A Review on Word Embedding Techniques for Text Classification

S. Selva Birunda and R. Kanniga Devi

Abstract Word embeddings are fundamentally a form of word representation that links the human understanding of knowledge meaningfully to the understanding of a machine. The representations can be a set of real numbers (a vector). Word embeddings are scattered depiction of a text in an *n*-dimensional space, which tries to capture the word meanings. This paper aims to provide an overview of the different types of word embedding techniques. It is found from the review that there exist three dominant word embeddings namely, Traditional word embedding, Static word embedding, and Contextualized word embedding. BERT is a bidirectional transformer-based Contextualized word embedding which is more efficient as it can be pre-trained and fine-tuned. As a future scope, this word embedding along with the neural network models can be used to increase the model accuracy and it excels in sentiment classification, text classification, next sentence prediction, and other Natural Language Processing tasks. Some of the open issues are also discussed and future research scope for the improvement of word representation.

Keywords Word embeddings · Natural language processing · Bag of words · Contextualized word embeddings · Text classification · Transformer

1 Introduction

Word embedding is a technique used to map words from vocabulary to vector of real numbers. This mapping causes the words that emerge from a similar context that can be correlated with each other.

S. Selva Birunda (✉) · R. Kanniga Devi
Department of Computer Science and Engineering, School of Computing, Kalasalingam Academy of Research and Education, Anandnagar, Krishnankoil, Srivilliputtur, Virudhunagar, Tamil Nadu 626126, India
e-mail: sbirunda@gmail.com

R. Kanniga Devi
e-mail: rkannigadevi@gmail.com

J. S. Raj et al. (eds.), *Innovative Data Communication Technologies and Application*,
Lecture Notes on Data Engineering and Communications Technologies 59,
https://doi.org/10.1007/978-981-15-9651-3_23

The free text words of the vocabulary are converted into numeric values (vectors). This transformation is necessary as many Machine Learning models can understand only the vector representation. One of the elementary transformation approaches is one-hot encoding, where each word determines one dimension and the binary value (1 or 0) indicates the presence or absence of the word. One-hot encoding is computationally impossible when dealing with the entire dictionary, as the representation demands thousands of dimensions. Besides, this encoding has no hidden relationships among the words.

Word embedding is different from the conventional embedding in a way that it represents phrases and words in vectors of numeric values (which is non-binary). It refers to a dense representation of words in low dimensional vector space. For example, the words "Boy," "grapes," "man" is distributed in *n-dimensional* vector space. The distance between the words boy and man is smaller than between boy and grapes; thus, there is a similarity between the words "boy" and "man." This embedding provides a hidden semantic relationship among the words.

Currently, word embeddings are one of the successful applications of unsupervised learning as they do not need pricey annotation. Word embeddings are represented in a way that similar words have similar encoding. Word embeddings are useful in feature generation and Natural Language Processing tasks like text classification, document clustering, and sentiment classification. Word embedding technique can be categorized into Traditional or Frequency word embedding, Static word embedding, and Contextualized word embedding.

Frequency-based embeddings are categorized into count vector, TF-IDF, and co-occurrence. Static word embeddings are categorized into Word2Vec, glove, and fast text. Contextualized word embeddings are categorized into Elmo, GPT-2, and BERT.

This paper focuses on the review of different word embedding techniques, how they are used to represent the text, capture meanings, similarity among the text, and importance of the text.

2 Literature Survey

It is found from the literature survey that there exist three major word embedding techniques namely, Traditional word embedding, Static word embedding, and Contextualized word embedding. The following subsections present survey on these techniques.

2.1 Traditional Word Embedding

Traditional word embedding is based on the frequency that considers the whole document and discovers the significance of rare words in the document, count occurrence of each word, and co-occurrence of words.

El-Din [1] proposed a new enhancement bag of words method to solve the issue of manual evaluation of words. This embedding automatically evaluated the sentiment polarity and score it by using words weight. The proposed technique obtained a sentiment classification accuracy of 83.5% and could be extended to concentrate on phrases for the detection of the polarity.

Soumya et al. [2] proposed an approach to identify co-occurrence features from Wikipedia pages and incorporated this feature with a bag of words model. Co-occurrence was added to solve the issue of neglecting the word order. These features along with the naive bayes classifier brought about an F1 score of around 95% and could be enhanced further to use different classifiers.

