Application of Enhanced Clustering For Different Data Mining Techniques

P.Suganyadevi
Department of Computer Science, Kovai kalaimagal College of Arts and Science, Coimbatore, India.

J.Savitha
Assistant Professor, Department of Computer Science, Kovai kalaimagal College of Arts and Science, Coimbatore, India.

Abstract – Iris data are rather very complex and it is very difficult to predict the behavior of runoff based on temporal data sets. This paper has been proposes a Modified approach K-Means clustering and enhanced pca algorithm which executes K-means algorithm this Algorithm approach is better in the process in large number of clusters and its time of execution is comparisons base on K-Mean,DBSCAN algorithm approach. If the process experimental result is using the proposed algorithm it time of computation can be reduced with a group in runtime constructed data sets are very promising. Modified Approach of K Mean Algorithm and enhanced pca is better than K Mean and dbscan for Large Data Sets.

Index Terms – Temporal, Clustering, Data mining, Hierarchical, Hard and soft clustering, Hydrological process, Time series sequences. Dbscan, Mkmeans, Epca.

1. INTRODUCTION

1.1. K-Medoids Methods

In k-medoids methods a cluster is represented by one of its points. We have already mentioned this is an easy solution since it covers any attribute types and that medoids have embedded resistance against outliers since peripheral cluster points do not affect them. When medoids are selected, clusters are defined as subsets of points close to respective medoids, and the objective function is defined as the averaged distance or another dissimilarity measure between a point and its medoid. Two early versions of k-medoid methods are the algorithm PAM (Partitioning around Medoids) and the algorithm CLARA (Clustering LARge Applications) [Kaufman & Rousseeuw 1990]. PAM is iterative optimization that combines relocation of points between perspective clusters with re-nominating the points as potential medoids. The guiding principle for the process is the effect on an objective function, which, obviously, is a costly strategy. CLARA uses several (five) samples, each with 40+2k points, which are each subjected to PAM. The whole dataset is assigned to resulting medoids, the objective function is computed, and the best system of medoids is retained. Further progress is associated with Ng & Han [1994] who introduced the algorithm CLARANS (Clustering Large Applications based upon Randomized Search) in the context of clustering in spatial databases. Authors considered a graph whose nodes are the sets of k medoids and an edge connects two nodes if they differ by exactly one medoid. While CLARA compares very few neighbors corresponding to a fixed small sample, CLARANS uses random search to generate neighbors by starting with an arbitrary node and randomly checking maxneighbor neighbors. If a neighbor represents a better partition, the process continues with this new node. Otherwise a local minimum is found, and the algorithm restarts until numlocal local minima are found (value numlocal=2 is recommended). The best node (set of medoids) is returned for the formation of a resulting partition. The complexity of CLARANS is O in terms of number of points. Ester et al. [1995] extended CLARANS to spatial VLDB. They used R*-trees [Beckmann 1990] to relax the original requirement that all the data resides in core memory, which allowed focusing exploration on the relevant part of the database that resides at a branch of the whole data tree.

1.2. Density-Based Partitioning

An open set in the Euclidean space can be divided into a set of its connected components. The implementation of this idea for partitioning of a finite set of points requires concepts of density, connectivity and boundary. They are closely related to a point's nearest neighbors. A cluster, defined as a connected dense component, grows in any direction that density leads. Therefore, density-based algorithms are capable of discovering clusters of arbitrary shapes. Also this provides a natural protection against outliers. Figure 4 illustrates some cluster shapes that present a problem for partitioning relocation clustering (e.g., k-means), but are handled properly by density-based algorithms. They also have good scalability. These outstanding properties are tempered with certain inconveniencies.

Figure 4. Irregular shapes difficult for *k*-means are

From a very general data description point of view, a single dense cluster consisting of two adjacent areas with significantly different densities (both higher than a threshold) is not very informative. Another drawback is a lack of interpretability. An excellent introduction to density based methods is contained in the textbook [Han & Kamber 2001].

