Accredited by National Journal Accreditation (ARJUNA) Managed by
Ministry of Research, Technology, and Higher Education, Republic Indonesia with Second Grade (Peringkat 2, Sinta 2)
since year 2017 to 2021 according to the decree No. 10/E/KPT/2019.

Published online on the journal's webpage: **http://jurnal.iaii.or.id**

RESTI JOURNAL
(System Engineering and Information Technology)
Vol. 5 No. 2 (2021) 359 - 368 ISSN Electronic Media: 2580-0760

High Scalability Document Clustering Algorithm Based On Top-K Weighted Closed Frequent Itemsets

Gede Aditra Pradnyana[1], Arif Djunaidy[2]
[1]Informatics Department, Faculty of Engineering and Vocational, Universitas Pendidikan Ganesha
[2]Information System Department, Faculty of Intellegent Electrical and Informatics Technology,
Institut Teknologi Sepuluh Nopember
[1]gede.aditra@undiksha.ac.id, [2]adjunaidy@its.ac.id

Abstract

Documents clustering based on frequent itemsets can be regarded a new method of documents clustering which is aimed to overcome curse of dimensionality of items produced by documents being clustered. The Maximum Capturing (MC) technique is an algorithm of documents clustering based on frequent itemsets that is capable of producing a better clustering quality in compared to other similar algorithms. However, since the maximum capturing technique employed frequent itemsets, it still suffers from such several weaknesses as the emergence of items redundancy that may still cause curse of dimensionality, difficult to determine the minimum support value from a set of documents to be clustered, and no weighting on items incurred to the resulting frequent itemsets. To cope with those various weaknesses, in this research, an algorithm of documents clustering based on weighted top-k closed frequent itemsets, which is called as Weighted Maximum Capturing (WMC) algorithm, is developed. The proposed algorithm involves the frequent pattern tree algorithm to mine closed frequent itemsets from a set of documents without specifying the minimum support value of items to be generated. Experimental results showed that improvement on the resulting clustering accuracy was produced. The resulting average values of F-measure of 0.713 and purity of 0.721 with improvement ratio of 1.4% for F-measure and 2% for purity. Nevertheless, results of the scalability test showed very significant improvement. The WMC algorithm only requires the average computing time of 623.77 minutes, 518.05 minutes faster than the average computing time required by the MC algorithm.

Keywords: documents clustering, frequent itemsets, weighted maximum capturing, top-k closed frequent itemsets

1. Introduction

The increase in the number of documents in text format significantly makes the process of grouping or document clustering becomes important. Zhao and Karypis define clustering is a process that divides a set of objects into the number of groups (clusters) specifically [1]. Document clustering aimed at dividing the document into several groups so that the documents in the same cluster (intra-cluster) have higher similarity, while the documents in the different cluster (inter-cluster) have low similarity [1,2,3].

In general there are two main techniques in clustering process is hierarchical and partitional clustering [2]. Algorithms clustering hierarchical and partitional do not fully meet the challenges in the case of clustering documents, such as: the algorithm is still classifying (high dimensional vector space) fully and centroid or an average of the cluster that forms do not provide a description which is understandable on the cluster , In the vector space model, the use of words that stand alone (individual words) will causes a higher-dimensional vectors. On the other side is actually not all documents in the collection also contains all the words used in the index vector. These problems drive the development of new methods in clustering documents that are not based on the use of the vector space model [4].

Frequent itemsets appear to address this issue. Some researchers have applied the concept of frequent itemsets in document clustering [4,5,6,7]. Started by Beil who used frequent items to represent a candidate of a cluster [4]. Fung introduces the use of global frequent itemsets and frequent itemsets cluster for later use in the process of hierarchical clustering documents [5]. Research conducted by Li added aspect of the order (sequence) of a frequent itemsets items to be used in clustering documents [6]. From experiments conducted on the above research, the quality of the clustering with the concept of frequent itemsets better than conventional clustering algorithms such as K-means, bisecting K-

Accepted by Editor: 27-02-2021 | Final Revision: 10-04-2021 | Online Publication: 30-04-2021

means and UPGMA. The results of the above studies also prove that the concept of frequent itemsets used can perform dimension reduction with and also can increasing efficiency and scalability of document clustering process. Furthermore, Zhang propose a method of text clustering frequent itemsets that proved to be better than methods based frequent itemsets before in terms of the accuracy of the results of clustering [7]. The proposed method uses the technique of capturing maximum in forming clusters of frequent itemsets there. In the document clustering process, the method proposed by Zhang have also adapted the minimum spanning tree algorithm into partitional clustering method [7].

Document clustering based on frequent itemsets with maximum capturing technique still has some drawbacks. First, the maximum capturing method does not pay attention to the weight of words (items) at the time of multiplication frequent itemsets and measurement of similarity between documents in forming a cluster. In the process of clusters formation, the similarity of a document with other documents in the maximum capturing method is only measured by the number of same frequent itemsets in the document. The more number of same frequent itemsets in one document to another document, so the document will have higher similarity. A frequent itemsets said to be equal if they have the same item without taking into account the weight of these items, although this weighting will greatly affect the similarity of documents. This will certainly reduce the accuracy of the clustering for a word that often appears in a document should have greater weight than words that rarely appear. Secondly, the adaptation of the minimum spanning tree method into the cluster formation process is less precise, because it does not pay attention to global similarity when updating the newly formed cluster similarity with another cluster. Research conducted by Pradnyana and Djuanaidy developed a method called Weighted Maximum Capturing (WMC) attempted to overcome this problem. Weighted maximum capturing method pay attention to the weight of frequent itemsets between documents [8]. Similarity of frequent itemsets between documents is measured using a method which is the combination between the Jaccard coefficient as a method asymetrical binary similarity with cosine similarity method. Jaccard similarity coefficient is used to view the number of same frequent itemsets possessed in a pair of documents, while the cosine similarity method used to calculate the similarity between frequent itemsets based on the weight of each item in the itemset. From the research results proved that the developed method has a better accuracy results than the maximum capturing method.

