ISSN:0975-9646

An approach to deal with time- evolving Categorical data based on NIR using Clustering

A. Shiva Prasad [#] , Prof P.Pradeep Kumar[#] , D.Prashanth Kumar[*]

[#]Department of Computer Science & Engg, Vivekananda Institute of Tech & Sciences, JNTUH,Karimnagar,Hyderabad, AP, INDIA

[*] Department of Computer Science & Engg, Kamala Institute of Tech & Science, JNTUH,Karimnagar, AP, INDIA

Abstract: Data clustering is an important technique for exploratory data analysis and has been the focus of substantial research in several domains for decades among which Sampling has been recognized as an important technique to improve the efficiency of clustering. However, with sampling applied, those points that are not sampled will not have their labels after the normal process. Although there is a straightforward approach in the numerical domain, the problem of how to allocate those unlabeled data points into proper clusters remains as a challenging issue in the categorical domain. In this paper, a mechanism named Maximal Resemblance Data Labeling (abbreviated as MARDL) is proposed to allocate each unlabeled data point into the corresponding appropriate cluster based on the novel categorical clustering representative, namely, N-Node set Importance Representative(abbreviated as NNIR), which represents clusters by the importance of the combinations of attribute values. MARDL has two advantages: 1) MARDL exhibits high execution efficiency. 2) MARDL can achieve high intra cluster similarity and low inter cluster similarity, which are regarded as the most important properties of clusters, thus benefiting the analysis of cluster behaviors. MARDL is empirically validated on real and synthetic data sets and is shown to be significantly more efficient than prior works while attaining results of high quality.

***Key Words*—Data mining, categorical clustering, data labeling.**

1. INTRODUCTION

1.1 DATA CLUSTERING

Data clustering is an important technique for exploratory data analysis and has been the focus of substantial research in several domains for decades. The problem of clustering is defined as follows: Given a set of data objects, the problem of clustering is to partition data objects into groups in such a way that objects in the same group are similar while objects in different groups are dissimilar according to the predefined similarity measurement. Therefore, clustering analysis can help us to gain insight into the distribution of data. However, a difficult problem with learning in many real world domains is that the concept of interest may depend on some hidden context, not given explicitly in the form of predictive features. In other words, the concepts that we try to learn from those data drift with time .For example, the buying preferences of customers may change with time, depending on the current day of the week, availability of alternatives, discounting rate, etc. As the concepts behind the data evolve with time, the underlying clusters may also change considerably with time .Performing clustering on the entire time-evolving data not only decreases the quality of clusters but also disregards the expectations of users, which usually require recent clustering results.

1.2 PROBLEM OF CLUSTERING TIME-EVOLVING DATA

Actually, categorical attributes also prevalently exist in real data with drifting concepts. For example, buying records of customers, Web logs that record the browsing history of users, or Web documents often evolve with time. Previous works on clustering categorical data focus on doing clustering on the entire data set and do not take the drifting concepts into consideration. Therefore, the problem of clustering time evolving data in the categorical domain remains a challenging issue. As a result, a framework for performing clustering on the categorical time-evolving data is proposed in this paper. Instead of designing a specific clustering algorithm, we propose a generalized clustering framework that utilizes existing clustering algorithms and detects if there is a drifting concept or not in the incoming data. In order to detect the drifting concepts, the sliding window technique is adopted. Sliding windows conveniently eliminate the outdated records and the sliding windows technique is utilized in several previous works on clustering time-evolving data in the numerical domain.

Therefore, based on the sliding window technique, we can test the latest data points in the current window if the characteristics of clusters are similar to the last clustering result or not. In, a similar strategy with our framework that utilizes clustering results to analyze the drifting concepts is proposed in the numerical domain. The online clustering algorithm is performed on each time frame, and several numerical characteristics such as the mean and standard deviation of cluster centers are used to represent clustering results and detect the changes between time frames. After the change is detected, an offline voting-based classification algorithm is performed to associate each change with its correspondent event. However, in the categorical domain, the above procedure is infeasible because the numerical characteristics of clusters are difficult to define. Therefore, for capturing the characteristics of clusters, an effective cluster representative that summarizes the clustering results is required. In this paper, a practical categorical clustering representative, named "Node Importance Representative" (abbreviated as NIR) , is utilized. NIR represents clusters by measuring the importance of each attribute value in the clusters. Based on NIR, we propose the "Drifting Concept Detection" (abbreviated as DCD) algorithm in this paper. In DCD, the incoming categorical data points at the present sliding window are first allocated into the corresponding proper cluster at the last clustering result, and the number of outliers that are not able to be assigned into any cluster is counted. After that, the distribution of clusters and outliers between the last clustering result and the current temporal clustering result are compared with each other. If the distribution is changed (exceeding some criteria), the concepts are said to drift. In the concept-drifting window, the data points will do re-clustering, and the last clustering representative will be dumped out. On the contrary, if the concept is steady, the clustering representative (NIR) will be updated. Moreover, the framework presented in this

