Soft clustering method for text mining, with an opportunity to attribute them to different semantic groups

Dmitriy Zhukov[1], *Elena* Andrianova[1] ,*Konstantin* Otradnov[1],and *Leonid* Istratov[1,*]

[1]Moscow Technological University (MIREA), 119454 Moscow, Russia

Abstract. The work describes new soft clustering method for text mining, developed by the authors, with an opportunity to attribute them to different semantic groups. The work of this algorithm is based on usage of definition parameter, when deterimining clustering accuracy, which can be defined for texts of various subject areas either on basis of percolation properties of text clusters, or estimated theoretically based on the redundancy model for text messages. By the author's assumption , proposed algorithm must have speed and accuracy for clustering of texts , depending on value of clustering accuracy parameter. The advantage of this algorithm is that no need to set an initial value of clusters.

1 Introduction

Analysis of unstructured or semi-structured natural language text date, is the very important task to search for hidden patterns in large volumes of data and for creating of predictive models of different processes (including social and economic ones) and extraction of knowledge. A significant number of existing models of analysis is associated with computer linguistics, based on the presentation of texts in vector form and their further clustering by semantic groups.

The construction of models for the clustering of textual documents is based on the methodology for selecting the essential features for the solution of the problem of information retrieval. It should be noted that the set of attributes required for analysis of hidden patterns in large amounts of data is a priori unknown, so when constructing such models it is necessary to use the minimum number of assumptions, with the maximum preservation of the various features used for clustering. However, it should be borne in mind that the run time of the model should remain acceptable, and the implementation of the computation does not require significant computing resources.

You can list the following requirements, which, if possible, the clustering algorithm should meet:

1. At operation of algorithm there should be a minimum possible quantity of passes on a database.
2. Work should be carried out in a limited amount of computer RAM.
3. The work of the algorithm can be interrupted with the preservation of intermediate results, in order to continue the calculations later.
4. The algorithm should work when objects from the database can be retrieved only in the mode of the unidirectional cursor (ie, in the navigation mode by records).

Thus, it is desirable that the algorithm should be scalable (if the capacity of the RAM is unchanged, with an increase in the number of records in the database, its operating time increases linearly).

2 Overview of some modern clustering methods

There are a large number of studies devoted to the development of methods and algorithms for clustering, the description of which can only be presented in a separate paper. It should be noted that a significant part of them is devoted to the improvement of existing algorithms that have already become classical.

Modern technologies provide a huge number of ways in which you can implement document clustering [1]: Custom Search Folders method; LSA/LSI – Latent Semantic Analysis/Indexing;

The methods of complex or single communication of the middle group; method of specific dispersion; method of recursive bisection, etc. It is worth highlighting several methods to which many articles are devoted.

The method of Custom Search Folders [2], which consists in the formation of folders for the separation of textual information by categories. This technology is no longer a new one and is implemented within the search system, with the choice of categories [2].

LSA / LSI - Latent Semantic Analysis / Indexing uses factor analysis techniques to search for hidden factors, on the basis of which cluster analysis groups are formed. The application of LSA is reflected in [3].

The methods of complex or single communication, called Single Link [4], Complete Link [4], involve

*Corresponding author: kuyahshtibov@gmail.com

splitting the entire set of processed documents into clusters using dendrograms and agglomerative clustering (constructing a dendrogram with such a special functionality that defines merging clusters).

At the same time, it should be noted that the issue of effective and qualitative clustering is not yet fully resolved and it is necessary to search for new models and algorithms.

Recently, the Expectation-maximization algorithm or the EM-algorithm [5, 6] is becoming very popular, which, like K-means, is also iterative. In it, first, some initial approximation for the existing clusters is set, and then a refinement is carried out. Vectors are redistributed and cluster centers are calculated, the positions of which are considered in this model as a parameter of the distribution function. Then a new redistribution is carried out until the specified clustering accuracy is reached.

In order to solve the problem of clustering with the help of this algorithm, it is necessary to estimate the probability of belonging of each vector to the cluster, in addition, it is necessary to know that there are a priori probabilities of clusters - w_1, ..., w_k, of cluster distribution density p_1 (x), ..., p_k (x), and of the distribution density of the feature vector x.

In work [7], to solve the problem of analyzing a billion pages of natural language texts in the internet, we propose a scalable parallel stream EM algorithm for clustering a sample of web scales that can be used on inexpensive hardware.