Enríquez et al. [3] showed a word representation that is based on vectors, obtained through Word2Vec based on a bag of words. Bag of words (BoW) model is relevant to word embedding techniques as both techniques transform the word/text to numerical form. In the BoW approach, each document is depicted by vectors, where each dimension agrees to a word or group of words, producing vectors of high dimensionality. The BoW is a lexical representation relying on frequency-based metrics that describes the presence of a particular word in the sentence resulting in huge, sparse vectors. Word embedding is an alternative approach that converts text to numbers. This technique represents the words of vocabulary in a vector space, which has dimensions lower than the vocabulary size. Unlike BoW, word embedding technique traverses the lexical boundary as this word representation capture syntactic and semantic aspects. The combined representation of both BoW and Word2Vec provides higher accuracy of around 75%.

Tripathy et al. [4] compared Naive Bayes and Support Vector Machine (SVM) classifiers. To depict textual data as a numerical form, count vectorizer and term frequency-inverted document frequency (TF-IDF) embeddings were used. These embeddings, along with the SVM classifier achieved a maximum classification accuracy of 94% and could be boosted further to compare the maximum entropy classifier and stochastic gradient classifier.

Qu et al. [5] proposed a new TF-IDF method which added an improvement weight formula, as TF-IDF embedding studied only the relationship between the documents and not between the characters. The new TF-IDF along with the distance vector classifier obtained classification effect with an F1 score of 80–90% and might be enhanced to obtain more classification accuracy.

Dadgar et al. [6] proposed a novel text mining approach to classify news based upon TF-IDF embedding. This embedding calculated the weight of the word in the document. TF-IDF along with the SVM classifier achieved better performance using BBC and 20 newsgroup datasets, with precision around 97 and 94%. This could be augmented using different classifiers.

Jing et al. [7] proposed a new TF-IDF approach based on the improved TF-IDF. TF-IDF embedding used the IDF function to adjust the feature, whereas, new TF-IDF used the mutual information evaluation function. This new TF-IDF along with the vector space model achieved a classification precision of 88% and could be enhanced further to improve feature selection.

Kuang et al. [8] proposed a new feature weighting method, TF-IDF. Ci, an improvement of TF-IDF embedding. TD-IDF could not reflect the importance of degree and categorical differences. This issue got solved using the feature weighting method, which achieved a large macro F1 value of around 92.806. This could be extended further to take the fingerprint of the text for a copy test study.

Matsuo et al. [9] proposed an approach to extract the keyword from a single document using word co-occurrence embedding. Co-occurrence between frequent terms and each term were generated. This representation showed the importance of a term in the document. The proposed method achieved precision around 0.5 and further applied to domain-independent keyword extraction.

Albathan et al. [10] presented a co-occurrence embedding that extracted the high-quality patterns from the text by using weighted patterns. A pattern co-occurrence matrix was used to identify the relationship among extracted patterns and to reduce the noisy and closed sequential patterns. As a result, the number of extracted patterns was 89.04 and the noise could be reduced further.

Kadhim et al. [11] proposed a TF-IDF and co-occurrence embedding which adopted the cosine similarity to extract the features. The cosine similarity score function is calculated among the pair of vectors that depend on the number of common words and the frequency of the word in the document. TF-IDF and TF-IDF Global were compared and the results showed that features were reduced by 10–20% and could be improved further to reduce the high dimensionality of feature space.

Lott [12] surveyed the TF-IDF embedding which assigns weight to a given term to decide how the term characterizes an individual document inside a corpus. TF-IDF embedding could be used along with the naive bayes classifier to discover key phrases present in the document. This might be extended to include semantic analysis, lexical analysis, or co-occurrence measures.

Wang [13] suggested feature extraction by calculating the weight based on word co-occurrence. The degree of co-occurrence was calculated by the degree of relationship among each text. When the classification was high, the improved method seemed to be higher with an accuracy of 74% and could be enhanced further to consider synonyms, and the changed words.

Wartena et al. [14] proposed a co-occurrence embedding that measured the semantic relations among the words by calculating the co-occurrence distribution of words. This embedding excels in terms of precision, recall, and F1 score. Improvement in percents for top 5 and top 10 keywords in ACM dataset afford 9.2 and 6.9% precision, 10 and 8% recall, 9.7, and 7.3% F1 score and could be extended to improve precision and recall furthermore.

