative pre-training.
[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR 2015 : International Conference on Learning Representations 2015*, 2015.
[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(6):1137–1155, 2003.
[4] R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.
[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the Association for Information Science and Technology*, 41(6):391–407, 1990.
[6] J. Devlin, M.-W. Chang, K. Lee, and K. N. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2018.
[7] P. Dhillon, D. P. Foster, and L. H. Ungar. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems 24*, volume 24, pages 199–207, 2011.
[8] W. B. Dolan and C. Brockett. Automatically constructing a corpus of sentential paraphrases. *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.
[9] J. R. Firth. *Studies in Linguistic Analysis.* 1974.
[10] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.
[11] Z. S. Harris. Distributional structure. *WORD*, 10(2):146–162, 1954.
[12] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
[13] R. Lebret and R. Collobert. Word emdeddings through hellinger pca. *arXiv preprint arXiv:1312.5542*, 2013.
[14] Y. Li and T. Yang. Word embedding for understanding natural language: A survey. pages 83–104, 2018.
[15] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *ICLR (Workshop Poster)*, 2013.
[16] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048, 2010.
[17] A. Mnih and G. Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648, 2007.
[18] F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *AISTATS*, 2005.
[19] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.
[20] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2227–2237, 2018.
[21] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.
[22] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of The ACM*, 18(11):613–620, 1975.
[23] E. F. T. K. Sang and F. D. Meulder. Introduction to the conll-2003 shared task: language-independent named entity recognition. In *CONLL '03 Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, pages 142–147, 2003.
[24] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
[25] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 151–161, 2011.
[26] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.
[27] F. Sun, J. Guo, Y. Lan, J. Xu, and X. Cheng. Learning word representations by jointly modeling syntagmatic and paradigmatic relations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 136–145, 2015.
[28] Y. Tay, L. A. Tuan, and S. C. Hui. A compare-propagate architecture with alignment factorization for natural language inference. *arXiv: Computation and Language*, 2017.
[29] P. D. Turney and P. Pantel. From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, 2010.
[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.
[31] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5753–5763, 2019.