Enhancing an Evolving Tree-based Text Document Visualization Model with Fuzzy *c*-Means Clustering

[1]Wui Lee Chang, [1*]Kai Meng Tay, [2]Chee Peng Lim
[1]Faculty of Engineering, Universiti Malaysia Sarawak, Kota Samarahan, Sarawak, Malaysia.
[2] Centre for Intelligent Systems Research, Deakin University, Australia
E-mail: *kmtay@feng.unimas.my

***Abstract*— An improved evolving model, i.e., Evolving Tree (ETree) with Fuzzy c-Means (FCM), is proposed for undertaking text document visualization problems in this study. ETree forms a hierarchical tree structure in which nodes (i.e., trunks) are allowed to grow and split into child nodes (i.e., leaves), and each node represents a cluster of documents. However, ETree adopts a relatively simple approach to split its nodes. Thus, FCM is adopted as an alternative to perform node splitting in ETree. An experimental study using articles from a flagship conference of Universiti Malaysia Sarawak (UNIMAS), i.e., Engineering Conference (ENCON), is conducted. The experimental results are analyzed and discussed, and the outcome shows that the proposed ETree-FCM model is effective for undertaking text document clustering and visualization problems.**

Keywords*— Evolving tree; text document clustering; visualization; online learning; fuzzy c-means

I. INTRODUCTION

Clustering is a task of assigning data objects into a number of groups (or clusters) so that the objects in the same cluster share the same similarities, as compared with those in other clusters [1]. It converts a set of non-linear data into a human and/or machine understandable format, which can be very useful for unsupervised learning systems. Examples of some popular clustering methods are the Self-Organizing Map (SOM) [2, 3], k-mean clustering [4], and fuzzy c-mean clustering (FCM) [5, 6]. With respect to SOM, it is an artificial neural network that maps a set of high-dimensional data onto a predefined low-dimensional grid of nodes [2, 7], and retains the topological relationship of the data. From the literature, various applications of SOM, e.g., speech recognition [8, 9], feature extraction [10], robotic arm [11], noise reduction in telecommunication [12], and textual documents clustering [13], have been reported. Indeed, various extensions of SOM, e.g., hierarchical search [14, 15], growing SOM [16, 17], growing hierarchical SOM [18], and evolving tree (ETree) [19], have been proposed over the years. Examples of applications of ETree to a variety of application domains can be found in [20-22]. In general, these approaches increase the flexibility of SOM and improve the learning time for processing large data samples.

Text document clustering (also known as text categorization) is a procedure to assemble similar text documents into groups based on their similarity [23]. Many text document clustering methods are available. Examples include the naive Bayes-based document clustering model [24], WEBSOM [25], and support vector machines-based model for imbalanced text document classification [26]. These approaches allow a collection of documents to be clustered (and visualized). Regardless of the popularity of these approaches, it is not sure how these approaches can be extended to evolving or online learning. Thus, it is important to develop a text document clustering model with evolving capabilities for the following reasons: (1) new documents are generated or created everyday; and (2) it is not practical to perform re-training of a model whenever a new document appears. Thus, an evolving model is useful for tackling text document clustering problems.

The focus of this paper is on an improved evolving model, i.e., ETree combined with FCM (denoted as ETree-FCM), as an alternative to SOM as well as other offline learning methods, for document clustering. Instead of SOM-based models (e.g. WEBSOM [25]), ETree-FCM allows the clustering method to have evolving features. Besides that, it serves as a solution to a few shortcomings of WEBSOM, i.e., the learning time [19], and the difficulty in determining the map size before learning [19]. Even through ETree has been proposed as an alternative to SOM, its application is still limited. To the best of our knowledge, this study is a new attempt to use ETree-FCM in document clustering [27].

It is worth mentioning that ETree adopts a relatively simple approach to allow a trunk node to be split into child nodes. We have previously developed an ETree-based text document clustering and visualization procedure [27]. In this paper, we further extend our previous work by adopting FCM to allow a trunk node to be split into child nodes. FCM is chosen because it is a reliable clustering method [28]. In our proposed ETree-FCM model, some salient features of WEBSOM, e.g., text pre-processing, are retained. A case study with information/data from a flagship conference of Universiti Malaysia Sarawak (UNIMAS), i.e., the Engineering Conference (ENCON), is conducted to evaluate the effectiveness of the proposed approach.

This paper is organized as follows. The background of ETree and FCM are provided in Section II. In Section III, the use of ETree-FCM for text document clustering and visualization is described. An experimental study to evaluate the usefulness of ETree-FCM in text document clustering is presented in Section IV, with the results analyzed and discussed. A summary of conclusion is included in Section V.