The authors of [7] present an algorithm for a parallel streaming EM tree for clustering a sample of web scales on inexpensive hardware. The quality of clustering solutions and the efficiency of processing depend on the representation of objects.

Therefore, when presenting the data model in [7], random projections are used to represent a collection of documents in the form of binary signatures. The authors of [7] apply the algorithm of the parallel streaming EM-tree, which is specialized for binary signatures for segmenting the sample into clusters. The proposed method is evaluated using Web data sets ClueWeb09 and ClueWeb12, containing respectively 500 and 733 million documents in English. This experiment was carried out by the authors on the basis of the two-element system Xeon E5-2665.

Each processor has an 8 × 2.4 GHz core frequency, connected in a ring network, for a total of 16 processor cores. The program used 64 GB of memory to store a tree in memory, where The data set takes from 240 GB to 350 GB on the disk. Data streams were transferred from the SATA disk at 7200 rpm to 3 TB, providing 150 MB per second of serial read performance.

The ClueWeb 09 and 12 sets took about two and three days to index and resulted in indices at 240GB and 350GB. This is a brief representation of the sets, which are measured in dozens of terabytes when uncompressed. The EM tree processes these documents to create clusters. ClueWeb 09 and 12 clustering took approximately 15 and 20 hours. Both worked in five iterations, and produced 700,000 and 600,000 clusters, respectively.

In [8], the authors presented, using the EM-tree algorithm, solutions for two basic problems of clustering on web pages-scalability and efficiency. The EM tree algorithm can assemble hundreds of millions of documents in hundreds of thousands of clusters on a single 16-core processor in less than 24 hours. Disadvantages of this work is that the approach does not cluster documents that are sparse and high-dimensional. Also, the approach requires significant computing resources, which is far from available for most organizations.

A new hierarchical clustering algorithm based on the principle of the minimum description length (MDL) is presented in [9].

The authors demonstrate that the proposed solution is much better than k-means or EM. In this paper, we propose to refine the algorithm, called EXDBSCAN, to cover data sets with multiple densities (multi-density), where the classical algorithm is inefficient, since it is not able to identify clusters of different density in the data set. This method only receives one parameter from the user, and besides, it correctly detects clusters with different density.

In the Internet there is a very important task of analyzing and optimizing the requests of visitors to sites or users of search engines.

In [10], to solve this problem, it is proposed to use Suffix Tree Clustering (STC) method based on the construction of a suffix clustering tree (Su-x). The tree is formed based on the selected keywords and word combinations from the documents.

In this paper, the authors solve the problem of mass clustering of short texts, whose vectors are by definition very sparse, and clustering methods based on vector space face difficulties in processing matrixes of high dimensionality. The authors distinguished two different characteristics, in the tree document model and the STC algorithm.

First, the document model proposes a new approach to identify all overlapping nodes (phrases) Second, one or more phrases are selected in such a way as to generate a summary of the topic to indicate the corresponding cluster when they are created.

The authors of [10] found that the algorithm of the STC method can show good results when clustering short texts, as well as fragments of the document. However, the STC algorithm sometimes generates some poor quality large-scale clusters when clustering standard documents.

These results reduce the overall effectiveness of the STC algorithm. In the opinion of the authors of [10], this problem can be solved by conducting a measure of the similarity of the Su-x tree.

A brief literature review shows that at the present time there is no clustering model that at the same time would have short execution time, would have absolute accuracy and would process large amounts of unstructured or semi-structured data.

Therefore, the development of new clustering algorithms is very urgent.

3 The proposed method for clustering text documents

3.1 Description of the proposed clustering algorithm

We propose the following clustering algorithm, which at the stage of creating the matrix: the term – document, match with the existing algorithms.

1. From the collection of N-text documents, using the dictionary containing M terms (words and n-grams), using the TF-IDF technique, we create the matrix L: term-document, which have N-columns and M-rows.

$$L_i = \begin{vmatrix} l_{1,i} \\ l_{2,i} \\ \cdot \\ l_{k,i} \\ \cdot \\ l_{M,i} \end{vmatrix}, \quad (1)$$

where $l_{j,i}$ —is the characteristic of the occurance of the j-th term in the i-th document.

For each document i of the collection N will have corresponding vector L_i:

The creation of vectors can be carried out in parallel taking into account the number of processors and cores of the computer system.

