Short Text Data Mining Based on Incremental AP Clustering

Fuyu Lu, Ying Guo, Peiyi Qu, and Yonglin Leng(✉)

College of Information Science and Technology, Bohai University, Jinzhou 121000, China
lengyonglin@qq.com

Abstract. The rapid development of mobile internet technology generates many short text data, which contains many hot topics. By clustering short text data, we can identify many hot topics in time. This information is crucial for discovering public opinion and analyzing user emotions. This paper proposes a hybrid vector representation model (HVRM) that combines weight and topic features to address the feature information loss caused by a single short text vector representation model and short text sparsity. Firstly, HVRM mines the local features using Word2Vec and TF-IDF to get the weighted vector of short text. Next, use BTM to obtain global feature vectors. And then connect the two feature vectors to form short text vectors. Finally, we use KNN to initialize the responsibility and availability matrices of incremental AP clustering (IAPC). The experimental results show that the hybrid vector representation model proposed in this paper can effectively improve the clustering effect.

Keywords: Short Text · Vector Representation Model · Incremental AP Clustering

1 Introduction

The rapid development of mobile internet technology allows people to express themselves online at any time. Weibo, WeChat and others have become the main ways for people to communicate. These platforms generate a large amount of text data, especially short text. These data contain lots of available information. Mining these data are beneficial for discovering hot topics, emotional tendencies that users are concerned about, and strengthening public opinion monitoring [1]. How to extract valuable information from short text is a hot research topic for scholars. Clustering technology is an unsupervised data classification method. It classifies data with similar features into one class by analyzing the similarity between data. Because the features of text data are not obvious, it is difficult for traditional processing methods to calculate the similarity between data accurately [2]. An effective method is to use deep learning to map short text to a low dimensional vector space and use vectors to describe text features. The self-feature extraction based on neural network is increasingly receiving attention from the industrial and academic communities. Based on previous research, Mikolov et al. [3] proposed the

V. C. M. Leung et al. (Eds.): Qshine 2023, LNICST 573, pp. 363–373, 2024.
https://doi.org/10.1007/978-3-031-65126-7_31

Word2Vec model in 2013 for calculating word vectors. Word2Vec model utilizes the contextual information of words to transform a word into a low dimensional real vector space, and the more similar words are, the closer they are in the vector space. The application of word vectors in natural language processing has been very successful and has been widely applied in fields such as Chinese word segmentation [4, 5], sentiment classification [6], and syntactic analysis [7–9]. Word2Vec model reflects the contextual associations, which only focuses on a certain range of neighboring vocabulary relationships, which can easily lead to the loss of global information. BiTerm Topic Model (BTM) [10] discovers the topic distribution characteristics of a text through the co-occurrence of vocabulary information and the probability distribution between documents, topics, and vocabulary, thereby discovering the global semantic information and feature expression of the text. This paper proposes a hybrid vector representation model(HVRM) that integrates weight and topic features to address the problems of current single text feature representation models. HVRM is based on local and global features of short text to vectorize the short text. Then, an incremental AP clustering(IAPC) algorithm is proposed to cluster short text. Finally, the effectiveness of the proposed model in short text topic discovery is verified by comparing it with traditional methods.

2 Preliminaries

2.1 Vectorization of Words

Vectorization is to map words to a vector space and expresses a word mathematically. There are usually two ways to vectorize words.

(1) One-hot Encoding

One-hot encoding is a vectorized representation of words with a vector length equal to the size of the dictionary [11]. The components of a vector are composed of 0 and 1, where the position of component 1 is the corresponding position of the word in the dictionary, and the rest of the bits are all 0. Although one-hot encoding can clearly represent a word, it cannot express its semantics. Furthermore, when the dictionary size increases, its dimensional features are sparse, and each sparse column has a linear relationship, which is prone to collinearity problems.