II. BACKGROUND

A. Structure of ETree

Fig. 1 depicts an example of the ETree structure. The tree structure consists of n_{node} nodes. Each node is denoted as $N_{l,j}$, where $l = 1,2,3, \ldots n_{node}$ is the identity of the node and

$j = 0,1,2,\ldots$ is its parent node. As an example, the parent node for $N_{2,1}$ is $N_{1,0}$. In other words, $N_{2,1}$ is a child node for $N_{1,0}$. There are three types of nodes, i.e., root node, trunk nodes, and leaf nodes. The root node is the first created node (i.e., $N_{1,0}$) located at the top layer of the tree. It has no parent node (i.e., $j = 0$). A trunk node is a parent node, e.g., $N_{2,1}$. The leaf nodes, i.e., N_{leaf}, are located at the bottom layer of the tree. They are nodes with no child nodes e.g., $N_{8,5}$.

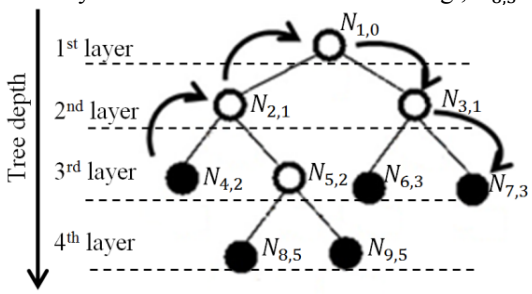


Fig. 1. The structure of ETree.

The tree structure can be described by its tree depth. As an example, the tree depth is 4 in Fig. 1. The distance between two leaf nodes is measured by the shortest path connecting the number of trunks together. As an example, the tree distance between $N_{4,2}$ and $N_{7,3}$ is 3, i.e., $d(N_{4,2}, N_{7,3})=3$, and the connected trunks are $N_{2,1}$, $N_{1,0}$, and $N_{3,1}$. Every leaf node is associated with two attributes: a weight vector, i.e., w_l, and a hit counter, i.e., b_l. Note that w_l is a n-dimension real-valued vector, i.e., $w_l = [w_{l,1}, w_{l,2}, \ldots, w_{l,n}] \in \mathbb{R}^n$, while $b_l = 0,1,2,\ldots$ is a counter that records the number of times $N_{l,j}$ is selected as the Best Matching Unit (BMU).

The degree of matching for a sample, $x(t) = [\xi_1, \xi_2, \ldots, \xi_n]$ corresponding to $N_{l,j}$ which is associated with a weight vector (i.e., w_l) is obtained by the Euclidian distance between $x(t)$ and w_l, as in Eq. (1). The lower the distance, the higher the degree of matching.

$$dist\big(x(t), w_l(t)\big) = \|x(t) - w_l(t)\| \tag{1}$$

B. The Learning Algorithm of ETree

Two pre-determined parameters, i.e., the splitting threshold (i.e., $b_{splitting}$) and the number of split node (i.e., n_{child_node}) are considered. Assuming a new sample, i.e. $x(t)$, is provided to ETree. The learning algorithm to update the tree comprises three steps, as follows:

a) Determining the BMU from root to leaf. A BMU determination process is a top-down process, starting from the 1st layer node (root node) downwards to the bottom layer nodes (leaf nodes). From the root node, the distance of its child nodes at the 2nd layer are computed with $x(t)$, based on Eq. 1. The child node with the minimum distance is the best matched child. Then, the distance of child nodes of the best matched child at the next layer are computed in the same manner. This process is repeated until a leaf node is obtained. The best matched leaf node is the BMU, and is denoted as N_{BMU}, $BMU \in [1,2,3,\ldots n_{node}]$.

b) Updating the leaf node. The weight vectors for each leaf node are updated using the SOM learning rule, as follows [2].

$$w_l(t+1) = w_l(t) + h_{BMU\_l}(t)[x(t) - w_l(t)] \tag{2}$$

where $h_{BMU\_l}(t)$ is a neighborhood function between N_{BMU} and $N_{l,j}$. The neighborhood function is calculated using Eq. 3.

$$h_{BMU\_l}(t) = \propto(t) exp\left(\frac{-d\big(N_{BMU}, N_{l,j}\big)^2}{2\sigma^2(t)}\right) \tag{3}$$

where $\propto(t)$ is the learning rate, $\sigma(t)$ is the Gaussian kernel width, and $d\big(N_{BMU}, N_{l,j}\big)$ is the tree distance between N_{BMU} and $N_{l,j}$. The learning rate and the Gaussian kernel width are calculated using Eq. 4 and Eq. 5, respectively.

$$\propto(t) = 1 - \frac{t}{|N_{leaf}|} \tag{4}$$

$$\sigma(t) = \frac{1}{t} \tag{5}$$

where $\big|N_{leaf}\big|$ is the total number of N_{leaf}.

c) Growing the tree. The hit counter for BMU, i.e, b_{BMU}, is updated using Eq. 6.

$$b_{BMU}(t+1) = b_{BMU}(t) + 1 \tag{6}$$

If $b_{BMU}(t+1) = b_{splitting}$, N_{BMU} is split into n_{child_node} child nodes. N_{BMU} now becomes a trunk node. The weight vectors for the child nodes are initialized as $W_{BMU}(t+1)$. One of its child nodes is randomly chosen as the BMU. The leaf node is further updated with the updating procedure in Step (b).