2. Find the position of the center (centroid) of the vector space of the document collection (the center of the information space), as the average arithmetic vector of all vectors:

$$L_c = \begin{vmatrix} \frac{\sum_{i=1}^{N} l_{1,i}}{N} \\ \frac{\sum_{i=1}^{N} l_{2,i}}{N} \\ \cdot \\ \frac{\sum_{i=1}^{N} l_{k,i}}{N} \\ \cdot \\ \cdot \\ \frac{\sum_{i=1}^{N} l_{M,i}}{N} \end{vmatrix}. \quad (2)$$

3. For each collection vector in a multidimensional space (RM), calculate the Euclidean distance (geometric distance) between the given vector and the vector that determines the position of the centroid:

$$d_i(l_{j,i}, L_C) = \sqrt{\sum_{j=1}^{M} \left\{ l_{j,i} - \frac{\sum_{i=1}^{N} l_{j,i}}{N} \right\}^2}. \quad (3)$$

4. We sort all the $d_i(l_{j,i}, L_C)$ from 0 to $max\{d_i(l_{j,i}, L_C)\}$ and divide the numerical sequences obtained for $d_i(l_{j,i}, L_C)$into subgroups of the size rξ:

$$0 \le r_1 < \xi \cdot d_i(l_{j,i}, L_C)$$
$$\xi \cdot d_i(l_{j,i}, L_C) \le r_2 < 2\xi \cdot d_i(l_{j,i}, L_C)$$
$$2\xi \cdot d_i(l_{j,i}, L_C) \le r_3 < 3\xi \cdot d_i(l_{j,i}, L_C)$$
……..
$$(m-1)\xi \cdot d_i(l_{j,i}, L_C) \le r_m \le max\{d_i(l_{j,i}, L_C)\} \quad (4)$$

We denote the number of subgroups as m. The definition of the quantity ξ (we call it the clarity parameter of clustering) will be discussed later. In each of the m subgroups we place the vectors $L_{i,p}$ that have entered it (let their number in each of the subgroups equal to some value of Sm):

$$L_{i,p} = \begin{vmatrix} l_{1,(i,p)} \\ l_{2,(i,p)} \\ \cdot \\ l_{k,(i,p)} \\ \cdot \\ l_{M,(i,p)} \end{vmatrix} \quad (5)$$

where p is the index (number) of the subgroup (p takes values from 1 to m), i is the number of the vector (document) in the collection N.

The algorithm for vector clustering within each of the subgroups m will be the same, and will not depend on the vectors available in other subgroups, so clustering can be carried out in all m subgroups in parallel, taking into account the number of processors and cores of the computer system.

5. Consider, for example, the first subgroup (m = 1). We choose any vector in it (for example, the first $L_{i,1}^{(1)}$) and find the Euclidean distance between this vector and each of the other vectors of the given subgroup:

$$d_{m=1}^{(1)}(L_{i,1}^{(1)}, L_{i,1}^{(q)}) = \sqrt{\sum_{q=2}^{S_m} \left\{ L_{i,1}^{(1)} - L_{i,1}^{(q)} \right\}^2} \quad (6)$$

From the whole set of vectors of a subgroup we choose vectors for which the following condition is satisfied:

$$d_{m=1}^{(1)}(L_{i,1}^{(1)}, L_{i,1}^{(q)}) \le \xi \cdot |L_{i,1}^{(1)}|, \quad (7)$$

where $|L_{i,1}^{(1)}|$ – is the length of a vector $L_{i,1}^{(1)}$

All vectors which satisfying this condition are placed in one cluster. If there is not a single vector for which the given condition was fulfilled, then the cluster consists of only one first vector $(L_{i,1}^{(1)})$.

6. We choose the second vector $(L_{i,1}^{(2)})$ from the vectors not included in the first cluster of the given subgroup, and find the Euclidean distance between this vector and each of the other vectors of the given subgroup (including those that could already be in the first cluster of this subgroup), with the exception of the first vector (they were already considered in pairs):

$$d_{m=1}^{(2)}(L_{i,1}^{(2)}, L_{i,1}^{(q)}) = \sqrt{\sum_{q=3}^{S_m} \left\{ L_{i,1}^{(2)} - L_{i,1}^{(q)} \right\}^2} \quad (8)$$

From the whole set of vectors of a subgroup we choose vectors for which the following condition is satisfied:

$$d_{m=1}^{(2)}(L_{i,1}^{(2)}, L_{i,1}^{(q)}) \le \xi \cdot |L_{i,1}^{(2)}|, \quad (9)$$

where $|L_{i,1}^{(2)}|$ – is the length of vector $L_{i,1}^{(2)}$

All vectors satisfying this condition are placed in one cluster. If there is not a single vector for which this condition was fulfilled, then the cluster consists of only one second vector $(L_{i,1}^{(2)})$.