paper not only detects the drifting concepts in the categorical data but also explains the drifting concepts by analyzing the relationship between clustering results at different times. The analyzing algorithm is named "Cluster Relationship Analysis" (CRA). When the drifting concept is detected by DCD, the last clustering representative is dumped out. Therefore, each clustering representative that had been recorded represents a successive constant clustering result, and different dumped-out representatives describe different concepts in the data set. By analyzing the relationship between clustering results, we may capture the time evolving trend that explains why the clustering results have changes in the data set.

2. ANALYSIS AND TECHNICAL RESULTS

2.1 PROBLEM FORMULATION

The problem of clustering the categorical time-evolving data is formulated as follows: Suppose that a series of categorical data points D is given, where each data point is a vector of q attribute values, i.e., $p_j = (p_j^1, p_j^2, \ldots, p_j^q)$. Let $A = \{A_1, A_2, \ldots, A_a\}$, where Aa is the ath categorical attribute, $1 \leq a \leq q$. In addition, suppose that the window size N is also given. The data set D is separated into several continuous subsets St, where the number of data points in each St is N. The superscript number t is the identification number of the sliding window and t is also called time stamp in this paper. For example, the first N data points in D are located in the first subset S^1. Based on the foregoing, the objective of the framework is to perform clustering on the data set D and consider the drifting concepts between S^t and S^{t+1} and also analyze the relationship between different clustering results.

For ease of presentation, several notations are defined as follows:

In our framework, several clustering results at different time stamps will be reported. Each clustering result C½t1;t2_ is formed by one stable concept that persists for a period of time, i.e., the sliding windows from t1 to t2. The clustering results $C^{[t1,t2]}$ contain $K^{[t1,t2]}$ clusters, i.e., $C^{[t1,t2]} = \{c_1^{[t_1,t_2]}, c_2^{[t_1,t_2]}, \ldots, c_{k^{[t_1,t_2]}}^{[t_1,t_2]}\}$

Where $c_i^{[t_1,t_2]}$, $1 \leq i \leq k^{[t_1,t_2]}$, is the ith cluster in $C^{[t1,t2]}$. If $t_1 \equiv t_2 \equiv t$, we simplify the superscript by t. For example, the first clustering result that is obtained from the initial clustering step is C1. Moreover, if we do not point out a specific time stamp, the superscript will be omitted for ease of presentation. In addition, when the DCD algorithm is performed, a temporal clustering result, which is utilized to detect the drifting concept at each sliding window, will be obtained. The notation C0t is used to represent the temporal clustering result at time stamp t.

2.2 NODE IMPORTANCE REPRESENTATIVE (NIR)

The basic idea behind NIR is to represent a cluster as the distribution of the attribute values, which are called "nodes". In order to measure the represent ability of each node in a cluster, the importance of a node is evaluated based on the following two concepts:

1. The node is important in the cluster when the frequency of the node is high in this cluster.
2. The node is important in the cluster if the node appears prevalently in this cluster

rather than in other clusters. The formal definitions of nodes and node importance are shown as follows:

Definition (node). A node, is defined as attribute name + attribute value.