C. The FCM Clustering Algorithm

FCM [28] is an iterative clustering method to produce a pre-determined number of clusters, c. Assuming $x_{r=1,2,3,\ldots,t} = [\xi_1, \xi_2, \ldots, \xi_n]$ as the data samples to be clustered, and t is the total number of data samples, $m \in [1, \infty]$ is a pre-defined weighting exponent that determines the level of fuzziness, $u_{j,r} \in [0,1]$ is the membership function of x_r in the j-th cluster, where $j = 1,2,\ldots,c$. v_j is the prototype for the center of cluster j, and ε is the pre-determined threshold for FCM errors. The FCM algorithm is as follows [29]:

a) Pre-determine c, m, and ε;
b) Initialize $u_{j,r}$ with a set of random values;
c) Set the iteration counter, $iter = 0$;
d) Calculate v_j with Eq. (7);

$$v_j = \frac{\sum_{r=1}^{t}\big(u_{j,r}^{(iter)}\big)^m x_r}{\sum_{r=1}^{t}\big(u_{j,r}^{(iter)}\big)^m} \tag{7}$$

e) Update $u_{j,r}^{(iter)}$ with E.q (8);

$$u_{j,r}^{(iter)} = \frac{1}{\sum_{k=1}^{c}\left(\frac{|x_r - v_j|}{|x_r - v_k|}\right)^{\frac{2}{m-1}}} \tag{8}$$

f) Check the stopping condition. If $max\{u^{(iter)} - u^{(iter+1)}\} < \varepsilon$, then stop. Otherwise, set $iter = iter + 1$ and repeat steps (d) and (e).

III. Text Document Clustering and Visualiation using ETree with FCM

In this paper, ETree and FCM are combined as a text document clustering and visualization model. Fig. 2 depicts the proposed procedure to construct the ETree-FCM model for text document clustering and visualization. The proposed procedure is divided into two stages: updating the article terms (i.e., steps (1) to (4)) and performing tree learning or updating (i.e., steps (5) to (9)).

A corpus, i.e., D, that consists of $|D|$ articles, has a list of terms, denoted as ψ_D. Each term in ψ_D is denoted as ψ_{D,η_D}, where $\eta_D = 1,2,\dots,n_D$. There are n_D terms. A term (denoted as ψ_{D,η_D}) is assigned to an unique numerical representation, i.e., η_D. An article (i.e., d_i, where $d_i \in D, i = 1,2,\dots,|D|$) is represented as a document vector, i.e., ψ_i. Each ψ_i consists of n_i terms, represented by ψ_{i,k_i}, where $k_i = 1,2,\dots,n_i$. The numerical description for ψ_{i,k_i}, i.e., η_{i,k_i} is the unique representation of ψ_{i,k_i} in ψ_D. Note that ρ_{i,k_i} counts the occurrence of ψ_{i,k_i} in D, while p_{i,k_i} counts the occurrence of ψ_{i,k_i} in d_i. Vector λ_{i,k_i} is a numerical description of the contexts in which term ψ_{i,k_i} has occurred.

The updating and learning algorithm for the proposed ETree-FCM model is as follows.

1) A new article, i.e., $d_{|D|+1}$, is provided.

2) The article is pre-processed, and the abstract is extracted. Symbols and numerical characters are removed. Then, $\psi_{|D|+1}$ and $\psi_{|D|+1,k_{|D|+1}}$ are determined. Less informative words (as defined in a list of stop words) are removed. Next, $\eta_{|D|+1,k_{|D|+1}}$, $\rho_{|D|+1,k_{|D|+1}}$, $p_{|D|+1,k_{|D|+1}}$, and $\lambda_{|D|+1,k_{|D|+1}}$ are determined. If $\psi_{|D|+1,k_{|D|+1}}$ is new (i.e., it does not exist in ψ_D), ψ_D is updated with a new ψ_{D,η_D}, and a new unique numerical representation (i.e., $\eta_D = n_D + 1$) is assigned to ψ_{D,η_D}. After that, $\eta_{|D|+1,k_{|D|+1}}$ is determined.