(2) Word2Vec

Hinton first proposed mapping a word into a low dimensional, dense real vector space. If the meaning of two words is closer, the distance between them in the vector space is closer. Obviously, using this representation can better distinguish the similarity between words. Mikolov proposes Word2Vec by drawing inspiration from Bengio's Neural Network Language Model [12] and Hinton's Log_Linear model [13]. Word2Vec expresses words in vector through optimized training models based on a fixed corpus. There are two models for Word2Vec, namely CBOW and Skip-gram model. CBOW model uses k words before and after the current predicted word to predict the current word, while Skip-gram model uses the current word to predict its k words before and after.

2.2 BiTerm Topic Model

The topic model is an unsupervised learning algorithm that automatically organizes, searches, and understands a lot of documents. Such models can find topics that span many documents together. Latent Dirichlet Allocation (LDA) builds a model by calculating the importance of terms in documents [14]. When analyzing hot topics, it is difficult to determine the importance of words due to the short text, which leads to data sparsity. To address this defect, BiTerm Topic Model (BTM) obtains a pair of unordered words to model and learn the whole data corpus. This method avoids the sparse short text data in topic modeling.

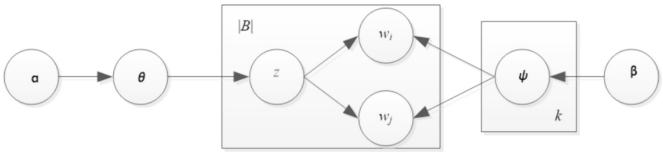

Fig. 1. BTM model

The structure of BTM is shown in Fig. 1, where α, β is the Dirichlet prior parameter, the matrix θ represents the probability distribution of the document topic, and k represents the number of topics. Each line of the matrix θ represents the probability distribution of each document under each topic, such as $\theta_i = (z_{i1}, z_{i2}, ..., z_{ik})$ is the topic vector of the document D_i.

The matrix φ is the probability distribution of topic words. Each column of the matrix φ represents the probability distribution of each word under each topic, such as $\varphi_i = (z_{1i}, z_{2i}, ..., z_{ki})$ is the topic vector of the word w_i in the vocabulary.

The number of word pairs is |B| in corpus. For the word pairs in the corpus, the modeling process of BTM is described as follows:

(1) For the corpus, there is a top distribution $\theta \sim Dir(\alpha)$.
(2) For each topic z, the word distribution under that topic is $\varphi_z \sim Dir(\beta)$.
(3) For each word in the set of B: $b = (w_i, w_j)$:
   a. Randomly select a topic z from the distribution of topics θ in the corpus, and then get z~*Multi*(θ).
   b. Randomly select two different words $w_i$andw_j that make up the word pair b from the extracted topic z, and there are w_i, w_j ~*Multi* (φ_Z).

2.3 Affinity Propagation Clustering

Affinity Propagation (AP) clustering is a new clustering algorithm first proposed in the article "Clustering by Passing Messages Between Data Points" in 2007 [15]. AP takes the similarity matrix of data points as the input. In the initial stage of clustering, AP does not require setting the clustering numbers and centers. The algorithm considers all data as the centers of potential clustering, and then continuously searches for suitable data points through iterative calculations, automatically identifies the clustering centers between data points and determines the clustering numbers. Data points find potential

samples by calculating similarity. Give any two data objects i and j, $S(i, j)$ represents the suitability of data point i as the clustering center of data point j. The diagonal of the similarity matrix indicates the data point i as the reference of the clustering center. The larger the value, the more likely it will become the clustering center.

In order to obtain a suitable clustering center, AP clustering needs to continuously transmit information during the iteration process. We use R and A to represent the responsibility and availability matrix in message passing. $R(i, k)$ indicates that data i sends information to data k, reflecting the degree to which data k serves as the clustering center of data i. $A(i, k)$ indicates that data j sends information to data i, reflecting the suitability of data point k as the clustering center for data i. Initially, the availability matrix A is initialized to zero. Then calculate the responsibility matrix R using the following formula:

$$R_{t+1}(i,k) = (1-\lambda)R_{t+1}(i,k) + \lambda R_t(i,k) \tag{1}$$

$$R_{t+1}(i,k) = \begin{cases} S(i,k) - \max\limits_{j \neq k}\{A_t(i,j) + R_t(i,j)\},\ i \neq k \\ S(i,k) - \max\limits_{j \neq k}\{S(i,j)\},\ i = k \end{cases} \tag{2}$$