Previous research conducted by Pradnyana and Djunaidy focuses only on the cluster formation process improvement [8]. The process of searching frequent itemsets of a document can generate itemsets in a very large number and often occur redundancy. This will certainly cause scalability problems of clustering method because of high dimensional representation of the document. On existing research also found that a large number of frequent itemsets will have a value similar to parent support of the itemset. Therefore, frequent itemsets certainly would not be very significant if it is used in the process of document clustering. When searching for frequent itemsets, the value of the minimum support (minsup) set by the user will greatly affect the results and the number of frequent itemsets are found. This will affect the scalability and accuracy of the document clutering process. Without specific knowledge of the documents to be grouped, the user will have difficulty in establishing values minsup to produce the right results clustering. If the value of minsup set too large will produce frequent itemsets in small amounts or even not produced any frequent itemsets. Frequent itemsets that too little will only provide information that is too abstract or may not include all the information contained in those documents. In this case, users generally will set the value minsup small and then mine back that might give better results. However, choosing minsup value that is too small may result in frequent itemsets generated number is too large and will increase costs and more computing time.

This research propose a new method of document clustering process using frequent closed itemsets which is the development of a method of WMC. This new method perform document clustering process with the use of top-k weighted closed frequent itemsets. The method is expected to improve the accuracy and scalability of the previous method with the use of the concept of top-k closed frequent itemsets and frequent itemsets excavation without limitation the determination of minimum support (minsup). In practice, the number of frequent itemsets obtained from the transaction data may be very large, it is helpful to obtain a concise itemset representation of frequent itemsets where more can be derived from this concise representation. Closed itemset is a concise representation of itemset without loss of information support of the subset [9]. An itemset X is said to be closed if there is no immediate superset which has a support value similar to X. In other words, X is not closed if there is at least one superset immediate support having a value similar to X. Using top-k frequent closed itemsets to represent a document which is a concise representation of frequent itemsets. The use of closed frequent itemsets is expected to address the issue of redundancy and height dimensions of the use of frequent itemsets so that scalability can be improved. Minimum support (minsup) value is a value that is very influential on extracting frequent itemsets, but very difficult to determine because it depends on the characteristics of the database and the absence of clear boundaries. The method developed at this research adapting closed node count array and descendant sum contained in TFP

DOI: https://doi.org/10.29207/resti.v5i2.2987

algorithm to generate minsup dynamically during the mining process closed frequent itemsets. Without the determination minsup it is expected that the accuracy of this method is better than the previous method.

2. Research Method

In this section described the process flow used in the proposed algorithm. This section provides a description of what and how the proposed algorithm is applied. The design of the the proposed algorithm in this study can be divided into two main algorithms, ie top-k closed frequent itemsets mining algorithms of a collection of documents in order to overcome the problem of scalability and algorithm of cluster documents creation based on weighted top-k closed frequent itemsets to improve the accuracy of the results of clustering.

2.1. Mining Top-K Closed Frequent Itemsets Algorithm for Document Clustering

In this study, TFP algorithm applied in the mining process of top-k closed frequent itemsets of a collection of documents without the determination minsup. TFP algorithm proposed by Wang is generally used on a transactional database. Therefore, in this study, there are several processes performed at the TFP algorithms to be used in the process of clustering documents effectively [10]. The overall flow of the process of extracting frequent top-k closed itemsets of a set of documents using the algorithm TFP can be seen in Figure 1.

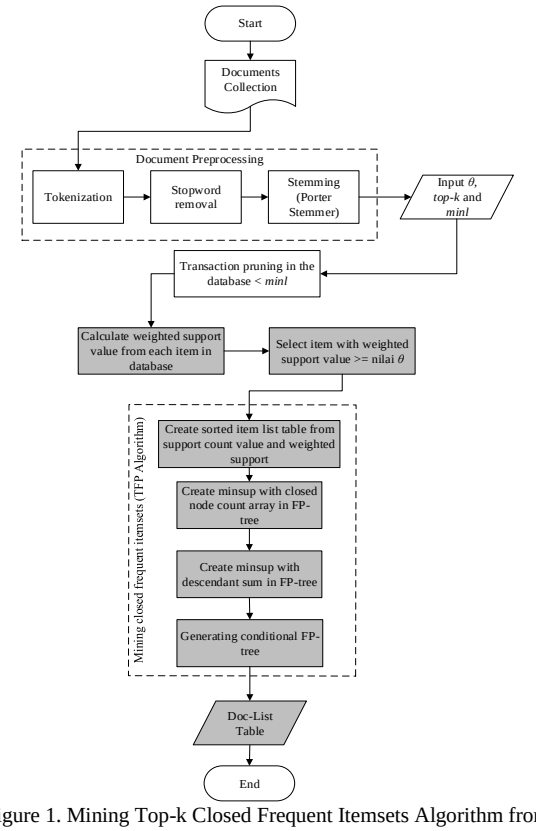

Figure 1. Mining Top-k Closed Frequent Itemsets Algorithm from Documents Collection

As clearly shown in Figure 1, the process of extracting top-k closed frequent itemsets consists of two main stages, namely documents preprocessing stage and the stage of mining top-k closed frequent itemsets. Document preprocessing stage consists of several processes as follows: (a) Stopwords Removal : the removal of the words are not important (stopwords) from a document. The removal process is done by matching words in the document against a table that contains a list of words are not important (stoplist). (b) Stemming : the process returns to the basic form of a word (root word). In this research used Porter Stemmer algorithm in finding the base word (root word) of a word.