3) The contexts are encoded, whereby $\lambda_{|d|+1,k_{|d|+1}}$ is obtained by numerical descriptions of terms before and after $\psi_{|D|+1,k_{|D|+1}}$, (as suggested in [13]), as follows. Note that $\eta_{|d|+1,(k_{|D|+1}-1)}$ or $\eta_{|d|+1,(k_{|D|+1}+1)}$ is equal to zero, if the respective term is in the list of stop word.

$$\lambda_{|D|+1,k_{|D|+1}} = \begin{bmatrix} \eta_{|d|+1,(k_{|D|+1}-1)} \\ \eta_{|d|+1,(k_{|D|+1}+1)} \end{bmatrix} \quad (9)$$

4) Document $d_{|D|+1}$ is encoded. The vector space model [30] is used, whereby the encoded document, $d_{|D|+1}$, i.e., $x_{|D|+1}$, is computed with Eq. 10. The term frequency-inverse document frequency function [13] is used to determine the importance of the terms in $d_{|D|+1}$.

$$x_{|D|+1} = \sum_{k=1}^{n_{|D|+1}} \left(p_{|D|+1,k_{|D|+1}} \times log \frac{|D|+1}{p_{|D|+1,k_{|D|+1}}} \times \lambda_{|D|+1,k_{|D|+1}} \right) \quad (10)$$

Document $d_{|D|+1}$ is then mapped onto a 2D data space based on $x_{|D|+1}$.

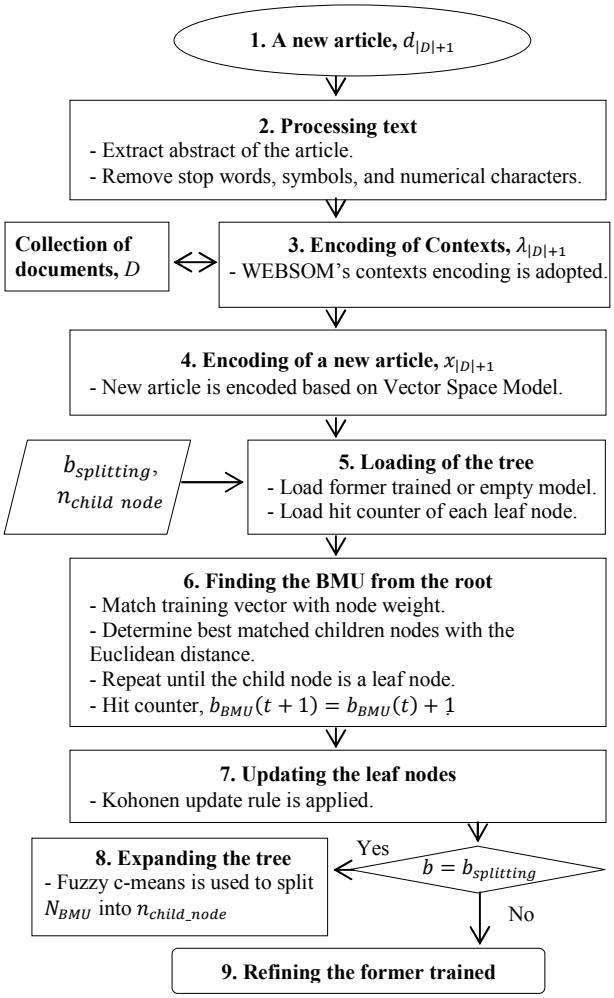


Fig. 2. A flowchart of the proposed ETree-FCM model for text document clustering and visualization.

5) A (empty) tree with its predefined parameters (i.e., $b_{splitting}$, n_{child_node}) is loaded. Each node, i.e., $N_{l,j}$, is attributed with x, and a weight vector (i.e., $w_l(t)$, and an BMU hit counter (i.e., $b_l(t)$).

6) The BMU is determined, as explained in Section II(B). From the root node, the associated documents in its child nodes at the 2nd layer are matched with $x(t)$ based on Eq. 1. The child node with the minimum distance is the best matched child. If the best matched child is not N_{leaf}, repeat step (5). Otherwise, the best matched child is N_{BMU}, whereby $x_{|d|+1}$ is merged into the node, and $b_{BMU} = b_{BMU} + 1$.

7) The leaf nodes are updated using the SOM learning rule (Eqs. 2 – 5).

8) If b_{BMU} equals to $b_{splitting}$, N_{BMU} is split into n_{child_node}. FCM is used for tree splitting. Note that m and ε have to pre-defined. Articles in N_{BMU} is clustered (or split) to n_{child_node} nodes (or clusters). The cluster centre, v, is used to initialize the weight vector of the child nodes.

9) Finally, the trained tree is refined or updated.