The iterative formula for the availability matrix is as follows:

$$R_{t+1}(i,k) = (1-\lambda)R_{t+1}(i,k) + \lambda R_t(i,k) \tag{3}$$

$$A_{t+1}(i,k) = \begin{cases} min\{0, R_{t+1}(k,k) + \sum_{j \notin \{i,k\}} max(0, R_{t+1}(j,k)\},\ i \neq k \\ \sum_{j \neq k} max\{0, R_{t+1}(j,k)\},\ i = k \end{cases} \tag{4}$$

The availability matrix represents the sum of self attraction $R(k, k)$ and positive attraction obtained by other points, i.e. $R(k, k) > 0$, because only positive attraction supports k as the clustering center. During the implementation of the algorithm, a damping coefficient λ with a value of [0.5,1] is used to prevent numerical oscillations during the update process.

3 Hybrid Vector Representation Model

3.1 Construction of Hybrid Vector Representation Model

HVRM combines weights and subject features. Figure 2 depicts the process of generating short text vectors. The work mainly includes word segmentation, removing stop word, low frequency word and word of non-Chinese characters, filtering out low-quality and repeated text. Then, for the stage of processed short text, Word2Vec training is used to obtain the word vector representation of each word segment. TF-IDF algorithm [16] is used to calculate the weight value of each word in the short text, and the word vector is multiplied with the weight of the word in the short text to obtain the weighted word vector. BTM is used to obtain the document topic distribution matrix. Finally, we connect the weighted vector with BTM document topic vector to form a short text feature vector.

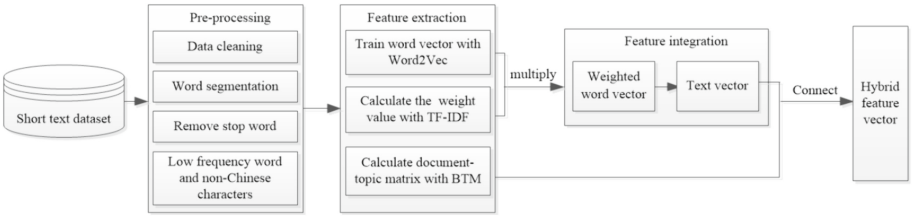

Fig. 2. Hybird vector representation model

3.2 Representation of Document Weighted Vector

Given a short text set $D = \{D_1, D_2, D_3, \ldots, D_m\}$, each short text is divided into several words. We use Word2Vec to obtain the word vector. The vector of w_i is represented as $w_i = (w_{i1}, w_{i2}, \ldots, w_{id})$ in the word list, where d represents the dimension of each word vector, and the size of word list is n.

Word2Vec reflects the semantic association between short text vocabulary, but Word2Vec only focuses on a certain range of vocabulary relationships, which can easily lead to the loss of global information. For the whole short text set, the contribution of each word to the topic of a certain short text is different. Therefore, this paper uses TF-IDF to calculate the weight value of w_i in the short text D_j. As shown in formula (5), where $tf(w_i, D_j)$ represents the frequency of word w_i appearing in a single short text D_j, $idf(w_i)$ represents the weight of word w_i in set D, $N(w_i, D_j)$ represents the frequency of word w_i appearing in D_j, $N(D_j)$ represents the total number of word in set D, and $N(w_i)$ is the number of short text where word w_i appears.

$$K(wi, Dj) = \frac{tf(w_i, D_j) \times idf(w_i)}{\sqrt{\sum_{w_i \in D_j} [tf(w_i, D_j) \times idf(w_i)]^2}} \tag{5}$$

$$tf(w_i, D_j) = \frac{N(w_i, D_j)}{N(D_j)} \tag{6}$$

$$idf(wi) = \log\left(\frac{M}{N(w_i)} + 0.01\right) \tag{7}$$

Assuming the short text $D_j = (w_1, w_2, \ldots, w_t)$, which w_i is a word that corresponds to a vector. We multiply the word vector of w_i and the weight of w_i in the short text to obtain the weighted word vector.