Since density-based algorithms require a metric space, the natural setting for them is spatial data clustering [Han et al. 2001; Kolatch 2001]. To make computations feasible, some index of data is constructed (such as R*-tree). This is a topic of active research. Classic indices were effective only with reasonably low-dimensional data. The algorithm DENCLUE that, in fact, is a blend of a density-based clustering and a grid based preprocessing is lesser affected by data dimensionality. There are two major approaches for density-based methods. The first approach pins density to a training data point and is reviewed in the sub-section Density-Based Connectivity. Representative algorithms include DBSCAN, GDBSCAN, OPTICS, and DBCLASD. The second approach pins density to a point in the attribute space and is explained in the sub-section Density Functions. It includes the algorithm DENCLUE.

1.3. Supervised Learning

Relation to Supervised Learning Both Forgy's k-means implementation and EM algorithms are iterative optimizations. Both initialize k models and then engage in a series of two-step iterations that: (1) reassign (hard or soft) data points, (2) update a combined model. This process can be generalized to a framework relating clustering with predictive mining [Kalton et al. 2001]. The model update is considered as the training of a predictive classifier based on current assignments serving as the target attributes values supervising the learning. Points' reassignments correspond to the forecasting using the recently trained classifier. Liu et al. [2000] suggested another elegant connection to supervised learning. They considered binary target attribute defined as Yes on points subject to clustering, and defined as No on non-existent artificial points uniformly distributed in a whole attribute space. A decision tree classifier is applied to the full synthetic data. Yes–labeled leaves correspond to clusters of input data. The new technique CLTree (CLustering based on decision Trees) resolves the challenges of populating the input data with artificial No– points such as: (1) adding points gradually following the tree construction; (2) making this process virtual (without physical additions to input data); (3) problems with uniform distribution in higher dimensions.

2. LITERATURE SURVEY

We will first review the concepts of temporal data mining and how is differ from conventional time series sequences is depicted, then its various tasks along with different classes are described. Temporal data mining is concerned with extraction of hidden information of large sequential data sets. Sequential data means data that is ordered with respect to some constraint index. For example, time series constitute a popular class of sequential data, where records are indexed by time. It is clear that in temporal data mining it is the ordering among the records is very important and that ordering is the core to the data description/modeling rather than notion of time [3].Discovery of casual relationships and the discovery of similar patterns within the same time of sequences or among different temporally-oriented events (often called as time series analysis or trend analysis), are the two primarily task of temporal data mining [5]. The supreme goal of temporal data mining is to get wind of hidden relations between sequences and subsequence of events. One main difference between temporal and conventional time series data mining lies in the size and nature of data sets and the manner in which the data is collected [18]. The second major difference lies in the type of query that we want to estimate or discover from the data [3].

2.1. Temporal Data Mining Task:

The possible objectives (or more often we called as „tasks") of temporal data mining can be classified as Association, Prediction, Classification , Clustering, Characterization, Search and retrieval, Pattern discovery, Trend analysis and lastly the Sequence Analysis [1].

Classes of Temporal Data

A. Static Data

Data are called static if all their feature values do not change with time, or change negligibly 16].

B. Sequences

Sequences are commonly referred as ordered sequence of the events or transaction. Though there may not be any explicit reference to time, yet there exists a sort of qualitative temporal relationship (like before, after, during, meet and overlap etc.) between data items.

C. Time Stamped

This category of the temporal data has explicit time related informat ion. Relationship can be quantitative i.e. we can find the exact temporal distance between data element. The consequences obtained through this type of data may be temporal or non-temporal in nature.

D. Time Series

Time series data is special case of the time stamped data. In time series data events have uniform distance on the time scale.

E. Fully Temporal

Data of this category is fully time dependent. The inferences are also strictly temporal [1].Clustering has a long history, with lineage dating back to Aristotle [6]. In our text, we presented some important survey papers on clustering techniques,

1. Pedro Pereiva Rodrigous et al. [22] developed an incremental system for clustering streaming time series, using Online Divisive Agglomerative Clustering ODAC system using top-down strategy i.e. hierarchy of clusters. The system uses correlation as similarity measure. It do not need a predefined number of target clusters. It provides a good performance on finding the correct number of clusters obtained by a bunch of runs of k-Means. The disadvantage of this system is when the tree structure expands, the variables should move from root to leaf, when there is no statistical confidence on the decision of assignment may split variables.