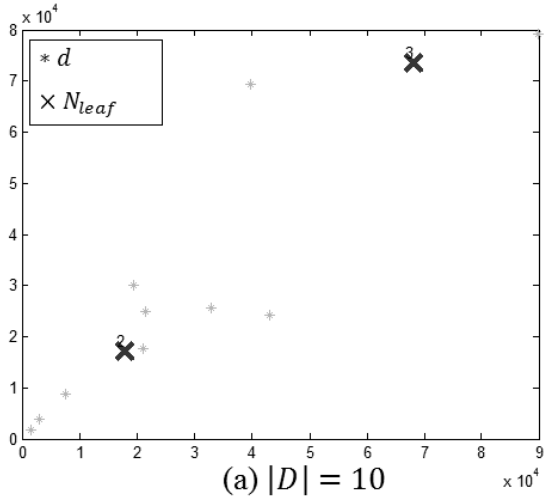
IV. A CASE STUDY

A case study to evaluate the effectiveness of ETree-FCM for text document clustering and visualization was conducted using ENCON, a flagship conference organized by Faculty of Engineering, UNIMAS. In this study, the abstracts of 50 randomly selected articles from ENCON 2008 were used for experimentation. A predefined list of stop words, which consisted of 119 words, was used. The ETree parameters used were: $b_{splitting} = 10{,}15$ and 20, and $n_{child\_node} = 2$. Parameters for FCM were: m=1.25 and ε=0.0001. A laptop with i7-3612QM quad core processer and Windows7 (64-bit) was used. Matlab R2012A was used for software development. The experimental results in terms of the tree structure and computational complexity are analyzed and discussed, as follows.

A. Growing (evolving) of ETree

Fig. 3 shows the growing pattern of ETree for $b_{splitting} = 10$. The document vectors, d, are visualized on a 2-dimensional graph. The clusters are N_{leaf}, and indexed with a numerical index as the identity of the tree, $i \in N_{i,j}$. When $|D| = 13$, 13 documents are clustered into two leaf nodes, i.e., $N_{2,1}$ and $N_{3,1}$, located at the center of the cloud of the documents, as shown in Fig. 3(a). When $|D| = 20$, 20 documents are clustered into four leaf nodes, i.e., $N_{3,1}$, $N_{4,1}$, $N_{6,1}$ and $N_{7,1}$, as shown in Fig. 3(b). Notice that $N_{6,1}$ and $N_{7,1}$ are the leaf nodes split from $N_{5,1}$ or the data from $N_{5,1}$ alone are further clustered into $N_{6,1}$ and $N_{7,1}$. When $|D| = 50$, the 50 articles are clustered into six clusters as shown in Fig. 3 (c).

The clusters contain details of the similarity relationships of the articles. The evolving feature allows calibrations or corrections of the relationship between the clusters. Fig. 4 depict the tree structures formed with $b_{splitting} = 10$. As an example, cluster 4 ($N_{4,2}$) and cluster 3 ($N_{3,1}$), which are the nearest nodes in the tree are expected to have the closest relationship between each other at the first place. But, the use of FCM shows that $N_{3,1}$ has the closest relationship with cluster 7 ($N_{7,5}$), as shown in Fig. 3 (c).