The vector representation of each short text in set D is shown in formula (9)

$$y_i = w_i * K(w_i, D_j) \tag{8}$$

$$\delta_i = \frac{\sum_{j=1}^{t} y_j}{|D_i|} \tag{9}$$

Word2Vec reflects the semantic association between lexical sequences and ignores the global semantics of the text. In order to make up for the lack of global information,

this paper selects BTM to find the topic distribution characteristics of the text through the probability distribution of lexical co-occurrence information and documents, topics, words. BTM first assigns a unique serial number i-1 to each word w_i in the corpus dictionary W. At the same time, it constructs the structured short text $D_i = (w_{i1}, w_{i2}, w_{i3},\ldots, w_{in})$ after pre-processing, where w_{in} represents the serial number corresponding to the n_{th} word in short text D_i. Input the structured short text of the corpus and the short text set into BTM, BTM forms word pair set B by extracting all co-occurrence word $b = (w_i, w_j)$, and then randomly assigns topic z to the co-occurrence word pair. After Gibbs sampling, the document topic distribution matrix θ is obtained. Each line of θ represents the probability distribution of short text under each topic, which constitutes a document topic vector.

3.3 Connection of Short Text Vector

BTM and Word2Vec have their own emphasis when vectoring short text. To highlight their respective characteristics, we connect the text weighted word vector with BTM's document topic vector to form a document feature vector R, which makes up for the shortcomings of BTM and Word2Vec and enriches the semantic information of short text vector.

$$R_i = \delta_i \oplus \theta_i \tag{10}$$

4 Incremental AP Clustering

The topic of short text has the characteristics of wide range, strong timeliness, and fast update. The static clustering algorithms cannot meet the demand for real-time discovery of hot topic. In order to quickly discover relevant hot topics in massive changing data, this paper proposes an incremental AP clustering algorithm(IAPC). Assuming that the original short text dataset $D = \{D_1, D_2, \ldots, D_t\}$, the new short text dataset is $I = \{ID_1, ID_2,\ldots ID_p\}$, $D \cap I = \varnothing$, and $t + p = n$. IAP first clusters the original short text dataset to obtain the responsibility and availability matrices. Secondly, we use KNN to obtain the extended responsibility and availability matrices of the newly added data, and then continue iteratively passing the message until convergence.

After clustering the original dataset, the data points in the dataset have accumulated a lot of support. That is, through message transmission, the responsibility and availability matrices are non-zero. For the new data, the responsibility and availability between the new data and other data are zero. If we continue with the previous message transmission, the differences in the responsibility and availability make it difficult for the new data to become the new clustering center. If the data are similar, they not only have a high probability of belonging to the same clustering, but also should have similar responsibility and availability matrices. Based on this, we use KNN to construct the responsibility and availability matrices of the new data, where the corresponding values in the responsibility and availability matrices of the new data are replaced by the values of the K-nearest neighbors of the data points.

Given the responsibility matrix R_t and availability matrix A_t, for the new data $ID_{i'}(t + 1 \leq i' \leq t + p)$, calculate its similarity with the original data, and obtain K-nearest neighbors of the new data, that is, $KNN(I_{i'}) = \{D_{q1}, D_{q2}, \ldots D_{qk}\}$, where $1 \leq q_1, q_2, \ldots q_k \leq t$. The extended responsibility matrix is shown in formula (11):

$$R_{t+p}(i,j) = \begin{cases} R_t(i,j) & i \leq t, j \leq t \\ \frac{1}{k}\sum_{m \in KNN(i)} R_t(m,j) & i > t, j \leq t \\ \frac{1}{k}\sum_{m \in KNN(j)} R_t(i,m) & i \leq t, j > t \\ 0 & i > t, j > t \end{cases} \tag{11}$$