From the detailed survey of Traditional word embedding techniques, it is found that the “Co-Occurrence matrix” provides better results as this technique maintains the semantic relationship between the words.

2.2 Static Word Embedding

Static word embedding is a prediction-based that provides probabilities to the words and maps each word into a vector. Static embeddings learn by training the lookup tables which converts words into dense vectors. This embedding is static in the way that they do not alter the context once been learned and also the embedding tables do not change among different sentences.

Ge et al. [15] proposed a word embedding, Word2Vec that analyzed the semantic similarity between the words at a high precision rate. Feature reduction was achieved by loosely clustering similar features using a graph search technique. Multinomial Naïve Bayes classifier, SVM classifier, K nearest neighbor and random forest classifiers were compared. Word2Vec embedding with the SVM classifier achieved the highest accuracy of 93.93% and extended to clarify the loose clustering technique and a refined text corpus.

Pennington et al. [16] proposed a word embedding model called "Glove," Global vectors for word representation. The glove was a global log bilinear regression model based on the count data, it captured the linear substructures. Glove model performed well in the word similarity, word analogy, and named entity recognition tasks. The glove was based on the unsupervised model to generate vectors, and thus, provided accuracy and F1 score around 90% and could be improved to perform additional tasks.

Alrashdi et al. [17] proposed a model for processing the twitter tweets, it got preprocessed by removing hashtags, punctuation, emojis, and stop words. Glove embedding and crisis embedding and two architectures CNN and Bi–LSTM. Glove word embedding along with the Bi-LSTM reported the highest performance with the F1 score of 62.04%. This could be instructed further to use N-gram, CNN for crisis response in classifying the tweets.

Zhou et al. [18] proposed a Fast Text word embedding along with the KNN classifier provided high performance and preserved with pruning. This approach was suitable for online text classification applications. TC-Apte (Reuters version 3 corpus) and TC-PARC (Reuters version 4) generated a pruning ratio of about 82.8 and 72.1%. This might be extended for Chinese text corpus to knob multi-label and multi-class text classification.

Joulin et al. [19] proposed a word embedding approach "Fast Text," which was evaluated on two different tasks such as sentiment analysis and tag prediction. Fast text was trained on more than one billion words within a limited period. This method was able to classify half a million sentences within a minute. The maximum test accuracy of Fast Text evaluated with and without bigrams was 98.1 and 98.6%, and it could be improved furthermore.

Kuyumcu et al. [20] proposed a new approach Fast Text word embedding developed by Facebook. Fast Text embedding took into account the internal structure of words in the Turkish language. Fast text embedding assumed a word to be n-gram of characters. Fast Text embedding classified the text achieving an accuracy of 94% and might be enhanced further to different language texts.

Rezaeinia et al. [21] proposed a word embedding method improved word vector (IWV) which enhanced the accuracy of word embedding vectors by combining the POS tagging, lexicon-based approaches, and Word2Vec/Glove methods. Pre-trained word embeddings were trained on a large text corpus. The experimental result showed that the IWV method achieved accuracy around 80% and it could be considered as the base for sentiment analysis techniques.

Lilleberg et al. [22] suggested an approach based on the combination of word embeddings Word2Vec and TF-IDF. This combination without stop words outperformed the individual performance of themselves, and along with the SVM classifier, it provided 90% accuracy on text classification. This could be improved further to combine TF-IDF with Word2Vec along with the meaning of the words.

Vora et al. [23] proposed a word embedding approach to classify tweets based on emotions using the random forest classifier. The twitter tweets were classified into emotion categories. Word2Vec, Glove, and Fast Text had been used to generate word representations for the text. These word embeddings along with the random forest classifier achieved 91% precision and embellished further by classifying the text as more fine-grained classes.

Stein et al. [24] analyzed word embeddings along with the classifiers showed improvement in automatic document classification tasks. Word embeddings such as Glove, Word2Vec, and Fast Text were compared. The experimental results showed that the Fast Text with a classifier achieved an LCA F1 score of 0.893. Furthermore, applying Fast Text to the PubMed data analyzed whether this embedding extended to the medical text context.