2. S. Mishra et al. [17] presented a comparative study based on K-means clustering and agglomerate hierarchical clustering for developing a predictive model for the discharge process. The analysis is carried out in hydrological daily discharge time series of Panchratna station in the river Brahmaputra and Barak Basin Organization in India. The author used Dynamic Time warping (DTW) for measuring similarities in the data.

3. Ramoni et al. [23] the author presented a study on BCD, a Bayesian algorithm for clustering by dynamics. BCD transforms a set S of n numbers of univariate discrete-valued time series into a Markov chain (MC) and then clusters similar MCs to discover the most probable set of generating processes. BCD is basically an unsupervised algorithms based on agglomerative clustering method. The clustering result is evaluated mainly by a measure of the loss of data information induced by clustering, which is specific to the proposed clustering method. They also presented a Bayesian clustering algorithm for multivariate time series [24]. The algorithm Searches for the most probable set of clusters given the data using a similarity-based heuristic search method. The measure of similarity is an average of the Kullback–Liebler distances between comparable transition probability tables.

4. Van Wijk and Van Selow [25] in [1999] analyse an agglomerate ive hierarchical clustering of daily power consumption data based on the root mean square distance. How the clusters attributed over the week and over the year were also explored with calendar-based visualization.

5. Kumar et al. [26] in [2002] presented a distance function based on the assumed independent Gaussian models of data errors and used a hierarchical clustering method to group seasonality sequences into a desirable number of clusters. The experimental results based on simulated data and retail data showed that the new method outperformed both k-means and Wards method that do not consider data errors in terms of (arithmetic ic) average estimation error.

6. Vlachos et al. [27] in [2003] introducing a novel anytime version of k-Means clustering algorithm for time series. It is an approach to perform incremental clustering of time-series at various resolutions using the Haar wavelet transform. Using *k-Means* clustering algorithm, for the next level of resolution, they modified the final centers at the end of each resolution as the initial centers. By applying this approach the problem associated with the choices of initial centers for k-Means is completely resolved and it significantly improves the execution time and clustering quality.

7. Li and Biswas [28] the authors described a clustering methodology for temporal data using the hidden Markov model representation. The temporal data are assumed to have Markov property, and may be viewed as the result of a probabilistic walk along a fixed set of (not directly observable) states. The proposed continuous HMM clustering method can be summarized in terms of four levels of nested searches. The HMM refinement procedure for the third-level search starts with an initial model configuration and incrementally grows or shrinks the model through HMM state splitting and merging operations. They generated an artificial data set from three random generative models: one with three states, one with four states, and one with five states, and showed that their method could reconstruct the HMM with the correct model size and near perfect model parameter values.

8. Bicego, M. Et al. [29] in 2003 studied a novel scheme for HMM based sequential data clustering is proposed, inspired on the similarity based paradigm recently introduced in the

Supervised learning context. With this approach, a new representation space is built, in which each object is described by the vector of its similarities with respect to a predeterminate set of other objects. These similarities are determined using hidden Markov models. Clustering is then performed in such a space. By way of this, the difficult problem of clustering of sequences is thus transposed to a more manageable format, the clustering of points (vectors of features). Experimental evaluation on synthetic and real data shows that the proposed approach largely outperforms standard HMM clustering schemes. The main drawback of this approach is the high dimensionality of the resulting feature space, which is equal to the cardinality of the data set.

9. Paredes and Vargas [30] in [2012] their paper presents a novel method to perform clustering of time-series and static data. The method, named Circle-Clustering (CirCle), could be classified as a partition method that uses criteria from SVM and hierarchical methods to perform a better clustering. Different heuristic clustering techniques were tested against the CirCle method by using data sets from UCI Machine Learning Repository. In all tests, CirCle obtained good results and outperformed most of clustering techniques considered in this work. Results showed that Circle can be used with both static and time-series data.