The next stage is mining process of top-k closed frequent itemsets using the TFP algorithm. In mining closed frequent itemsets of a collection of documents, there are several terms used, that is a word to represent an item and a document will represent a transaction. Top-k closed frequent itemsets result of this mining process will be stored in Doc-List table. Doc-List table is a table containing the documents and their frequent closed itemsets contained in the document. The steps in mining the top-k closed frequent itemsets can be explained as follows:

a. Determination of initial parameter values. Parameters that are used in the process of extracting include a threshold value *θ*, *minl*, and *top-k*.
b. Pruning documents that do not meet minl value. Documents with the number of types of words (unique word) that is less than minl will be removed from the database and are not included in the next process.
c. Calculation of weighted support and the support of each word count. Support count is the number of documents that contain the word in the document in the database. The calculation of the value of the support of each word (item) in the document are calculated using the following equation:

$$\text{Support(i)} = \frac{\sum_{t=1}^{n} \frac{tf_{i,t}}{length_t}}{n} \quad (1)$$

In equation (1), the variable *support (i)*, *n*, $tf_{i,t}$ and *length* respectively represent the value of the support the weighted of item *i*, the number of documents, frequency kemuculan of item *i* in document all *t*, and total frequency of items contained in document all *t*. Table 1 shows an example of the preprocessing of five documents. In this table, the values in the column represents the number of times an item or items and their word frequency of occurrence in a document. With the calculation of the weighted support, a word that the high frequency of occurrence in some documents but rarely appears in other documents will be taken into account in the mining process. Instead word that has the opposite properties can be eliminated in the mining process.

DOI: https://doi.org/10.29207/resti.v5i2.2987

Table 1. Example Result from Document Preprocessing

Document Id	Item Frequency
Dok1	*d:15, a:10, e:8, f:7, g:9*
Dok2	*b:13, a:10, g:7*
Dok3	*a:17, b:10, c:13, d:5. e:7, f:8*
Dok4	*a:9, d:6, f:4, c:16*
Dok5	*e:20, f:15, g:3, a:10, b:5, d:2*

Based on the example shown in Table 1, the value of the weighted support of each item can be calculated as follows. Complete results of each item can be seen in Table 2.

$$a = \frac{\frac{10}{45}+\frac{10}{30}+\frac{17}{60}+\frac{9}{35}+\frac{10}{55}}{5} = \frac{0,22 + 0,33 + 0,28 + 0,26 + 0,18}{5} = \frac{1,27}{5}$$

$$= 0,254$$

Table 2.Weighted Support of Each Item

Item	Weighted Support
a	0,254
b	0,138
c	0,136
d	0,124
e	0,130
f	0,132
g	0,078

Pruning a word with weighted support less than the value of θ. Based on the weighted value of the support that has been obtained will have a word with weighted support value greater than or equal to the threshold value θ, to be added to the sorted item list table. Sorted item list table is a table containing the word with weighted support value greater than or equal to the threshold value θ, sorted in descending order based on the value of support count. For example those described in section (c) above, if it is determined the value of θ = 0.1 (10%) then the word (item) g will be removed from the list of items sorted table for support terbobotnya value smaller than the value of the parameter θ. In this case, although the item g actually appear in more elsewhere (3 documents) compared to occurrences item c (2 documents), but the frequency of occurrence of item g in those documents is very low and not significant when compared with the frequency of appearance items c, so that the value of the support weighted item c higher than item g.

Table 3. Sorted Item List

Document Id	Sorted Item
Dok1	*a, f, d, e*
Dok2	*a, b*
Dok3	*a, f, d, b, e, c*
Dok4	*a, f, d, c*
Dok5	*a, f, d,b, e*

In the process of sorting the items at the time of making sorted item list, TFP algorithms generally do not pay attention to the existence of some of the items that have the same support count. In the the proposed algorithm in this study, if there are multiple items that have the same value of support count, then the order of the item will be determined based on the weighted value of its support. Thus, for example described previously will be obtained sorted items in the order {a, f, d, b, e, c}. Based on that sorted items, then be prepared sorted item list table as shown in Table 3.

d. Mining top-k frequent closed itemsets. After the sorted items list table is formed, the next process is the mining process of top-k closed frequent itemsets using the TFP algorithm. In the mining process, TFP algorithm using FP-tree structure that builds based on the sorted item list table. Minsup generation process during the formation of the FP-tree is done using a closed node count array method, on the other hand when the FP-tree has been formed minsup generated using descendant sum method. The results of the mining process by using TFP algorithms is a frequent closed itemsets list of documents that meet the parameter *θ, minl*, and *top-k.*

e. Doc-List table creation. The next process is the process of making Doc-List based on the list of closed frequent itemsets that was gathered. Doc-List table prepared by matching the items contained in each closed frequent itemsets generated from the process of extracting the items owned by a document. Doc-List produced will then be used in the formation of clusters of documents. Examples of Doc-List table resulting from this process are shown in Table 4.