7. Next, we repeat step № 6 for all vectors that have not been considered. When performing this stage of clustering, it may turn out that some vectors turn out to belong to different clusters.

8. We find centroid for each of the clusters obtained in the steps 5–6 (its coordinates are calculated in the standard way, as the average arithmetic vector of all the vectors of the cluster).
9. For each of the clusters subgroups, we find Euclidean distances (we denote each of them for brevity as d, without considering the membership indices, to any cluster) between its centroid and each of the vectors of the given cluster. The value of the centroid, for brevity, we denote as R, without considering the membership indices, to any cluster. Next, in each of the clusters, we check that each of its vectors satisfies the condition: $d \leq \xi R$. We remove vectors that do not satisfy this condition from clusters. If the removed vector was included in several clusters, then we leave it in those of them for which this condition was fulfilled, and remove from further consideration.
10. We calculate new centroids for the clusters obtained at step 9. Then, we find the Euclidean distances between each of the centroids and unit vectors which have not entered any clusters. The vectors for which the condition is satisfied: $d \leq \xi R$, is added to this cluster.
11. We calculate new centroids for the clusters obtained at step 10. Further, we find Euclidean distances between each of the centroids, and unit vectors which have not entered any clusters. The vectors for which the condition is satisfied: $d \leq \xi R$, is added to this cluster. Repeat this step until the vectors belong to the clusters is changing. Note that there may exist clusters containing only one vector. Steps 5–11 can be performed in parallel for each of the m subgroups, taking into account the number of processors and cores of the computer system.
12. Further, it is necessary to check whether clusters located near the boundaries of each of the m subgroups enter into one common boundary cluster. For example, a common cluster can form clusters located on the boundary of the first (m = 1) and second (m = 2) subgroups, or the second and third, etc. Note that this operation can also be performed in parallel taking into account the number of processors and cores of the computer system.
13. We take the centroids of the first (m = 1) and the second (m = 2) subgroups. We select any of the centroids in the first subgroup and find the Euclidean distance between this vector and each of the vectors of the centroids of the second subgroup (we denote each of these Euclidean distances for brevity as d, without considering the membership indices, to any cluster). We check in each pair the fulfillment of the condition: $d \leq \xi R$ (where R is the length of the centroid vector selected from the first subgroup). For a pair of centroids, for which this condition is fulfilled, conditionally combine their clusters into a single new cluster.
14. We calculate the centroid for the conditional cluster obtained at step 13. We find Euclidean distances (we denote each of them for brevity as d, without considering the membership indices, to any cluster) between its centroid and each of the vectors of the given cluster. For brevity, we denote the centroid value as R (without considering the membership indices, to any cluster). We check for each of its vectors the following condition: $d \leq \xi R$. If this condition is fulfilled for all vectors, then we join the clusters into one common one. If this condition is not filfilled, then we consider the belonging of each of the vectors of one of the clusters of the given pair to the second participant of the pair. Any of the vectors can simultaneously enter into different clusters, i.e. we have two different intersecting clusters. For this, we find the Euclidean distances between the vectors of one cluster and the centroid of the second, we check them for the condition $d \leq \xi R$ (R is the length of the centroid). The vectors for which this condition is fulfilled are added to another cluster. Next, after the addition of vectors, we calculate the position of the new centroids and check the condition that some vectors do not drop out of the new cluster. If the vectors do not drop out, then we accept the addition condition. If any vectors drop out, then we do not accept the addition condition, and the vector remains only in its cluster.
15. In step 13 it may turn out that there are several pairs of the same centroid from the first subgroup, for which the condition: $d \leq \xi R$ (where R is the length of the centroid chosen from the first subgroup) is satisfied. Then we consider each pair separately using step No. 14, and after merging the two pairs, if there are still pairs that satisfy the merger condition for the same centroid from the first subgroup, then repeat steps 13–15 to merge the clusters until the membership of the vectors to the cluster will not cease to change.
16. We select the second centroid in the first subgroup and repeat steps 13 to 15 until the vectors belong to the clusters will not cease to change.
17. Repeat steps 13–16 for all the remaining centroids of the first subgroup.
18. We now turn to the consideration of the centroids of the second (m = 2) and third (m = 3) subgroups, and repeat steps 13 to 17. Next, we turn to the centroid of the third (m = 4) and fourth (m = 4) subgroups, and furher until all the subgroups are considered.