Similarly, the availability matrix A_{t+p} is defined as follows:

$$A_{t+p}(i,j) = \begin{cases} A_t(i,j) & i \leq t, j \leq t \\ \frac{1}{k}\sum_{m \in KNN(i)} A_t(m,j) & i > t, j \leq t \\ \frac{1}{k}\sum_{m \in KNN(j)} A_t(i,m) & i \leq t, j > t \\ 0 & i > t, j > t \end{cases} \tag{12}$$

The process of IAPC algorithm is:

For the original data, the responsibility information of each data in the similarity matrix is updated, and the availability information is calculated;

Update the availability information and calculate the availability of each data;

Sum up the responsibility and availability information of each data to make a decision. If the preset number of iterations is reached, or the clustering center no longer changes, or the decision of the sample data within a sub region does not change after several iterations, the iteration can be terminated.

Calculate the K-nearest neighbors of each new data at regular intervals, construct the responsibility and availability matrices, and continue iteration until convergence.

5 Experiments

The dataset comes from the Sina Weibo in March 2021, including a total of 10125 pieces of data. The data have 8 topics which are "Burden Reduction", "Property Market", "Russia-Ukraine Conflict", "Vaccine", "Huawei Chip", "Tokyo Olympics", "Japanese nuclear wastewater", "Suez Canal ship grounding". In the stage of pre-processing, the data is denoised, including removing stop words and repeated text. Finally, we obtained 9832 valid data. The experimental environment is Intel(R) Core(TM) i5-12600KF 3.70 GHz, 16.0GB memory, 500GB hard disk, Windows 11 operating system, and Python programming language.

5.1 Determination of the Number of Topics

In order to verify the feasibility of BTM in the short text clustering process, first we calculate the perplexity of different topic numbers and determine the topic category K through the perplexity curve. At the same time, we extract the top10 keywords under

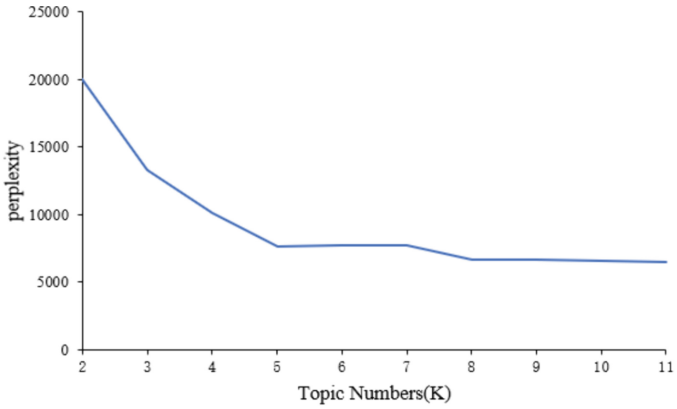

Fig. 3. Perplexity of BTM

each topic. As shown in Fig. 3, the perplexity curve tends to be stable when K = 8, which is consistent with the real topic category of the dataset.

In addition, we compare the top10 keywords of BTM and LDA under each topic. Through these keywords, it is possible to understand the content expressed by each topic. However, due to the wide coverage of some topics, such as the "Tokyo Olympics" includes multiple sub topics. Therefore, the topic center is not prominent. Comparing the keywords determined by BTM and LDA, we find that some keywords determined by LDA are not related to the topic, such as the keywords "OPPO" in the topic "Huawei, Chip", and the keywords "flu" in the topic "Vaccine". This further indicates that BTM is more suitable for short text mining.

5.2 Clustering Accuracy Analysis

In the experiment, the dimension of word vector is set to 100, and the window size is set to 4. According to the perplexity of BTM, the document theme dimension is set to 8, the hyper-parameters $\alpha = 0.5, \beta = 0.01$. The number of iterations in the Gibbs sampling process is set to 1000.