Tezgider et al. [25] improved the word representations by tuning the Word2Vec parameters for Turkish text. For measuring vector size, word count, and window size, parameters were used. Word embedding quality was improved by properly selecting the parameters. TF-IDF, Fast Text, Glove, and Word2Vec embeddings along with the deep learning classifier model improved the accuracy of classification around 89% and could be enhanced to other language text.

Elsaadawy et al. [26] proposed Fast Text and Word2Vec embedding that create a vector using a weighted average of words representation. Naïve Bayes log count ratio was used to find the weight of each word. These embeddings were compared with the traditional representation such as TF-IDF and Naïve Bayes (SVM). Both the models provided a maximum classification accuracy of 90.26 and 89.93% and might be extended to use the ensemble classifiers.

Mikolov et al. [27] focused on the importance of pre-trained word representations predicted from large text corpus like Wikipedia, Web Crawl, and news collections. Fast Text and Glove embeddings were compared and pre-trained on text classification tasks along with the above corpus. Fast Text embedding resulted in achieving the maximum classification performance on an average accuracy of 82.7% and enhanced further to use this model in various tasks.

Raunak et al. [28] presented a novel technique that combined principal component analysis (PCA) based dimensionality reduction along with the post-processing algorithm. This algorithm could improve the pre-trained word vectors. Word embeddings used in memory-constrained devices could be improved by reducing their size. The

reduced embeddings achieved maximum accuracy around 90% and explored further to use the word embeddings influenced by the contextualized embeddings.

From the detailed survey of Static word embedding techniques, it is found that the "Fast Text" provides better results as this technique can efficiently handle a lot of rare words that are present in the text corpus.

2.3 Contextualized Word Embedding

Contextualized word embedding is based on the context of a particular word, in which, similar words will have contrast context representations. These representations dynamically vary based upon the context in which a word appears.

Peters et al. [29] proposed deep Contextualized word representations, embeddings from language models (ELMo). ELMo embedding was pre-trained on a large text corpus. It depicted the syntactic and semantics information and word that had different contexts. ELMo embedding along with the neural network achieved classification accuracy around 97%. This might be improved further by applied to different NLP tasks.

Manoharan [30] proposed a smart algorithm for image processing by recognizing the text and vocalization for the visually challenged. LattePanda Alpha system on board was used to process the scanned images. Following the pre-processing, segmentation, extraction of features, images were classified into its similar alphanumeric characters. The textual data was converted into audio output by making use of a speech synthesizer. This model provided accuracy of about 97% and enhanced further to use Machine Learning algorithm to process the textual information and to boost the accuracy of the system and minimize the time delay for processing the information.

Maslennikova [31] compared the performance of ELMo and Word2Vec embedding. ELMo embedding increased the quality of prediction and to solve a binary classification problem. By using ELMo embedding, the F1 score was increased by 10% than the Word2Vec embedding. ELMo embedding could be enhanced by combining it with contrasting pre-trained models adopting different datasets.

Alsentzer et al. [32] explored Bidirectional Encoder Representations for Transformers (BERT) models for clinical text. Clinical researchers got benefitted from these embeddings. Some models trained on the clinical text and some others fine-tuned as BioBERT. BioBERT trained the BERT model over PubMed which was the source of biomedical research article corpus. This clinically trained BERT model provided high accuracy and an F1 score above 90% and could be further use for other clinical NLP research.

Wu et al. [33] proposed a novel data augmentation method called BERT for labeled sentences. BERT was a deep bidirectional model that is more powerful than the unidirectional and shallow bidirectional model. The conditional masked language model, an improvement of the masked language model to which conditional constraint was applied. This model outperformed other embeddings with an accuracy of around

79.60%. This might be enhanced further by applying BERT contextual augmentation to paragraph or document level.

Chang et al. [34] proposed X-BERT, a solution to fine-tune BERT embedding. This solved the difficulty in capturing dependencies that occurred in BERT. X-BERT denoted eXtreme multi-label text classification. X-BERT was used to abduct the contextual relations between the induced label cluster and the input text. This embedding achieved a precision rate of 68% and further extended by many fine-tuned BERT models to achieve high performance.