Table 4. Example of Doc-List from Document Collection

Document Id	*Closed Frequent Itemsets*
Dok1	*{I_1: 5 , I_3: 2, I_4: 3}, { I_2: 4, I_5: 3}*
Dok2	*{I_1 : 4, I_3: 3, I_4: 4}, { I_1: 4, I_2: 5}, { I_2: 5, I_5: 8}*
Dok3	*{I_2 : 3, I_4: 5, I_5: 4}, { I_1: 3, I_2: 5,}*
Dok4	*{ I_2: 4, I_5: 6,} { I_1:3, I_2:4,}*
Dok5	*{I_1: 2 , I_3: 3, I_4: 4}*

2.2. Cluster Formation Algorithm based on Weighted Top-K Closed Frequent Itemsets

After a closed frequent itemsets of each document unearthed and has produced Doc-List table, then performed the document cluster formation process based on the Doc-List obtained. In the process of clusters formation in this study was an improvement on the maximum capturing method which adapts the minimum spanning tree method so that the accuracy of the clustering can be improved. Improvements made to the two main sub-processes of cluster formation process, namely the sub-process of calculating the similarity (similarity) between the documents based on the Doc-List table formed and sub-clusters reconstruction process will be established. The overall flow of the process of the formation of clusters based on the table Doc-List can be seen in Figure 2.

DOI: https://doi.org/10.29207/resti.v5i2.2987

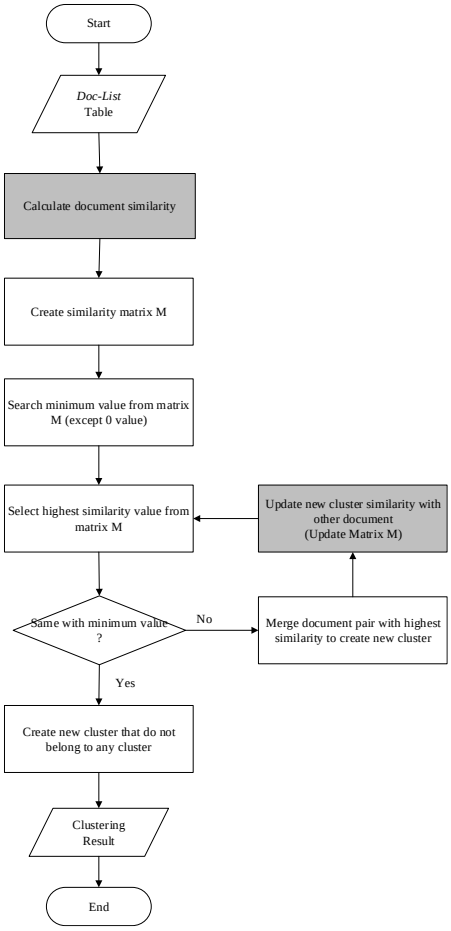

Figure 2. Cluster Formation Proces

As clearly seen in Figure 2, the process of cluster formation based on top-k closed frequent itemsets consists of several stages. Calculation of similarity between documents using a method which is a merger between the Jaccard coefficient as a method asymetrical binary similarity with cosine similarity method. Jaccard similarity coefficient is used to view the number of frequent closed itemsets possessed from the pair of documents, while the cosine similarity method used to calculate the similarity between the closed frequent itemsets based on the weight of each item in the itemset. The weights of the items in an itemset is calculated by the method of TF-IDF. Table 3 shows an example of a Doc-List table from the collection of documents, which is the result from the the mining process. In Table 3, column closed frequent itemsets contains information from the each document along with the frequency of appearance of items from the closed frequent itemsets on the appropriate document.

Based on the example in Table 3, the process of calculating the similarity between documents Dok1 with Dok2 document can be described as follows:

a. The calculation of the weight from each item in closed frequent itemset a document according to the weighting method TF-IDF. For example, the following is a calculation of the weight of the item I1 owned by Dok1:

$$w_{i,j} = tf_{i,j}\ x\ log_2\left(\frac{N}{df_j}+1\right) = 5\ x\ log_2\left(\frac{5}{5}+1\right) = 1{,}51$$

Similar calculations are also carried out on all the items contained in Dok2. The result of the weight calculation from each item contained in Dok1 and Dok2 shown in Table 5.

Table 5. Item Weight in Dok1 and Dok2

Item	Weight in *Dok1*	Weight in *Dok2*
I_1	1,51	1,20
I_2	1,40	1,76
I_3	0,85	1,28
I_4	1,06	1,40
I_5	1,06	2,82

b. The calculation of the similarity between each closed frequent itemsets of each document with cosine similarity method. Based on the examples provided in Table 4 and Table 5, the similarity closed frequent itemsets contained in Dok1 and Dok2 can be calculated as follows:

$$CosSim(Q,D) = \frac{Q \bullet D}{|Q||D|}$$

$$CosSim(Q,D)_{134} = \frac{(1{,}51\ x\ 1{,}20\ +\ 0{,}85\ x\ 1{,}28\ +\ 1{,}06x1{,}40\ )}{\sqrt{1{,}51^2+0{,}85^2+1{,}06^2}\ x\ \sqrt{1{,}20^2+1{,}28^2+1{,}40^2}} = \frac{4{,}384}{2{,}031\ x\ 2{,}245} = \frac{4{,}384}{4{,}56} = 0{,}96$$

$$CosSim(Q,D)_{25} = \frac{(1{,}40\ x\ 1{,}76\ +\ 1{,}06\ x\ 2.82\ )}{\sqrt{1{,}40^2+1{,}06^2}\ x\ \sqrt{1{,}76^2+2{,}82^2}} = \frac{5{,}453}{1{,}756\ x\ 3{,}324} = \frac{5{,}453}{5{,}836} = 0{,}93$$