Definition(node importance). The importance value of the node Iir is calculated as the following equations:

$$w(c_i, I_{ir}) = \frac{|I_{ir}|}{m_i} * f(I_r),$$

$$f(I_r) = 1 - \frac{-1}{\log k} * \sum_{y=1}^{k} p(I_{yr}) \log\big(p(I_{yr})\big),$$

where

$$p(I_{yr}) = \frac{|I_{yr}|}{\sum_{z=1}^{k} |I_{zr}|}.$$

$w(c_i, I_{ir}^n)$ represents the importance of node Iir in cluster ci with two factors, the probability of Iir being in ci and the weighting function f(I_r). Based on the concepts of the node importance, the probability of Iir being in ci computes the frequency of Iir in the cluster ci, and the weighting function is designed to measure the distribution of the node between clusters based on the information theorem [28]. Entropy is the measurement of information and uncertainty on a random variable. Formally, ifX is a discrete random variable with possible state x1… x_n and $p(x_i) = \Pr(X = x_i)$ is the probability of the ith state of X, the entropy E(X) is defined as shown in the following equation:

$$E(X) = -\sum_{i=1}^{n} p(x_i) \log(p(x_i)).$$

The weighting function $f(I_{ir}^n)$ measures the entropy of the node between clusters. The entropy E(X) is maximal when the random variable X has a uniform distribution, which means that X possesses maximal uncertainty or minimum information when we obtain a value of X. The weighting function $f(I_{ir}^n)$measures the entropy of the node between clusters. Suppose that there is a node that occurs in all clusters uniformly. The node that contains the maximum uncertainty provides less clustering characteristics. Therefore, this node should have a small weight. Moreover, the maximum entropy value of a node between clusters equals log k. In order to normalize the weighting function from zero to one, the entropy value of a node between clusters is divided by log k. After that, the normalized entropy is subtracted by one so that the node containing large entropy will obtain a small weight. The importance of the node Iir in cluster ci is measured by multiplying the first concept, i.e., the probability of Iir being in ci, and the second concept, i.e., the weighting function f(I_r). Note that the range of both the probability of Iir being in ci and the weighting function f(I_r) is [0, 1], implying that the range of the important value $w(c_i, I_{ir})$ is also [0, 1]. NIR is related to the idea of conceptual clustering, which creates a conceptual structure to represent a concept (cluster) during clustering. However, NIR only

analyzes the conceptual structure and does not perform clustering, i.e., there is no objective function such as category utility (CU) in conceptual clustering to lead the clustering procedure. Furthermore, NIR considers both the intra cluster similarity and the inter cluster similarity in the representation by integrating the first and the second concepts.

2.3 DRIFTING CONCEPT DETECTION (DCD)

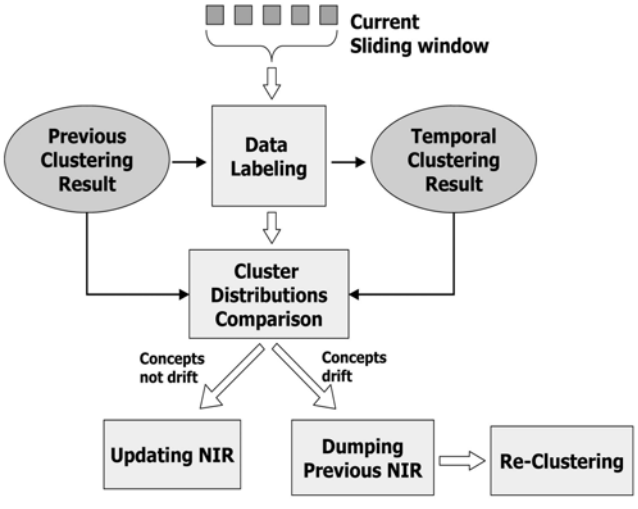
The objective of the DCD algorithm is to detect the difference of cluster distributions between the current data subset St and the last clustering result $C^{[t_e,t-1]}$ and to decide whether the re-clustering is required or not in St. Therefore, the incoming categorical data points in St should be able to be allocated into the corresponding proper cluster at the last clustering result efficiently. We named the allocating process as "data labeling".

The data point that does not belong to any proper cluster is called an outlier. After labeling, the last clustering result $C^{[t_e,t\ 1]}$ and the current temporal clustering result C^{rt} obtained by data labeling are compared with each other. If the difference of cluster distributions is large enough, the sliding window t will be considered as a concept-drifting window, and St will perform reclustering. The flowchart of the DCD algorithm is shown in Fig.

In the Cluster Distribution Comparison step, the last clustering result and the current temporal clustering result obtained by data labeling are compared with each other to detect the drifting concept. The clustering results are said to be different according to the following two criteria:

1.The clustering results are different if quite a large number of outliers are found by
data labeling.