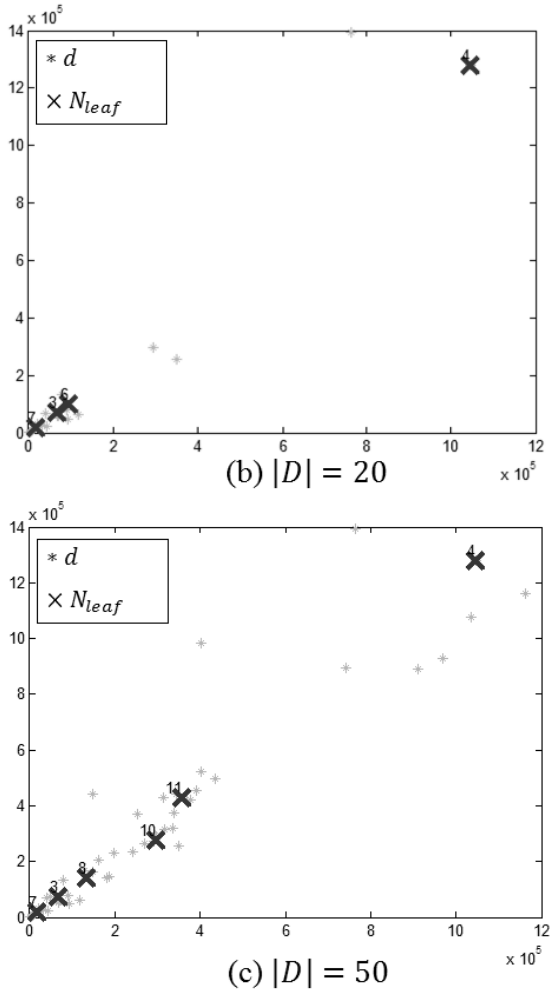


Fig. 3. The growing/evolving patterns of the tree nodes for $b_{splitting} = 10$

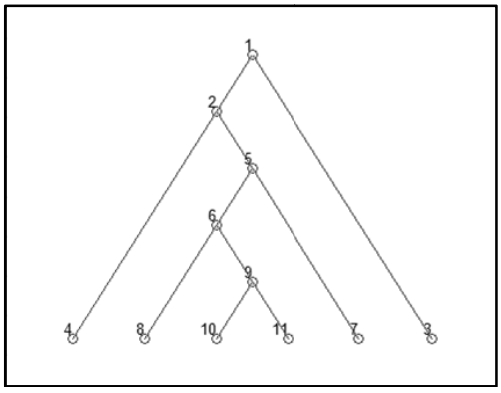


Fig. 4. The evolving tree structure with at $b_{splitting} = 10$.

Figs. 5-6 depict the tree structures formed with $b_{splitting} = 15$ and 20. As $b_{splitting}$ affects the growing rate of the tree, the higher $b_{splitting}$, the smaller the tree size (i.e., smaller number of nodes).

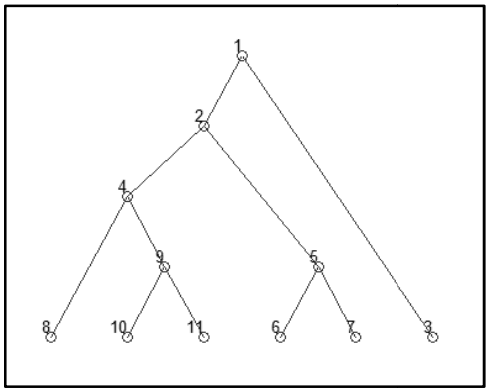


Fig. 5. The evolving tree structure with $b_{splitting} = 15$.

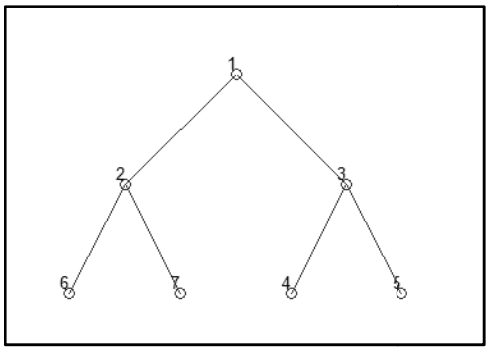


Fig. 6. The evolving tree structure with $b_{splitting} = 20$

Table I summarizes the number of clusters created, tree size (i.e., total number of nodes), and tree depth for $b_{splitting} = 10{,}15$, and 20. The results show the number of clusters created, tree size, and tree depth decreases as $b_{splitting}$ increases. This is because $b_{splitting}$ determines the maximum number of documents at each node. The higher the value of $b_{splitting}$, the lower the tendency a node would split. As a result, the growing rate is reduced. At $|D| = 50$, the tree size is 11, 11, and 7 with $b_{splitting} = 10{,}15$, and 20, respectively.

TABLE I. TREE STRUCTURES WITH DIFFERENT SPLITTING THRESHOLDS

$b_{splitting}$	Number of Clusters	Tree size	Tree depth
10	6	11	6
15	6	11	5
20	4	7	3

Fig. 7 shows the clustering structure using 50 similar articles with $b_{splitting} = 15$. As compared with the results with $b_{splitting} = 10$, the articles are clustered into six clusters. Each cluster represents a larger group, as compared with $b_{splitting} = 10$.

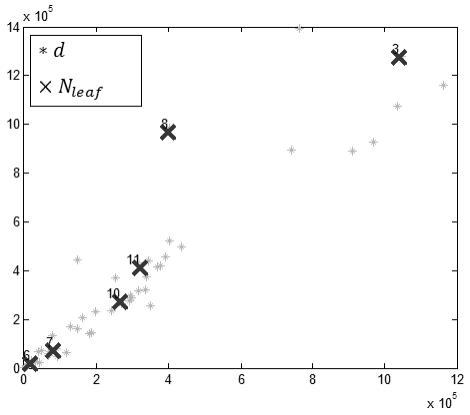


Fig. 7. Comparison of clustering results at $b_{splitting} = 15$.

Fig. 8 shows the clustering structure by using $b_{splitting} = 20$. The number of clusters formed is four, and the distributtions of the clusters are similar to those of $b_{splitting} = 15$.