3.2. The clarity parameter of clustering (ξ)

The clarity parameter of clusterization (ξ) characterizes the semantic difference (or similarity) of textual information. To determine the value of the clarity parameter of clusterization (ξ), we use the notion of redundancy, which is used in information theory.

Redundancy – is exceeding the amount of information used to transmit or store a message over its information entropy. According to one of the definitions, redundancy is a measure of uselessly performed alternative choices when reading the text.

It shows how much extra information is contained in the texts of the given language, and can be restored without an explicit indication in alphabetic form. For example, in telegraphic texts the abbreviations "ZPT"

and "TCHK" can be used instead of full words without a loss of meaning, conjunctions and prepositions are excluded from the texts of the proposals. Human easily perceives the meaning of words with missed letters etc. The presence of redundancy allows the reduction of texts without loss of their semantic content.

Redundancy determined by the structure of the language used, and its numerical value can be used to determine the boundary of loss of meaning contained in textual information. Text information can be represented by a message consisting of some sequence of characters that are the alphabet of the given language.

In the simplest case, redundancy can be defined as follows. Let's consider messages consisting of one sign of the alphabet. We first find the information, which is an average of one sign of the alphabet. To do this, you can imagine the appearance of any sign of the alphabet as an event, the probability of which outcomes (the probability of occurrence of a particular letter in the text) are the same.

According to the theory of information entropy, the uncertainty introduced by each of the K equiprobable outcomes is:

$$H = -\sum_{n=1}^{K} p_n \, log_2 p_n,$$

where p_n is the probability of any of the K equiprobable outcomes. With this approach, one sign of the Russian alphabet on average should be:

$$H_0 = -\sum_{n=1}^{K} p_n \, log_2 p_n = -\frac{34}{34} log_2 \frac{1}{34} = 5{,}087$$

bits of information. The value of 34 is used because the Russian alphabet contains 33 letters and, in addition, it is necessary to take into account the gaps between words, which will also be a symbol of the alphabet. However, in reality, the probability of the appearance of letters in the text is different. For the Russian alphabet, these probabilities are presented in Table 1.

Using the probability of occurrence of letters in a message (see Table 1), we can find information obtained from a message source with one sign:

$$H_1 = -\sum_{n=1}^{K} p_n \, log_2 p_n = 4{,}350 \text{ bits.}$$

According to the definition of redundancy, for messages consisting of one character of the alphabet, redundancy can be calculated by the following formula: $D_1 = (H_0 - H_1)/H_0 = (5.0876\text{-}4.350)/5.087 = 0.13$.

In this case, the clustering clarity parameter (ξ) can be specified as follows. If ξ≤0,13 text documents should have the same semantic value, with ξ> 0,13 - the text documents will have different meanings.

In reality, it is necessary to take into account that messages have longer length than one character, i.e. it is necessary to talk about the average length of messages (words) for the Russian language. For two characters, H_2= 3.52 bits; for three, H_3= 3.01 bits; for an infinitely long word $H_\infty \rightarrow 1$ bit.

According to Leonid Delitsyn's statistical research, the average word length in Russian is about 6 characters, but depends on the body (set) of texts. For the body of Russian colloquial speech, the average word length will usually be from 3.9 to 4.9 letters. For the body of Russian fiction – from 4.9 to 5.9. For the news-news corps – from 5.9 to 6.9. For the body of scientific and business literature – from 6.9 to 7.9 letters. If the body is made so that each of the four specified types (functional styles) entered equally (for example, for a million words), then the average word will be 6 letters (5.9). Given that scientific and business speech is less common, you can reduce their share. For example, if you reduce to zero, the average word length will be reduced to 5.5 letters.