2. The clustering results are different if quite a large number of clusters are varied in the ratio of data points.



Since the idea of data labeling is to present the original clustering characteristics to the incoming data points, , the outliers that are not able to allocate to any cluster may be generated based on different concepts. Therefore, if too many outliers are detected by the data labeling step, the drifting concept may happen in the current sliding window. As a result, a threshold Θ named outlier threshold is set in this step. If the ratio of outliers in the current sliding window is larger than the outlier threshold, the clustering results are said to be different, and the concept is said to drift.

Moreover, another type of drifting concept is also detected in this step. The ratio of data points in a cluster may be changed dramatically by a drifting concept, e.g., the cluster that contains half of the data points in the last clustering result has suddenly disappeared in the current clustering result. In order to detect the change, we adopt a double-threshold method. One threshold € named cluster variation threshold is utilized to determine that the variation of the ratio of data points in a cluster is big enough. The cluster that exceeds the cluster variation threshold is seen as a different cluster. And then, the number of different clusters is counted, and the ratio of different clusters is compared with the other threshold _ named cluster difference threshold. If the ratio of different clusters is larger than the cluster difference threshold, the concept is said to drift in the current sliding window.

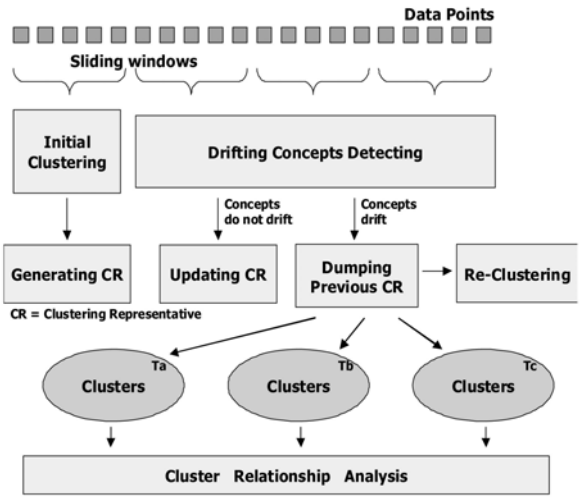
2.4 CLUSTERING RELATIONSHIP ANALYSIS (CRA)

After we perform clustering on the entire data set D where the drifting concepts are considered, several clustering results with time stamps are obtained and represented by NIR. Each clustering result is generated from one concept that persists for a period of time. In addition to reporting each clustering result, we also provide the CRA algorithm, which tries to explain the drifting concepts based on the evolving clustering results. A similar strategy that analyzes the clustering results for mining evolving user profiles in the Web is proposed. This method defines the birth, persistence, atavism, and death of the profiles (clusters) to model the profiling Web usages. In order to deal with the Web data set, a particular hierarchical clustering algorithm is applied, and the algorithm for tracking evolving user profiles is based on the clustering results. In contrast, in this paper, we propose a generalized framework that detects the drifting concept and try to show the evolving clustering results in the categorical domain.

CRA measures the similarity of clusters between the clustering results at different time stamps and links the similar clusters. The linked clusters show the relationship from the last clustering result to the current result. Explicitly, the linking situation presents that the clusters in the current clustering result are split, merged, or changed in size from the last clustering result. Based on the relationship analysis, the evolving clusters will provide clues for us to catch the time evolving trends in the data set. In the following, we will introduce the main problem in CRA—how to link similar clusters at different time stamps, i.e., how to calculate the similarity between each pair of clusters at different time stamps.

2.5 SYSTEM ARCHITECTURE

The problem of clustering time evolving data in the categorical domain remains a challenging issue. A framework for performing clustering on the categorical time-evolving data is proposed in this paper. Instead of designing a specific clustering algorithm, we propose a generalized clustering framework that utilizes existing clustering algorithms and detects if there is a drifting concept or not in the incoming data.



The above Figure shows our entire framework of performing clustering on the categorical time-evolving data. In order to detect the drifting concepts, the sliding window technique is adopted. Sliding windows conveniently eliminate the outdated records, and the sliding windows technique is utilized in several previous works on clustering time-evolving data in the numerical domain. Therefore, based on the sliding window technique, we can test the latest data points in the current window if the characteristics of clusters are similar to the last clustering result or not.

A similar strategy with our framework that utilizes clustering results to analyze the drifting concepts is proposed in the numerical domain. The online clustering algorithm is performed on each time frame, and several numerical characteristics such as the mean and standard deviation of cluster centers are used to represent clustering results and detect the changes between time frames. After the change is detected, an offline voting-based classification algorithm is performed to associate each change with its correspondent event. However, in the categorical domain, the above procedure is infeasible because the numerical characteristics of clusters are difficult to define.