c. Calculation of similarity between Dok1 and Dok2 with Jaccard coefficient method. Based on the example in Table 5, the similarity between documents Dok1 and document Dok2 can be calculated as follows:

$$Sim(Dok_1, Dok_2) = \frac{Dok_1 \cap Dok_2}{Dok_1 \cup Dok_2} = \frac{0{,}96+0{,}93}{5} = 0{,}378$$

d. Making the similarity matrix M. After all similarity between the documents obtained, the next process is to make the similarity matrix. Similarity matrix M is a matrix that illustrates the value of similarity between documents with one another. Here is an example of the similarity matrix of five documents:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 5 & 4 & 5 & 10 \\ & 0 & 8 & 6 & 9 \\ & & 0 & 7 & 3 \\ & & & 0 & 9 \\ & & & & 0 \end{pmatrix} \end{array}$$

DOI: https://doi.org/10.29207/resti.v5i2.2987

e. Search the minimum value of the matrix M, unless the value 0 The minimum value in the example above matrix is 3.

f. Merger documents with the highest similarity becomes a new cluster. In the example above the matrix M, the highest similarity score is 10, so the document 1 and document 2 can be combined into clusters (1.5).

g. Updates similarity newly formed clusters with other clusters. Similarities new cluster with a cluster is the lowest similarity value between the members of the new cluster. Based on the example of the new cluster formed in step d above, below is an example of the similarity calculating a new cluster with other clusters:

$d_{(15)2} = \min \{d_{12}, d_{52}\} = \min \{5, 9\} = 5$

$d_{(15)3} = \min \{d_{13}, d_{53}\} = \min \{4, 3\} = 3$

$d_{(15)4} = \min \{d_{14}, d_{54}\} = \min \{5, 9\} = 5$

h. Updates similarity matrices M. After the formation of new clusters and similarity between the new cluster with cluster or other document obtained, performed the update process similarity matrices M. Here is an example of the results of updating the matrix M matrix contained in step b above.

	2	3	4	1,5
2	0	8	6	5
3		0	7	3
4			0	5
1,5				0

i. Repeat steps d-f to find the maximum similarity value equal to the minimum similarity value found in step c. From the sample matrix M contained in the above step f, the cluster formation process begins with selection of the biggest similarity value in the matrix M. Selected similarity between documents 2 and 3 document, thus forming a cluster (2,3)

$max\ (d_{ik}) = d_{23} = 8$

Update cluster similarity (2.3) with clusters or other documents.

$d_{(23)4} = \min \{d_{24}, d_{34}\} = \min \{6, 7\} = 6$

$d_{(23)(15)} = \min \{d_{2(15)}, d_{3(15)}\} = \min \{5, 3\} = 3$

After obtaining similarity between the new cluster (2.3) with clusters or other document, followed by the process of updating the matrix M.

	2,3	4	1,5
2,3	0	6	3
4		0	5
1,5			0

After updating the matrix M, the next process is selection of the biggest similarity value in the matrix M above. Selected similarity between clusters (2.3) with 4 document, thus forming a cluster (2,3,4)

$max\ (d_{ik}) = d_{(23)4} = 6$

Updates similarity cluster (2,3,4) with clusters or other documents.

$d_{(234)(15)} = \min \{d_{(23)(15)}, d_{4(15)}\} = \min \{3, 5\} = 3$

Therefore, the maximum similarity value equal to the minimum similarity value, then the cluster formation process is stopped.

j. If there is a document that does not include any clusters, then these documents are used to form a new cluster.

3. Result and Discussion

In this section will be evaluated the performance of document clustering method that was developed by looking at the results of clustering and comparison with previous algorithms. The proposed method is implemented with the Java programming language on a computer with specs Intel Core i7 1.9 GHz, 4GB.

3.1. Test Data

Measurement of the algorithm performance in this study conducted by using test data Reuters 21578 Dataset. This dataset obtained from the UCI Knowledge Discovery in Databases (http://kdd.ics.uci.edu/databases/).

Table 6. Characteristics of *Reuters 21578*

Characteristics	***Reuters 21578***
Item / Word	13.639
Average Documents Length	89
Record	5.000
Category	10

Table 7. *Reuters 21578* Test Data Category

Topic / Category	**Number of Documents**
Acq	1.440
Coffee	119
Crude	379
Earn	1.552
Interest	343
Money-fx	317
Money-supply	108
Ship	216
Sugar	146
Trade	380
Total Document	5.000

DOI: https://doi.org/10.29207/resti.v5i2.2987

Reuters 21578 data set is a set of documents in English which consisted of 21578 documents, compiled by Reuters Newswire in 1987. Table 6 shows the characteristics of Reuters 21578 test data used in this study. While Table 7 shows categories along with the number of documents for each category of Reuters 21578 Dataset.

3.2. Clustering Result Evaluation

For the evaluation of the results of the document clustering process used F-measure and Purity. F-measure is a harmonic combination of recall and precision values used in information retrieval system. Using test data that has been described previously, each cluster produced can be regarded as the result of a query, while each set of documents that have not been classified can be considered as a document which is expected from the query. So the precision value P (i, j) and recall value R (i, j) in each cluster j-th class all i can be calculated.