Table 1. The probability of the appearance of symbols for texts in Russian

Letter	**Space**	**О**	**Е, Ё**	**А**
Relative frequency	0.175	0.090	0.072	0.062
Letter	**И**	**Т**	**Н**	**С**
Relative frequency	0.062	0.053	0.053	0.045
Letter	**Р**	**В**	**Л**	**К**
Relative frequency	0.040	0.038	0.035	0.028
Letter	**М**	**Д**	**П**	**У**
Relative frequency	0.026	0.025	0.023	0.021
Letter	**Я**	**Ы**	**З**	**Ь. Ъ**
Relative frequency	0.018	0.016	0.016	0.014
Letter	**Б**	**Г**	**Ч**	**Й**
Relative frequency	0.014	0.013	0.012	0.010
Letter	**Х**	**Ж**	**Ю**	**Ш**
Relative frequency	0.009	0.007	0.006	0.006
Letter	**Ц**	**Щ**	**Э**	**Ф**
Relative frequency	0.004	0.003	0.003	0.002

An experimental study of the effect of the clarity parameter on the clustering of texts can be carried out using the following algorithm:

1. Take a statistically significant (for example, N = 100000) sample of text documents (for example, news reports for any year).
2. Then we vectorize text documents.
3. Set the value ξ = 0.01, and clusterize the vectors according to the algorithm described earlier. Determine the number of resulting clusters (C) and calculate the value F = C / N.
4. Change ξ by Δξ = 0.01 and repeat step 3 until ξ is equal to 1.
5. We plot the dependence of F on the value of ξ.

4 Discussion

In one article, it is impossible to compare developed model of text clustering with all existing approaches and algorithms

However, analysis shows that all existing algorithms have both advantages and disadvantages. In particular, for example, the algorithm can be fast and scalable, but you must specify the initial number of clusters for which you need to divide a set of texts, and the result of clustering will depend on the initialization phase of the algorithm and be poorly reproducible in the structure of the resulting clusters. Or, on the contrary, to have a high quality of clustering, but to have a long execution time and poor scalability, to be sensitive to the initial data, to work badly with sparse or dense vector spaces, etc.

Unfortunately, the disadvantages and advantages of existing algorithms do not intersect, this leads to the need to find ways to combine several algorithms in one

text clustering method . However, in our opinion, it is more promising to continue searching for an algorithm which could combine advantages and would be deprived of the listed disadvantages. Proposed algorithm does not need to specify in advance the number of clusters and the position of their centers, so the result of clustering will not depend on the initialization stage. The algorithm does not depend on the density of the location of objects, does not have a high complexity, which should affect.

5 Conclusion

According to our assumptions, the proposed algorithm should have good speed and accuracy of text clustering, depending on the value of the clarity parameter of clustering. This parameter can be determined for texts of different thematic groups experimentally or given by selection. The advantage of this algorithm is that its operation does not require the initial set of clusters.

This work was supported by the Ministry of Education and Science of the Russian Federation [grant number 28.2635.2017/ПЧ, under project title “Development of stochastic dynamics models with memory and self-organization for weakly structured information, to forecast news events based on natural language textual arrays”].

References

1. M. Śmieja, Ł Struski., J. Tabor, Expert Systems with Applications, 85, 146 (2017)
2. E.N.C. Wai, P.-W. Tsai, J.-S. Pan, Advances in Intelligent Systems and Computing, **536**, 36 (2017)
3. Y. Zhang, J. Mańdziuk, C.H. Quek, B.W. Goh Information Sciences, **415/416**, 414 (2017)
4. Y. Zhang, G. Geng, X. Wei, C. Shi, S. Zhang, Xi'an Dianzi Keji Daxue Xuebao/Journal of Xidian University, **44**, 114 (2017)
5. L. Yaohui, M. Zhengming, Y. Fang, Knowledge-Based Systems, **133**, 208 (2017)
6. D. Agnihotri, K. Verma, P. Tripathi, Expert Systems with Applications, **81**, 268 (2017)
7. V. Radhakrishna, C. Srinivas, C.V. Gururao, A modified Gaussian similarity measure for clustering software components and documents, *In Proceedings of the International Conference on Information Systems and Design of Communication* pp. 99-104 (2014)
8. C.M. De Vries, L. De Vine, S. Geva, , R. Nayak Parallel streaming signature EM-Tree: A clustering algorithm for web scale applications, *In WWW 2015 Proceedings of the 24th International Conference on World Wide Web*, pp. 216-226 (2015)
9. A. Ghanbarpour, B. Minaei, EXDBSCAN: An extension of DBSCAN to detect clusters in multi-density datasets, *In 2014 Iranian Conference on Intelligent Systems, ICIS 2014*, article number 6802561 (2014)
10. M. Peng, J. Huang, J. Zhu, J. Huang, J. Liu, Jisuanji Yanjiu yu Fazhan/Computer Research and Development, **52**, 1941 (2015)