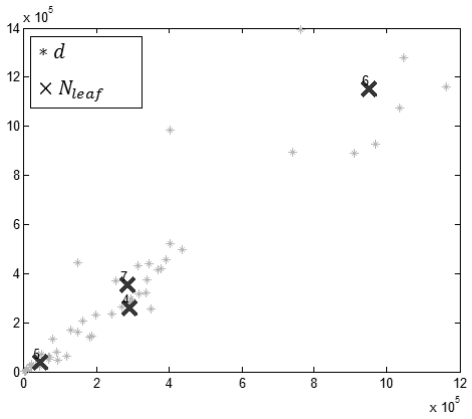


Fig. 8. Comparison of clustering results at $b_{splitting} = 20$.

B. Computational Complexity

In this section, the computational complexity of ETree-FCM is analyzed in two scenarios: (1) analysis of $b_{splitting} = 10$; (2) comparisons of $b_{splitting} = 10{,}15$, and 20.

Fig. 9 depicts the learning time (in second) for each document, with $b_{splitting} = 10$. As can be seen in Fig. 9, the learning time increases in line with $|D|$. The increase in $|D|$ results an increase in the vocabularies in the corpus. The size of the tree structure also increases. This, in turn, leads to a higher degree of computational complexity for learning new articles.

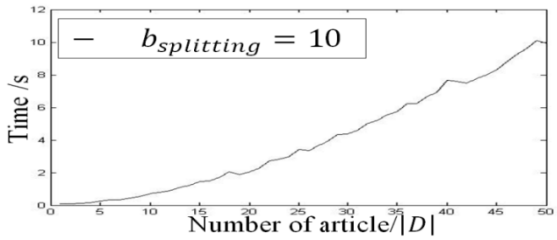


Fig. 9. Time consumptions of evolving learning at $b_{splitting} = 10$.

Fig. 10 depicts the learning time for different $|D|$, with $b_{splitting} = 10,15$, and 20. With $|D| = 50$, the time consumed is less than 12 seconds. From Fig. 10, we can observe that the learning time is less affected by the $b_{splitting}$ setting. The higher the $b_{splitting}$, the higher the computational complexity when the splitting process is executed. However, a lower $b_{splitting}$ setting leads to a higher tree growing rate. The higher the tree size, the higher the computational complexity.

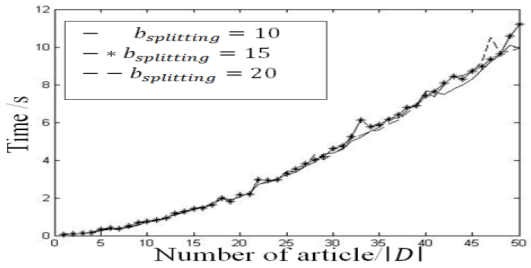


Fig. 10. Time consumptions of evolving learning at $b_{splitting} = 10,15,20$

V. SUMMARY

A new ETree-FCM model for text document clustering and visualization has been proposed in this paper. A case study using articles from ENCON 2008 has been conducted. The results demonstrate that articles from ENCON 2008 could be clustered and visualized effectively as a tree structure. The ETree-FCM model is useful for assisting the conference organization committee in analyzing the manuscripts received, e.g., detecting plagiarism and designing the conference tracks. In short, the proposed model serves as a useful tool for conference organizers.

For further work, the proposed model will be further evaluated using a larger number of articles. This would lead to an expected increase in the computational complexity. Besides that, ETree-FCM with a dynamic n_{child_node} setting can be investigated.

ACKNOWLEDGEMENTS

The financial support of the FRGS grants (No. 6711195, No. 6711229, and FRGS/1/2013/ICT02/UNIMAS/02/1) and ERGS grants (ERGS/02(01)/807/2011(02)) is gratefully acknowledged.

REFERENCES

[1] X.Rui and C.W Donald, Clustering, IEEE Press/Wiley, 2009.

[2] T. Kohonen, Self Organiizing Maps, 3rd ed., Springer, 2001.

[3] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map," IEEE Trans. Neural Netw, vol.11, no.3, pp.586-600, 2000.

[4] W. C. Chang, J. Luo, and K. J. Parker, "Image segmentation via adaptive K-Mean clustering and knowledge-based morphological operations with biomedical applications," IEEE Trans. Image Process., vol.7, no.12, 1998.

[5] M. R. Rezaee, B.P.F. Leliveldt, and J. H. C. Reiber, "A new cluster validity index for the fuzzy c-mean," Pattern Recogn. Lett., vol.19, pp.237-246, 1998.

[6] C. B. James, E. Robert, and F. William, "FCM: the fuzzy c-means clustering algorithm," Comput. Geosci., vol.10, no.2-3, pp.191-203, 1984.

[7] T. Kohonen, S. Kaski, K. Lagus, J. Salojarvi, J. Honkela, V. Paatero, and A. Saarela, "Self organization of a massive document collection," IEEE Trans. Neural Netw., vol.11, no.3, pp.574-595, 2000.