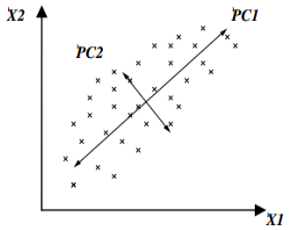
3. PROPOSED METHODS

As a preprocessing stage of data mining and machine learning, dimension reduction not only decreases computational complexity, but also significantly improves the accuracy of the

learned models from large data sets. PCA [11] is a classical multivariate data analysis method that is useful in linear feature extraction. Without class labels it can compress the most information in the original data space into a few new features, i.e., principal components. Handling high dimensional data using clustering techniques obviously a difficult task in terms of higher number of variables involved. In order to improve the efficiency, the noisy and outlier data may be removed and minimize the execution time and we have to reduce the no. of variables in the original data set The central idea of PCA is to reduce the dimensionality of the data set consisting of a large number of variables. It is a statistical technique for determining key variables in a high dimensional data set that explain the differences in the observations and can be used to simplify the analysis and visualization of high dimensional data set.

3.1. Principal Component

A data set xi (i= 1,…,n) is summarized as a linear combination of ortho-normal vectors called principal components, which is shown in the Figure 1.



The first principal component is an axis in the direction of maximum variance. The steps involved in PCA are

Step1: Obtain the input matrix Table

Step2: Subtract the mean

Step3: Calculate the covariance matrix

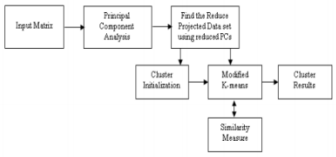
Step4: Calculate the eigenvectors and eigenvalues of the covariance matrix

Step5: Choosing components and forming a feature vector

Step6: deriving the new data set.

The eigenvectors with the highest eigenvalue is the principal component of the data set. In general, once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. To reduce the dimensions, the first d (no. of principal components) eigenvectors are selected. The final data has only d dimensions. The main objective of applying PCA on original data before clustering is to obtain accurate results so that the researchers can do analysis in better way. Secondly, minimize the running time of a system because time taken to process the data is a significant one. Normally it takes more time when the number of attributes of a data set is large and sometimes this dataset not supported by all the clustering techniques hence the number of attributes are directly proportional to processing time. In this paper, PCA is used to reduce the dimension of the data. This is achieved by transforming to a new set of variables (Principal Components) which are uncorrelated and, which are ordered so that the first few retain the most of the variant present in all of the original variables. The first Principal Component is selected to find the initial centroid for the clustering process.

The proposed method that performs data partitioning with Principal component. It partitions the given data set into k sets. The median of each set can be used as good initial cluster centers and then assign each data points to its nearest cluster centroid. The Proposed model is illustrated in Figure 2.



Algorithm 1: The proposed method

Steps: 1.Reduce the dimension of the data into d dimension and determine the initial centroid of the clusters by using Algorithm 2.

2. Assign each data point to the appropriate clusters by using Algorithm

3. In the above said algorithm the data dimensions are reduced and the initial centroids are determined systematically so as to produce clusters with better accuracy.

Algorithm 2: Dimension reduction and finding the initial centroid using PCA.
Steps:
1. Reduce the D dimension of the N data using Principal Component Analysis (PCA) and prepare another N data with d dimensions (d<D).
2. The Principal components are ordered by the amount of variance.
3. Choose the first principal component as the principal axis for partitioning and sort it in ascending order.
4. Divide the Set into k subsets where k is the number of clusters.
5. Find the median of each subset.
6. Use the corresponding data points for each median to initialize the cluster centers.