Schwartz et al. [35] explored the combination of two-word level embeddings for performing binary classification tasks. ELMo and BERT combination did not improve the classification ability. ELMo or BERT combined with the Glove embedding could improve the classification performance with maximum x^2 value as 12 and 16.69 and improved further by evaluating their performance in multiple datasets.

Jwa et al. [36] adopted BERT embedding to design an automatic fake news detection model "BAKE" (BERT applied to FAKE). Extra unlabeled news information added to the BAKE is "exBAKE." Both models were an automated process to detect fake news. These models worked on the FNC-1 dataset by examining the relationship between headlines and the body text of news articles and exceeded the performance by 0.125 and 0.137 F1 scores. This could be enhanced further to add additional news in the pre-training phase.

Han et al. [37] proposed domain adaptive fine-tuning, a simple approach for the unsupervised labeling applied to new domains. The contextualized embeddings were adapted through masked language modeling in the text. Fine-tuning of ELMo and BERT embeddings gave ample improvements with impressive results. The proposed model achieved an accuracy of 83% and extended further to focus on the source and the target domains.

Chakraborty et al. [38] analyzed the performance of sparse vectorizer that outperformed the neural word embeddings like Word2Vec, Glove, and Fast Text and character embeddings like ELMo. This embedding improved their performance and increased their F1 score by 3–5% and could be extended further by using neural classifiers like CNN and RNN instead of static embeddings.

Felipe et al. [39] compared the different models. Prediction-based embeddings predicted the probability of the next word. Count-based models leveraged global co-occurrence statistics in word. High-Level word embeddings encoded multiple relationships between words. Word2Vec, Glove, and Fast Text resulted in more accurate and faster embeddings. This might be improved further for high-level entities like sentences and documents.

Ethayarajh [40] examined the Contextualized word embeddings such as ELMo, BERT, and GPT-2 and how contextual they were. The upper layers of these embeddings produced more context-specific representations than the lower layers. Less than 5% of the variance on average, in contextualized representations, might be defined by static embedding. This could be extended to accomplish static embeddings as even more isotropic.

From the detailed survey of Contextualized word embedding techniques, it is found that the "BERT" provides better results as this technique can perform various

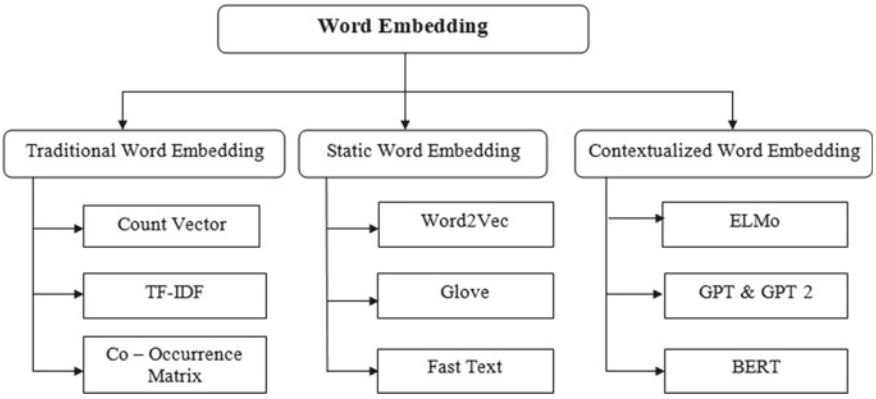

Fig. 1 Types of word embedding techniques

NLP tasks such as finding similarity between the texts, prediction of next sentence sequence, text classification.

Figure 1 shows the types of word embedding techniques which is found from the literature survey.

Table 1 Depicts the description, merits and demerits of each type of word embedding technique.

3 Future Scope

Contextualized word embeddings have attained incredible achievement in major NLP tasks. Even though, there are still a variety of problems that remain unexplored. In this section, some challenging problems are presented and future research directions.

Multi-Task Learning (MTL) is an active approach in Machine Learning and Natural Language Processing which desires to improve the performance of learning representations by making use of the information efficiently. In MTL, multiple learning tasks are figured out at the same time; i.e., multiple related tasks are studied together, and advantageous information is shared among the tasks. Liu et al. [41] conferred a multi-task deep neural network, to combine BERT language model pre-training and MTL for learning language representations obtaining state of the art performance on general language understanding evaluation (GLUE) benchmark at 82.2%. From the above perspective, it is better to manipulate further research by combining MTL and language model pre-training as both are integral to each other.