If ni is the number of class members to-i, nj is the number of members of the cluster j and nij is the number of class members to-i, which are in cluster j, then P (i, j) and R (i , j) can be calculated by equation (2) and (3) below [11]:

$$P(i,j) = \frac{n_{ij}}{n_j}, \quad (2)$$

$$R(i,j) = \frac{n_{ij}}{n_i}. \quad (3)$$

F-measure value of the i-th class in the j-th cluster and the overall F-measure values expressed in the following equations:

$$F(i,j) = \frac{2 \times P(i,j) \times R(i,j)}{P(i,j) + R(i,j)} \quad (4)$$

$$F = \sum_j \frac{n_i}{n} \max \{F(i,j)\}, \quad (5)$$

where n is the number of all documents, ni is the number of documents in class i, and max {F (i, j)} is the value of F (i, j), the largest found in the classroom to-i for the entire cluster j. In general, the value of the high F-measure represents a good clustering results [3].

Purity value of a cluster represents a part of a cluster according to the largest class of a document included in the cluster, the Purity of cluster j is defined as follows [1]:

$$Purity(j) = \frac{1}{n_j} \max_i n_{ij} \quad (6)$$

Purity overall value of the clustering process is the sum of each value Purity cluster:

$$Purity = \sum_j \frac{n_j}{n} Purity\ (j). \quad (7)$$

In general, the higher Purity value show better clustering results.

3.3. Test Results and Analysis

This test aims to evaluate how well the accuracy of the clustering and scalability of the algorithms developed in this study, compared to an algorithm based on frequent itemsets with maximum capturing technique. The accuracy of clustering results obtained by calculating the F-measure and purity. As for the scalability of the algorithm was done by evaluating the effect of the increase of the number of documents to the computation time of the algorithm. To determine the significance of differences in the value of the F-measure and purity of each algorithm performed paired t-test (paired-samples t test) with application software IBM SPSS Statistics 20. The paired t-test was used to compare the mean of the difference between the two values F- measure and purity of each algorithm. In this study, t-test was used paired t-test two sides (two-tailed test). Paired t-test was used to compare the two sides of the distribution of data does have a significant difference or not.

In the process of testing the scalability of algorithms, will use 5,000 documents with the addition of 250 documents in each of its stages. Then observed the computing time of each algorithm in addressing a number of the given document. To determine the significance of differences in the computational time on each of the algorithms will do the same as the t-test on the t-test comparison of the accuracy of the results of clustering.

In measurement accuracy and scalability, the algorithms developed in this study using the value parameter θ = 0.01%, minl = 2, and the top-k = 3300, which is the optimal parameter combination that produces the biggest F-measure value with the value of the smallest CoV. Meanwhile, to determine the optimal minimum support value of the maximum capturing algorithm then tested minsup parameter with a range of values from 0.01 to 0.03 at intervals of 0,005. From the test results, selected parameter value minsup which produces F-measure values highest, 0,015 henceforth use in trials comparing the accuracy and scalability of algorithms. In order to facilitate the writing, to the next algorithm developed in this study is called weighted maximum capturing (WMC). Table 6 shows the results of trials comparing the accuracy of the clustering of each algorithm based on the F-measure and Purity.

The scalability test results of the implementation WMC and MC algorithm respectively can be seen in Figure 3 and Figure 4. Table 9 clearly shows the computation time of the algorithm WMC and MC in addressing a number of the given data. Comparison graph scalability between WMC and MC algorithm in handling a number of given data can be seen in Figure 5.

DOI: https://doi.org/10.29207/resti.v5i2.2987

Table 8. *F-measure* dan *Purity* Value from Document Clustering Result

Number of Documents	F-Measure		Purity	
	MC	WMC	MC	WMC
2.500	0,73	0,738	0,713	0,716
2.750	0,711	0,729	0,729	0,738
3.000	0,675	0,672	0,682	0,671
3.250	0,726	0,742	0,735	0,754
3.500	0,671	0,669	0,677	0,663
3.750	0,662	0,675	0,707	0,725
4.000	0,742	0,747	0,738	0,768
4.250	0,688	0,694	0,692	0,696
4.500	0,722	0,718	0,674	0,722
4.750	0,703	0,721	0,718	0,727
5.000	0,698	0,736	0,711	0,751
Average	**0,703**	**0,713**	**0,707**	**0,721**

Table 9. Computational Time in WMC and MC Algorithm

Number of Documents	MC (minute)	WMC (minute)
2.500	851,617	416,650
2.750	873,183	458,217
3.000	944,767	487,800
3.250	993,983	526,717
3.500	1059,517	557,533
3.750	1078,65	615,483
4.000	1210,817	669,916
4.250	1257,733	733,267
4.500	1314,383	770,700
4.750	1397,450	794,483
5.000	1577,850	830,683
Average	**1141,815**	**623,768**

Based on data from the test results accuracy and scalability in Table 8 and Table 9, further testing data normality by Kolmogorov-Smirnov test before performed t-test. Because all the data are normally distributed, then performed the t-test to determine the significance of differences in the value of the F-measure, purity, and the computing time of MC algorithms and algorithms WMC. The results of the t-test of WMC and MC algorithm based on the F-measure, purity, and the computing time can in Table 10.

Table 10. T-test Result

Pair	t	df	Sig. (2-tailed)
F-Measure MC – F Measure WMC	-2,772	10	,020
Purity MC – Purity WMC	-2,395	10	,038
Comp. Time MC – Comp. Time WMC	18,312	10	,001

In Table 8 can be seen the value of the average F-measure and purity of the clustering process with weighted maximum capturing (WMC) algorithm is greater than the average value of F-measure and purity produced by document clustering process with the maximum capturing (MC) algorithm. From the t-test results are clearly shown in Table 8 of the F-measure value algorithms and algorithms MC WMC obtained p-value is more than the value of significance α (0.02> 0.01). Therefore, the hypothesis H0 is accepted, which means there is no significant difference between the F-measure value of the WMC algorithms than MC algorithm. Where the value of the average (mean) F-measure generated by the algorithm WMC is 0.713 higher at 0.01 compared to MC algorithm (0,703). Although the increase in value is not very significant, however, a higher value indicates that the accuracy of the WMC clustering algorithm is better than the MC algorithms.