We select four models: Word2Vec(WV), Word2Vec+TF-IDF(WT), Word2Vec+BTM(WB), and Word2Vec+TF-IDF+BTM(HVRM) to obtain short text vector. Then, AP algorithm is used to cluster the short text and get F1-score value for each clustering. Figure 4 shows the results of short text clustering under various feature vector models. It can be seen HVRM proposed in this paper has the best clustering effect. Except for the topic of "Tokyo Olympic", the overall accuracy is higher than other models. The reasons mainly include: Word2Vec focuses on local relationships between words to reduce the dimension of text vectors, while ignoring the global semantic information and the contribution of different words to the topic. So, there is a certain gap in clustering accuracy compared to HVRM. To some extent the feature weights or topic features of words enhances the topic features of word vectors, so the clustering effect is significantly higher than Word2Vec. HVRM not only overcomes the global semantic loss, but also enhances the

influence of feature words through TF-IDF method. At the same time, the integration of BTM does not significantly increase the feature dimensions of short text, so it does not reduce the efficiency of the algorithm.

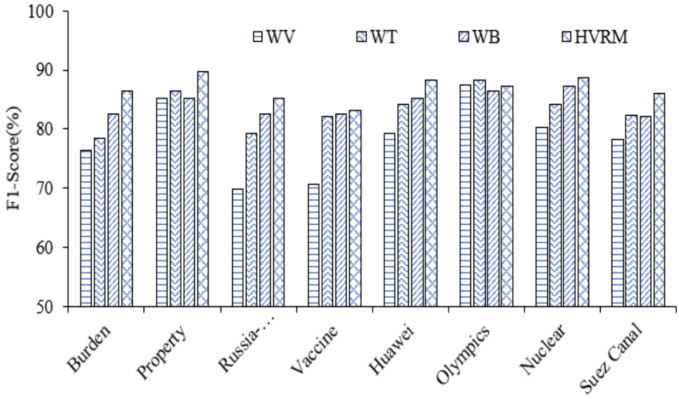

Fig. 4. Comparison for different feature vector models

5.3 Comparison AP and IAP

To verify the performance of IAP, we compare F1-Score values of each clustering after clustering the overall dataset using static AP and IAP. Set the whole dataset as C and randomly divide C into 10 subsets. When performing IAP, add one subset at a time. Figure 5 shows the experimental results.

We can see that the average F1-Score of clustering is 84.85%, while the average F1-Score value of IAP is 84.30%, with little difference between the two algorithms. When gradually adding subsets, their clustering accuracy is lower than the overall clustering accuracy at first, due to the uneven random extraction of subsets, resulting in lower accuracy than AP. But as the data increases, its clustering accuracy will increase, and the overall F1-Score of two algorithms are similar, indicating that IAP is effective. However, as shown in Fig. 6, the IAP efficiency is higher than AP.