[8] T. Kohonen, O. Simula, and A. Visa, "Engineering applications of the self-organizing map," Proc. IEEE, vol.84, no.10, pp.1358-1384, 1996.

[9] T. Kohonen, and P. Somervuo, "Self-organizing maps of symbol strings," Neurocomputing, vol.21, pp.19-30, 1998.

[10] J. Mao, and A. K. Jain, "Artificial neural networks for feature extraction and multivariate data projection," IEEE Trans. Neural Netw., vol.6, no.2, pp.296-317, 1995.

[11] J. L. Buessler, R. Kara, P. Wira, H. Kihl, and J. P. Urban, "Multiple self-organizing maps to facilitate the learning of visuo-motor correlations," IEEE Int. Conf. Syst., Man, Cybern., Syst., vol,3, pp.470-475, 1999.

[12] T. Kohonen, K. Raivio, O. Simula, J. Henriksson, "Start-up behaviour of a neural network assisted decision feedback equalizer in a two-path channel," Proc.IEEE Conf. Communications, vol.3, pp.1523-1527, 1992.

[13] K. Lagus, S. Kaski, and T. Kohonen, "Mining massive document collections by the WEBSOM method," Inf. Sci., vol.163, pp.135-156, 2004.

[14] C. H. Chung, H. L. Shu, and S. T. Wei, "Apply extended self-organizing map to cluster and classify mixed-type data," Neurocomputing, vol.74, pp.3832-3842, 2011.

[15] W. S. Tai, C. C. Hsu, and J. C. Chen, "A mixed-type self-organizing map with a dynamic structure," Intl. Joint Conf. Neural Netw., pp.1-8, 2010.

[16] S. Matharage, D. Alahakoon, J. Rajapakse, and H. Pin, "Fast growing self organizing map for text clustering," Springer, pp.406-415, 2011.

[17] R. J. Kuo, C. F. Wang, and Z. Y. Chen, "Integration of growing self-organizing map and continuous genetic algorithm for grading lithium-ion battery cells," Appl. Soft Comput., vol.12, pp.2012-2022, 2012.

[18] S. Y. Huang, and R. H. Tsaih, "The prediction approach with growing hierarchical self-organizing map," Intl. Joint Conf. on Neural Netw., pp.1-7, 2012.

[19] J. Pakkanen, J. Iivarinen, and E. Oja, "The evolving tree – analysis and applications," IEEE Trans. Neural Netw., vol.17, no.3, pp.591-603, 2006.

[20] S. Simmuntiet, F. M. Schleif, T. Villmann, and B, Hammer, "Evolving tree for the retrieval of mass spectrometry-based bacteria fingerprints," Knowl. Inf. Syst., vol.25, pp.327-343, 2010.

[21] X. T. Li, Y. M. Ye, M. J. Li, and M. I. Ng, "On cluster tree for nested and multi-density data clustering," Pattern Recogn., vol.43, pp.3130-3143, 2010.

[22] J. Pakkanen, J. Iivarinen, and E. Oja, "The evolving tree-a novel self-organizing network for data analysis," Neural Process. Lett., vol.20, pp.199-211, 2004.

[23] S. Fabrizio, Text Categorization. In Alessandro Zanasi (ed.), Text Mining and its Applications, WIT Press, Southampton, UK, pp.109-129, 2005.

[24] D. Lewis, "Naïve bayes at forty: the independence assumption in information retrieval," European Conf. Mach. Learn., 1998.

[25] A. P. Azcarraga, T. J. Yap, J. Tan, and T. S. Chua, "Evaluating keyword selection methods for WEBSOM text archives," IEEE Trans. Knowl. Data Eng, vol.16, no.3, pp.380-383, 2004.

[26] T. Liu, H. T. Loh, and A. Sun, "Imbalanced text classification: a term weighting approach," Expert Syst. Appl., vol.36, pp.690-701, 2009.

[27] W. L. Chang, K. M. Tay, and C. P. Lim, "An evolving tree for text document clustering and visualization," The 17th Online World Conf. Soft Comput. Ind. Appl., 2012, *Accepted for publication*.

[28] N. R. Pal, and J. C. Bezdek, "On cluster validity for the fuzzy c-means model," IEEE Trans. Fuzzy Syst., vol.3, no.3, pp.370-379, 1995.

[29] R. L. Connon, J. V. Dave, and J. C. Bezdek, "Efficient implementation of the fuzzy c-means clustering algorithms," IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-8, no.2, pp.248-255, 1986.

[30] C. Pablo, F. Miriam, and V. David, "An adaptation of the vector-space model for ontology-based information retrieval," IEEE Trans. Knowl. Data Eng., vol.19, no.2, pp.261-272, 2007.