Few Shot Learning refers to the application of providing a learning model with a limited amount of training data. Conventional methods rely on hand-crafted features. Currently, it focuses on metric learning; an approach that depends upon the distance metric which finds similarity or dissimilarity between the objects. This metric method

Table 1 Merits and demerits of the types of word embedding techniques

Word embedding		Description	Merits	Demerits
Traditional	Count vector	Count vector is a method of counting the number of times each word occurs in the document	Count vectors can be used on real text data to provide accurate counts of the word content	Count vector did not afford semantic information and similarity between two documents
	TF-IDF (term frequency-inverse document frequency)	TF-IDF is calculated by the product of the frequency of the word in a particular sample and frequency of the word in the whole document	(i) Insists on the importance of a particular term by the rare occurrence of the word in all the documents (ii) It can easily compute the similarity between two documents	(i) Could not capture semantics, co-occurrences in different documents (ii) In the task of retrieving, it could not differentiate between singular and plural words
	Co-occurrence vector	The co-occurrence matrix is based on the concept of similar words in the same context that tend to occur together	It can maintain the semantic relationship between the words and is foster	Needed a large memory size to store the co-occurrence matrix
Static	Word2Vec	Word2Vec method can generate word embedding by using dense representation. It is a predictive model and it provides probabilities to the words which excel in word similarity tasks	(i) It can alter unlabeled corpus into labeled data by mapping the target word to its context word (ii) Sub-linear relationships are not defined implicitly	(i) It does not use global information (ii) Sub-linear relationships are not defined explicitly

(continued)

Table 1 (continued)

Word embedding		Description	Merits	Demerits
	Glove	The Glove is a count-based model. It is an unsupervised model for generating word vectors. Glove combines two features namely, local context window method and global matrix factorization	Glow relies on both local context information of words and global statistics (word co-occurrence). It can derive semantic relationships. It can predict the neighboring words by maximizing probability through the log bilinear model with a weighted least square objective. Word vectors can apprehend sub linear relationships	The Glove is framed on the co-occurrence man ix. It needs more memory for storage
	Fast text	Fast text is an extension of Word2Vec. It identifies the words as *n*-gram of characters. It provides an efficient vector representation of rare words	Fast Text can supply the vector representations for the words that are not present in the OOV words (dictionary), It can handle unseen words and it is word Fragment-Based	It does not add contextual information
Contextualized	ELMo	ELMo embedding is context-dependent and is character-based. A word may have dissimilar meanings depending upon the context where it is being used	ELMo provides multiple word embed dings for a single word	ELMo is shallow bidirectional as it cannot take advantage of both left and right contexts simultaneously

(continued)

Table 1 (continued)

Word embedding		Description	Merits	Demerits
	GPT-2	GPT-2 denotes Generative Pre-trained Transformer 2. It is a decoder only transformer	GPT-2 can predict the next word by looking at parts of a sentence. It can give more than ten possible predictions that what will be the next word using the probability score	GPT-2 requires heavy computation, chance to create false information as it is trained over millions of websites
	BERT	BERT is the first unsupervised deep bidirectional system with a multi-layer bidirectional Transformer encoder	BERT is used to learn the contextual relations between the words or sub words. It holds both syntactic and semantic meanings of a text. It can make assumptions for the blank word in between the sentence	BERT can only handle sentences of minimum length

studies the sample representation, that acts as a word embedding for token level Natural Language Processing tasks. This leads to the demand for designing embedding committed to few-shot learning samples. Hou et al. [42] reduced representation ambiguity using a special query support arrangement in a few-shot learning and pairwise embedding.

Pre-training models like ELMo, BERT, and GPT are trained by **self-supervision**. Here, unlabeled corpus is reassembled to get the labeled data and these tasks depend upon the co-occurrence of sentences and words. Syntactic, semantic, and lexical information have been ignored which is very important. Sun et al. [43] suggested learning the tasks through MTL using a continual pre-training framework. Various pre-training tasks concerning semantic, structure, and word-level are examined which yields auspicious results. Thus, imposing this model to learn strong representations by designing challenging tasks.