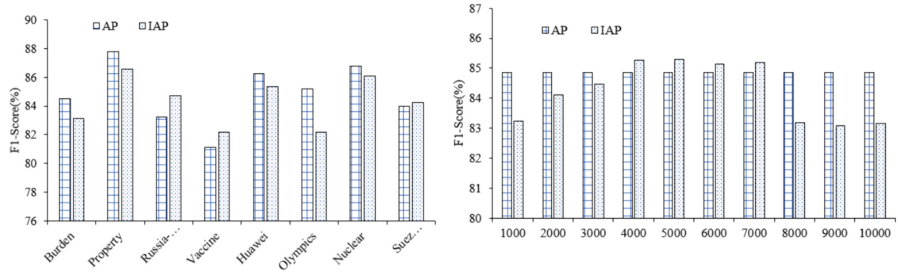

Fig. 5. Comparison for different clustering algorithm

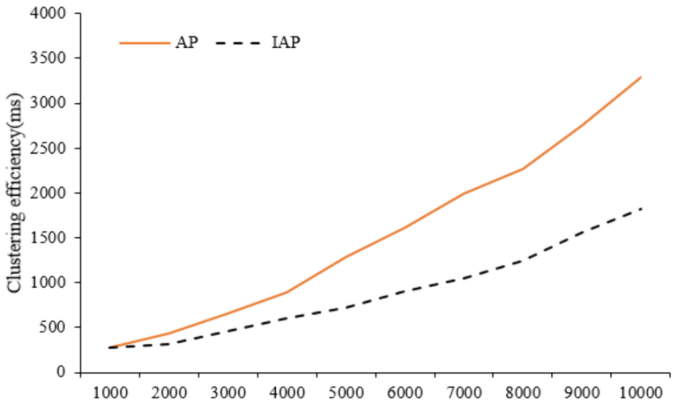

Fig. 6. Comparison for cluster efficiency

6 Conclusions

This article proposes a short text vector representation model that integrates weights and topic features. At the same time, it combines the improved incremental AP clustering algorithm to cluster the short text data of Sina Weibo and discover hot topics that the public is concerned for a period of time. HVRM mines short text data features from local and global perspectives, which solves the problem of feature information loss caused by sparse short text data. The improved incremental AP clustering solves the problem of asynchronous updating processes of responsibility and availability matrices and improves the AP clustering efficiency. Experimental results show that the short text vector representation model proposed in this paper can effectively improve clustering performance.

In future work, other feature factors of short text will be considered, such as the impact of the timeliness of short text on clustering algorithms to discover hot issues.

Acknowledgments. This work is partially supported by the Liaoning Social Science Planning Fund Project under Grant L14AGL002, L13AGL002 and the Liaoning Soft Science Foundation Project under Grant 22022JH4/10100052.

References

1. Yang, L., Li, C., Ding, Q., et al.: Combining lexical and semantic features for short text classification. Procedia Comput. Sci. **22**, 78–86 (2013)
2. Pradhan, R., Sharma, D.K.: A hierarchical topic modelling approach for short text clustering. Int. J. Inf. Commun. Technol. **4**, 20 (2022)
3. Mikolov, T., Sutskever, I., Chen, K., et al.: Distributed representations of words and phrases and their compositionality. Adv. Neural. Inf. Process. Syst. **26**, 3111–3119 (2013)
4. Reynolds, A.P., Richards, G., Rayward-Smith, V.J.: The application of K-Medoids and PAM to the clustering of rules. In: Yang, Z.R., Yin, H., Everson, R.M. (eds.) IDEAL 2004. Lecture Notes in Computer Science, vol. 3177, pp. 173–178. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28651-6_25
5. Aggarwal, C.C.: An introduction to uncertain data algorithms and applications. IEEE Trans. Knowl. Data Eng. **21**(5), 609–623 (2009)
6. Lu, M., Feng, G., Fan, M., et al.: New clustering algorithms for large data processing. Syst. Eng. Electron. **5**, 1010–1015 (2014)
7. Gullo, F., Ponti, G., Tagarelli, A.: Clustering uncertain data via k-medoids. In: Greco, S., Lukasiewicz, T. (eds.) Scalable Uncertainty Management, pp. 229–242. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87993-0_19
8. Xie, L., Li, L.: Research on multi-attribute group decision under interval number information. Comput. Eng. **40**(10), 210–213 (2014)
9. Zhou, B., Xu, Y., Tang, Q.: New method for determining optimal number of clusters in K-means clustering algorithm. Comput. Eng. Appl. **46**(16), 27–31 (2010)
10. Cheng, X., Yan, X., Lan, Y., et al.: BTM: topic modeling over short texts. IEEE Trans. Knowl. Data Eng. **26**(12), 2928–2941 (2014)
11. Gu, B., Sung, Y.: Enhanced reinforcement learning method combining one-hot encoding-based vectors for CNN-based alternative high-level decisions. Appl. Sci. **11**(3), 1291 (2021)
12. Bengio, Y., Schwenk, H., Senécal, J.S., Morin, F., Gauvain, J.L.: Neural probabilistic language models. In: Holmes, D.E., Jain, L.C. (eds.) Innovations in Machine Learning. Studies in Fuzziness and Soft Computing, vol. 194, pp. 137–186. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-33486-6_6
13. Mnih, A., Hinton, G.: Three new graphical models for statistical language modelling. In: 24th International Proceedings on Machine Learning, pp. 641–648, ACM, Oregon (2007)
14. Blei, D.M., Ng, A., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
15. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315**, 972–976 (2007)
16. Hall, P.: The SMART retrieval system: experiments in automatic document processing. Inf. Storage Retrieval **9**(3), 199 (1971)