Therefore, for capturing the characteristics of clusters, an effective cluster representative that summarizes the clustering results is required. In this paper, a practical categorical clustering representative, named "Node Importance Representative" (abbreviated as NIR), is utilized. NIR represents clusters by measuring the importance of each attribute value in the clusters. Based on NIR, we propose the "Drifting Concept Detection" (abbreviated as DCD) algorithm in this paper. In DCD, the incoming categorical data points at the present sliding window are first allocated into the corresponding proper cluster at the last clustering result, and the number of outliers that are not able to be assigned into any cluster is counted. After that, the distribution of clusters and outliers between the last clustering result and the current temporal clustering result are compared with each other. If the distribution is changed (exceeding some criteria), the concepts are said to drift. In the concept-drifting window, the data points will do re-clustering, and the last clustering representative will be dumped out. On the contrary, if the concept is steady, the clustering representative (NIR) will be updated.

Moreover, the framework presented in this paper not only detects the drifting concepts in the categorical data but also explains the drifting concepts by analyzing the relationship between clustering results at different times. The analyzing algorithm is named "Cluster Relationship Analysis" (CRA). When the drifting concept is detected by DCD, the last clustering representative is dumped out. Therefore, each clustering representative that had been recorded represents a successive constant clustering result, and different dumped-out representatives describe different concepts in the data set. By analyzing the relationship between clustering results, we may capture the time evolving trend that explains why the clustering results have changes in the data set.

2.6 TECHNICAL RESULTS

2.6.1 EVALUATION ON EFFICIENCY AND SCALABILITY

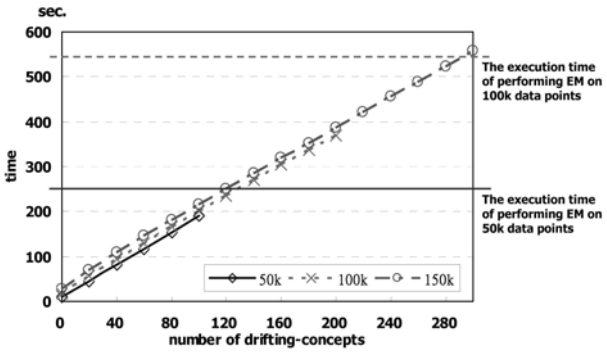


Fig: Evaluation on Efficiency

Figure shows the scalability with the data size of DCD. This study fixes the dimensionality to 20 and the number of clusters to 20 and also tests DCD in different numbers of data points, i.e., 50,000, 100,000, and 150,000. The sliding window size is set to 500. For the reason that the bottleneck of the execution time in DCD is to perform re-clustering on the concept-drifting window, the number of drifting concepts directly impacts the execution time of DCD. The result is, shown in Figure, that as the number of drifting concepts increases, the execution time of DCD also increases. Moreover, the increase of the data size only has little influence on the execution time when the number of drifting concepts is the same. In addition, even if all the windows are detected as the concept-drifting windows, the execution time of DCD is still faster than that of EM. Therefore, algorithm DCD can ensure efficient execution on clustering time-evolving data when the data size is large.

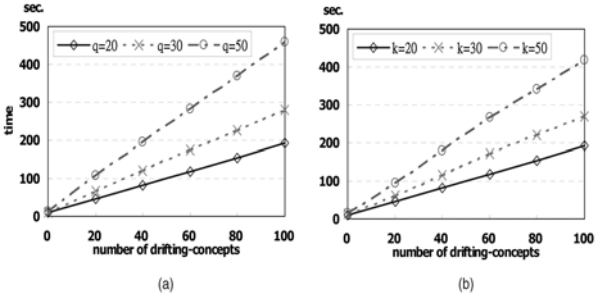


Fig: Evaluation on Scalability

The scalability of DCD with data dimensionality and the number of clusters are shown in Figure. In Fig. a, the study fixes the data size to 50,000 and the number of cluster to 20 and also varies the data dimensionality q as 20, 30, and 50. In Fig. b, the study fixes the data size to 50,000 and the data dimensionality to 20 and also varies the number of clusters k as 20, 30, and 50. Both studies set the sliding window size to 500. The experiments also show that the number of drifting concepts that require doing re-clustering is the bottleneck of the execution time in DCD when the data dimensionality and the number of clusters are varied. However, the execution time of DCD is still faster than that of EM regardless of how many drifting concepts happen, showing the robustness of the DCD algorithm.