4 Conclusion

From the detailed literature survey, it is found that there exist three major word embedding techniques namely, Traditional word embedding, Static word embedding, and Contextualized word embedding. Each word embeddings constitutes three techniques. Traditional word embeddings are frequency-based; Static word embeddings are Prediction-Based, and Contextualized word embeddings are contextually-based. Further, transformer-based Contextualized word embedding along with the neural network can be applied to tasks such as text classification, document clustering, and sentiment analysis to improve the accuracy of classification and clustering. Finally, open issues and promising future scope in word representations have been discussed.

References

1. El-Din DM (2016) Enhancement bag-of-words model for solving the challenges of sentiment analysis. Int J Adv Comput Sci Appl (IJACSA)7(1):244-252
2. Soumya George K, Joseph S (2014) Text classification by augmenting bag of words (BOW) representation with co-occurrence feature. IOSR J Comput Eng (IOSR-JCE) 16(1):34–38
3. Enríquez F, Troyano JA, López-Solaz T (2016) An approach to the use of word embeddings in an opinion classification task. Exp Syst Appl 66:1–6
4. Tripathy A, Agrawal A, Rath SK (2015) Classification of sentimental reviews using machine learning techniques. Proc Comput Sci 57:821–829
5. Qu S, Wang S, Zou Y (2008) Improvement of text feature selection method based on TFIDF. In: 2008 international seminar on future information technology and management engineering. Leicestershire, United Kingdom, pp 79–81
6. Dadgar SMH, Araghi MS, Farahani MM (2016) A novel text mining approach based on TF-IDF and support vector machine for news classification. In: IEEE international conference on engineering and technology (ICETECH), Coimbatore, pp 112–116
7. Jing L-P, Huang H-K, Shi H-B (2002) Improved feature selection approach TFIDF in text mining. In: Proceedings of the first international conference on machine learning and cybernetics. IEEE, vol 2, pp 944–946
8. Kuang Q, Xu X (2010) Improvement and application of TF·IDF method based on text classification. In: 2010 international conference on internet technology and applications. Wuhan, pp 1–4
9. Matsuo Y, Ishizuka M (2004) Keyword extraction from a single document using word co-occurrence statistical ınformation. Int J Artif Intell Tools 13(01):157–169
10. Albathan M, Li Y, Algarni A (2012) Using patterns co-occurrence matrix for cleaning closed sequential patterns for text mining. In: Li Y, Zhang Y, Zhong N (eds) Proceedings of the IEEE/WIC/ACM international conference on web ıntelligence and ıntelligent agent technology, vol 1, pp 201–205
11. Kadhim AI, Cheah Y-N, Ahamed NH, Salman LA (2014) Feature extraction for co-occurrence-based cosine similarity score of text documents. In: 2014 IEEE student conference on research and development, pp 1–4
12. Lott B (2012) Survey of keyword extraction techniques. UNM Educ 50:1–11
13. Wang LH (2014) An ımproved method of short text feature extraction based on words co-occurrence. Appl Mech Mater 519:842–845 (Trans Tech Publications Ltd.)
14. Wartena C, Brussee R, Slakhorst W (2010) Keyword extraction using word co-occurrence. In: 2010 workshops on database and expert systems applications. IEEE, pp 54–58