Aligned with the value of the F-measure, the average value of purity of the algorithm WMC is also higher than the MC algorithm. From the t-test results are shown in Table 4.11 of the value of purity algorithms and algorithms MC WMC obtained p-value is more than the value of significance α (0.038> 0.01). Therefore, the hypothesis H0 is accepted, which means there is no significant difference between the value of purity algorithm with algorithm MC WMC. Where the value of the average purity produced by WMC algorithm is equal to 0.721 higher than the algorithm MC 0.014 (0.707). High purity value reflects the degree of purity of a cluster is getting better, which means that the cluster contains most of the documents that are supposed to be part of the cluster. To determine the ratio of improvement in the accuracy of clustering results from the WMC algorithm to the MC algorithm, the improvement ratio (IR) is calculated for the F-Measure (IR_F) and Purity (IR_p) values.

$$IR_F = \frac{F^{WMC} - F^{MC}}{F^{MC}} = \frac{0{,}713 - 0{,}703}{0{,}703} = \frac{0{,}01}{0{,}703} = 0{,}014$$

$$IR_p = \frac{Purity^{WMC} - Purity^{MC}}{Purity^{MC}} = \frac{0{,}721 - 0{,}707}{0{,}707} = \frac{0{,}014}{0{,}707} = 0{,}02$$

Based on the IR calculations, WMC algorithm developed in this study make improvements F-measure value of the initial algorithm by 1.4%, and improve the value of the initial algorithm purity of 2%. Changes have been made such as the use of a weighted support as a selection item, document similarity calculation by taking into account the weight of frequent closed itemsets, and the clustering process that takes into account global similarity proven to increase the accuracy of clustering results, although not too significant.

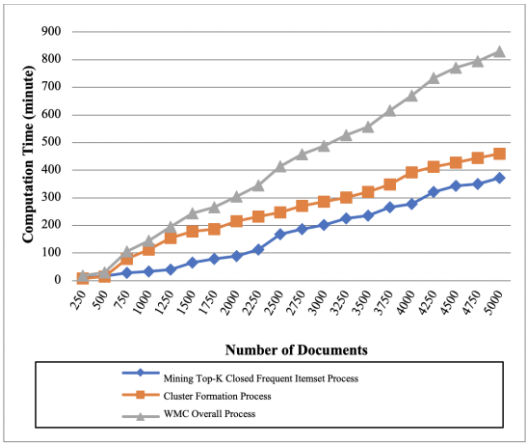

Figure 3. Scalability of Weighted Maximum Capturing Algorithm

DOI: https://doi.org/10.29207/resti.v5i2.2987

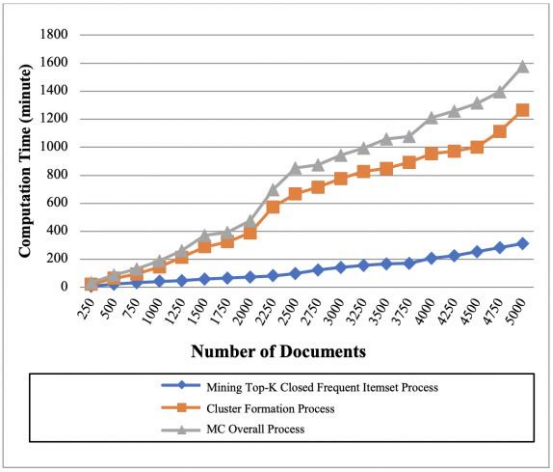

Figure 4. Scalability of Maximum Capturing Algorithm

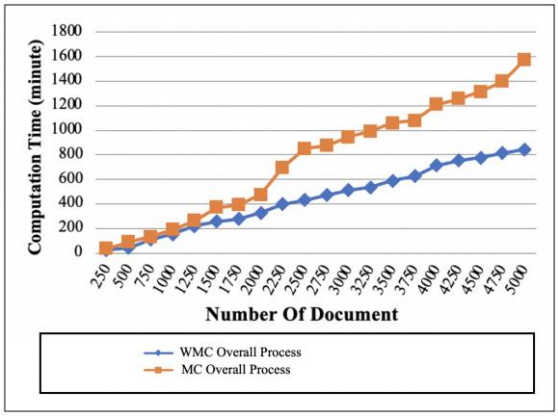

Figure 5. Scalability Algorithm Comparison

From scalability test of the WMC and MC algorithm as shown in Figure 3, Figure 4 and Figure 5 shows that the computing time needed for document clustering process is directly proportional to the number of documents used. The more the number of documents to be grouped then require the longer the computation time as well, and vice versa. In Figure 3 and Figure 4 shows that the computational time on the mining process is shorter than the process of cluster formation on each algorithm. The computing time of MC algorithms are generally smaller than the algorithm WMC at the stage of mining frequent itemsets of the document. Long computation time is due to the process of calculating the weighted support and the weight of each item or word (TF-IDF) on WMC algorithm, where both of these processes require considerable computing time.