2.6.2 EVALUATION ON ACCURACY

We test the accuracy of DCD on both synthetic and real data sets. First, we will test the accuracy of drifting concepts that are detected by DCD. And then, in order to evaluate the results of clustering algorithms, we adopt the following two widely used methods. The CU function. The CU function attempts to maximize both the probability that two data points in the same cluster obtain the same attribute values and the probability that data points from different clusters have different attributes. The expression to calculate the expected value of the CU function is shown in the following equation:

$$CU = \sum_{i=1}^{k} \frac{m_i}{N} \sum_{r=1}^{z} \left[P(I_r|c_i)^2 - P(I_r)^2 \right],$$

where the number of data points in cluster ci is mi, and there are totally z distinct nodes in the clustering results. Confusion matrix accuracy (CMA). Since the synthetic data sets contain the clustering label on each data point, we can evaluate the clustering results by comparing with the original clustering labels. In the confusion matrix , the entry (i ; j) is equal to the number of data points assigned to output cluster ci and that contain the original clustering label j. We measure the accuracy in this matrix (CMA) by maximizing the count of the one-to one mapping in which one output cluster ci is mapped to one original clustering label j

3. CONCLUSION

Here we proposed a framework to perform clustering on categorical time-evolving data. The framework detects the drifting concepts at different sliding windows, generates the clustering results based on the current concept, and also shows the relationship between clustering results by visualization. In order to detect the drifting concepts at different sliding windows, we proposed the algorithm DCD to compare the cluster distributions between the last clustering result and the temporal current clustering result. If the results are quite different, the last clustering result will be dumped out, and the current data in this sliding window will perform re-clustering. In addition, in order to observe the relationship between different clustering results, we proposed the algorithm CRA to analyze and show the changes between different clustering results. The experimental evaluation shows that performing DCD is faster than doing clustering once on the entire data set, and DCD can provide high-quality clustering results with correctly detected drifting concepts in both synthetic and real data cases. Therefore, the result demonstrates that our framework is practical for detecting drifting concepts in time-evolving categorical data.

4. REFERENCES

1. Hung-Leng Chen, Ming-Syan Chen, and Su-Chen LinG. “Catching the Trend: A Framework for Clustering Concept-Drifting Categorical Data” IEEE 2009

2. G. Hulten, L. Spencer, and P. Domingos, “Mining Time-Changing Data Streams,” Proc. ACM SIGKDD, 2001.

3. B.-R. Dai, J.-W. Huang, M.-Y. Yeh, and M.-S. Chen, “Adaptive Clustering for Multiple Evolving Streams,” IEEE Trans. Knowledge and Data Eng., vol. 18, no. 9, pp. 1166-1180, Sept. 2006.

4. C.C. Aggarwal, J.L. Wolf, P.S. Yu, C. Procopiuc, and J.S. Park, “Fast Algorithms for Projected Clustering,” Proc. ACM SIGMOD ’99, pp. 61-72, 1999

5. F. Cao, M. Ester, W. Qian, and A. Zhou, “Density-Based Clustering over an Evolving Data Stream with Noise,” Proc. Sixth SIAM Int’l Conf. Data Mining (SDM), 2006.

6. H.-L. Chen, K.-T. Chuang, and M.-S. Chen, “Labeling Unclustered Categorical Data into Clusters Based on the Important Attribute Values,” Proc. Fifth IEEE Int’l Conf. Data Mining (ICDM), 2005.

7. A.P. Dempster, N.M. Laird, and D.B. Rubin, “Maximum Likelihood from Incomplete Data via the EM Algorithm,” J. Royal Statistical Soc., 1977.

8. O. Nasraoui, M. Soliman, E. Saka, A. Badia, and R. Germain, “A Web Usage Mining Framework for Mining Evolving User Profiles in Dynamic Web Sites,” IEEE Trans. Knowledge and Data Eng., vol. 20, no. 2, pp. 202-215, Feb. 2008

9. A.K. Jain, M.N. Murty, and P.J. Flynn, “Data Clustering: A Review,” ACM Computing Surveys, 1999.