15. Ge L, Moh T-S (2017) Improving text classification with word embedding. In: 2017 IEEE ınternational conference on big data (big data), Boston, MA, pp 1796–1805
16. Pennington J, Socher R, Manning CD (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
17. Reem A, O'Keefe S (2019) Deep learning and word embeddings for tweet classification for crisis response. In: The 3rd national computing colleges conference. arXiv preprint arXiv:1903.11024.
18. Zhou S, Ling TW, Guan J, Hu J, Zhou A (2003) Fast text classification: a training-corpus pruning based approach. In: Eighth international conference on database systems for advanced applications, 2003 (DASFAA 2003). Proceedings, Kyoto, Japan, pp 127–136
19. Joulin A, Grave E, Bojanowski P, Mikolov T (2017) Bag of tricks for efficient text classification. In: Proceedings of the 15th conference of the European chapter of the association for computational linguistics, vol 2, Short Papers, pp 427–431
20. Kuyumcu B, Aksakalli C, Delil S (2019) An automated new approach in fast text classification (fastText). A case study for Turkish text classification without pre-processing. In: Proceedings of the 2019 3rd international conference on natural language processing and ınformation retrieval, pp 1-4
21. Rezaeinia SM, Ghodsi A, Rahmani R (2017) Improving the accuracy of pre-trained word embeddings for sentiment analysis. arXiv preprint arXiv:1711.08609
22. Lilleberg J, Zhu Y, Zhang Y (2015) Support vector machines and Word2Vec for text classification with semantic features. In: IEEE 14th international conference on cognitive informatics & cognitive computing (ICCI*CC), Beijing, pp 136–140
23. Vora P, Khara M, Kelkar K (2017) Classification of tweets based on emotions using word embedding and random forest classifiers. Int J Comput Appl 178(3):1–7
24. Stein RA, Jaques PA, Valiati JF (2019) An analysis of hierarchical text classification using word embedding. Inf Sci 471:216–232 (Elsevier)
25. Tezgider M, Yıldız B, Aydın G (2018) Improving word representation by tuning Word2Vec parameters with deep learning model. In: 2018 international conference on artificial intelligence and data processing (IDAP), Malatya, Turkey, pp 1–7
26. Elsaadawy A, Torki M, Ei-Makky N (2018) A text classifier using weighted average word embedding. In: 2018 international Japan-Africa conference on electronics, communications and computations (JAC-ECC). Egypt, pp 151–154
27. Mikolov T, Grave E, Bojanowski P, Puhrsch C, Joulin A (2017) Advances in pre-training distributed word representations. arXiv preprint arXiv:1712.09405
28. Raunak V, Gupta V, Metze F (2019) Effective dimensionality reduction for word embeddings. In: Proceedings of the 4th workshop on representation learning for NLP (RepL4NLP-2019), pp 235–243
29. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. arXiv preprint arXiv:1802.05365 [cs.CL]
30. Manoharan S (2019) A smart image processing algorithm for text recognition, information extraction and vocalization for the visually challenged. J Innov Image Process (JIIP) 1(01):31–38
31. Maslennikova E (2019) ELMo word representations for news protection. In CLEF (Working Notes).
32. Alsentzer E, Murphy J, Boag W, Weng W-H, Jindi D, Naumann T, McDermott M (2019) Publicly available clinical BERT embeddings. In: Proceedings of the 2nd clinical natural language processing workshop, pp 72–78
33. Wu X, Lv S, Zang L, Han J, Hu S (2019) Conditional BERT contextual augmentation. In: Rodrigues J et al (eds) Computational science—ICCS 2019. Lecture notes in computer science, vol 11539. Springer, Cham
34. Chang W-C, Yu H-F, Zhong K, Yang Y, Dhillon I (2019) X-BERT: eXtreme multi-label text classification using bidirectional encoder representations from transformers. arXiv preprint arXiv:1905.02331 [cs.LG]

35. Schwartz A (2020) Combining word embeddings for binary classification tasks
36. Jwa H, Oh D, Park K, Kang JM, Lim H (2019) exBAKE: automatic fake news detection model based on bidirectional encoder representations from transformers (BERT). Appl Sci 9:4062
37. Han X, Eisenstein J (2019) Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing, pp 4238–4248. arXiv:1904.02817 [cs.CL]
38. Chakraborty R, Elhence A, Arora K (2019) Sparse victory—a large scale systematic comparison of count-based and prediction-based vectorizers for text classification. In: Proceedings of the international conference on recent advances in natural language processing (RANLP 2019), pp 188–197
39. Felipe A, Xexéo G (2019) Word embeddings: a survey. arXiv preprint arXiv:1901.09069
40. Ethayarajh K (2019) How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. arXiv preprint arXiv:1909.00512
41. Liu X, He P, Chen W, Gao J (2019) Multi-task deep neural networks for natural language understanding. arXiv preprint arXiv:1901.11504
42. Hou Y, Zhou Z, Liu Y, Wang N, Che W, Liu H, Liu T (2019) Few-shot sequence labeling with label dependency transfer. arXiv preprint arXiv:1906.08711
43. Sun Y, Wang S, Li Y, Feng S, Tian H, Wu H, Wang H (2019) Ernie 2.0: a continual pre-training/framework for language understanding. arXiv preprint arXiv:1907.12412