From the comparison scalability is performed, as shown in Figure 5, the computation time of the MC algorithm longer when compared with WMC algorithm. This is due to the use of minsup and mining frequent itemsets at MC algorithms that caused higher dimensionality of the representation of a document when the clustering process is done. Determining the value of a static minsup will result in the number of frequent itemsets produced will increase as number of documents increases. In addition, the use of frequent itemsets also led to redundancy in the representation of a document so that it can increase the high dimensionality of a document. On the other hand, the WMC algorithm which was developed in this study, the use of closed frequent itemsets that represent the compact of frequent itemsets, excavation minsup dynamic with the determination of the value of minl as well as top-k, as well as restrictions on the items with the determination of support weighted shown to overcome the problem of high dimensionality from the representation of a document. The reduced dimensions of the document represetation will create a document clustering process used a shorter time than the MC algorithm. Based on t-test results of the computation time of each algorithm contained in Table 9 was obtained p-value less than the significance value α (0.001 <0.05). Therefore, H0 is rejected, which means that there are significant differences between the MC algorithm computing time with WMC algorithm computation time. Where the average computation time of the WMC algorithm is 623.768 which 518.047 minute faster than MC algorithm (1141.815 minutes).

If seen in Figure 5, the addition of WMC algorithm computing time relatively more stable (constant) when compared to MC algorithm, where each additional 250 documents led to an increase in computation time by an average of 42 minutes. Inconsistency changes computational time on MC algorithm due to restrictions in the use of minsup in mining frequent itemsets as a representation of a document. Determination of an good minsup value is largely determined by the number and characteristics of the data. The use of static or constant minsup on the number and type of documents that change causes the computing time is unstable due to the number of frequent itemsets generated change. While at WMC algorithm which was developed in this study, the use of top-k parameter and minl will make the magnitude of the dimensions of the representation of a document to be relatively constant because it will always produce the number of closed frequent itemsets same and not affected the number of existing data.

4. Conclusion

In this research, a new method weighted maximum capturing based document clustering frequent itemsets has been successfully developed. From the results of the tests that have been carried out, it can be concluded that the method proposed in this study does not significantly improve the accuracy of the clustering results. However, the WMC algorithm developed was able to significantly reduce the computation time required to cluster documents. This shows the scalability of the method developed is very good. For Reuters 21578 data test, obtained an average F-measure value of 0.713 (there is an increase of 0.01 or improvement ratio was 1.4%

DOI: https://doi.org/10.29207/resti.v5i2.2987

compared with the comparison algorithm) and the average purity value of 0.721 (there is an increase of 0,014 or improvement ratio was 2% compared with the comparison algorithm). The computing time required by the MC algorithm and algorithm WMC is directly proportional to the number of documents to be grouped. In terms of scalability, WMC algorithm can be said to be better than the comparison algorithm (MC algorithm). The computing time required by the WMC algorithm in handling diverse amount of data tested is always faster than the computation time required by the MC algorithm. From the t-test results for different variations of the amount of data which have been tested show a significant difference in computation time of the two algorithms are compared. For the whole Reuters 21578 test data, the average computation time required by the WMC algorithm is 623.768 minutes (518.047 minutes faster than the computation time required by the algorithm MC).

In future studies, it is highly recommended to try other clustering techniques when forming clusters so that they can be combined with the use of top-k closed frequent itemset. The application of various optimization algorithms such as genetic algorithms and swarm intelligence can also be implemented to increase the accuracy of clustering results.

References

[1] Zhao, Y. & Karypis, G., Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55 (3), pp. 50-62, 2004.

[2] Jain, A.K., Murty, M.N. & Flynn, P.J., Data Clustering : A Review. *ACM Computing Survey*, 31(3), pp. 264-323, 1999.

[3] Steinbach, M., Krypis, G. & Kumar, V., 2000. A Comparison of Document Clustering Techniques. *Proceeding Text Mining Workshop, KDD 2000*.

[4] Beil, F., Ester, m. & Xu, X., Frequent Term-Based Text Clustering. *Proceeding of The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 436-442, 2002.

[5] Fung, B., Wang, K. & Ester, M., Hierarchical Document Clustering using Frequent Itemsets. *Proceeding of The 3rd SIAM International*, 2003.

[6] Li, Y.J., Chung, S.M. & Holt, J.D., Text Document Clustering based on Frequent Word Meaning Sequences. *Data & Knowledge Engineering*, pp. 381-404, 2008.

[7] Zhang, W., Yoshida, T., Tang, X. & Wang, Q., Text Clustering using Frequent Itemsets. *Knowledge-Based System, 23*, pp. 379-388, 2010.

[8] Pradnyana G.A. & Djunaidy, A., Metode Weighted Maximum Capturing untuk Klasterisasi Dokumen Berbasis Frequent Itemsets. *Jurnal Ilmu Komputer Udayana University*, 6 (2), pp. 1-10., 2013.

[9] Tan, P.N., Stainbach, M. & Kumar, V., *Introduction to Data Mining*. 4th ed. New York: Pearson Addison Wesley, 2006.

[10] Wang, J., Han, J., Lu, Y., & Tzvetkov, P., TFP: An efficient algorithm for mining top-k frequent closed itemsets. *IEEE Transactions on Knowledge and Data Engineering*, 17(5), pp.652-663, 2005.

[11] Dalli, A., Adaptation of The *F-measure* to Cluster-Based Lexicon Quality. *Proceedings of the EACL 2003 Workshop on Evaluation Initiatives in Natural Language Processing: are evaluation methods metrics and resources reusable*, pp. 51-56, 2003.

DOI: https://doi.org/10.29207/resti.v5i2.2987