The initial centroids of the clusters are given as input to Algorithm 3. It starts by forming the initial clusters based on the relative distance of each data-point from the initial centroids. The Euclidean distance is used for determining the closeness of each data point to the cluster centroids. For each data-point, the cluster to which it is assigned and its distance from the centroid of the nearest cluster are noted. For each cluster, the centroids are recalculated by taking the mean of the values of its data-points. The procedure is almost similar to the original k-means algorithm except that the initial centroids are computed systematically. The next stage is an iterative process which makes use of a heuristic method to improve the efficiency. During the iteration, the data-points may get redistributed to different clusters. The method involves keeping

track of the distance between each data-point and the centroid of its present nearest cluster. At the beginning of the iteration, the distance of each data-point from the new centroid of its present nearest cluster is determined. If this distance is less than or equal to the previous nearest distance, that is an indication that the data point stays in that cluster itself and there is no need to compute its distance from other centroids. This result in the saving of time required to compute the distances to k-1 cluster centroids. On the other hand, if the new centroid of the present nearest cluster is more distant from the data-point than its previous centroid, there is a chance for the data-point getting included in another nearer cluster. In that case, it is required to determine the distance of the data-point from all the cluster centroids. This method improves the efficiency by reducing the number of computations.

Algorithm 3: Assigning data-points to clusters

Steps: 1. Compute the distance of each data-point xi (1<=i<=n) to all the centroids cj(1<=j<=k) using Euclidean distance formula..

2. For each data object xi, find the closest centroid cj and assign xi to the cluster with nearest centroid cj and store them in array Cluster[ ] and the Dist[ ] separately. Set Cluster[i] = j, j is the label of nearest cluster. Set Dist[i]= d(xi, cj), d(xi, cj) is the nearest Euclidean distance to the closest center.

3. For each cluster j (1<=j<=k), recalculate the centroids;

4. Repeat

5. for each data-point

5.1 Compute its distance from the centroid of the present nearest cluster

5.2 If this distance is less than or equal to the previous nearest distance, the data-point stays in the cluster Else For every centroid cj Compute the distance of each data object to all the centre Assign the data-point xi to the cluster with nearest centroid cj 6. For each cluster j (1<=j<=k), recalculate the centroids;

Until the convergence criteria is met.

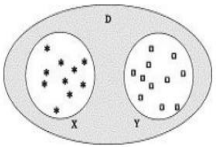
This algorithm requires two data structure Cluster [ ] and Dist[ ] to keep the some information in each iteration which is used in the next iteration. Array cluster [ ] is used for keep the label if the closest centre while data structure Dist [ ] stores the Euclidean distance of data object to the closest centre. The information in data structure allows this function to reduce the number of distance calculation required to assign each data object to the nearest cluster, and this method makes the improved k-means algorithm faster than the standard k-means algorithm.

Modified approach K-mean algorithm: The K-mean algorithm is a popular clustering algorithm and has its application in data mining, image segmentation, bioinformatics and many other fields. This algorithm works well with small datasets. In this paper we proposed an algorithm that works well with large datasets. Modified k-mean algorithm avoids getting into locally optimal solution in some degree, and reduces the adoption of cluster -error criterion. Algorithm: Modified approach (S, k), S={x1,x2,…,xn } Input: The number of clusters k1(k1> k) and a dataset containing n objects(Xij+). Output: A set of k clusters (Cij) that minimize the Cluster - error criterion.

Algorithm 1. Compute the distance between each data point and all other data- points in the set D 2. Find the closest pair of data points from the set D and form a data-point set Am (1<= p <= k+1) which contains these two data- points, Delete these two data points from the set D 3. Find the data point in D that is closest to the data point set Ap, Add it to Ap and delete it from D 4. Repeat step 4 until the number of data points in Am reaches (n/k) 5. If p<=p<=k) find the arithmetic mean of the vectors of data points Cp(1<=p<=k) in Ap. Select nearest object of each Cp(1•<=p<=k) as initial centroid. Compute the distance of each data-point di• (1<=i<=n) to all the centroids cj (1<=j<=k+1) as d(di, cj) For each data-point di, find the closest• centroid cj and assign di to cluster j Set Clustered[i]=j; // j:Id of the closest• cluster Set Nearest_Dist[i++]= d(di, cj)• For each cluster j (1•<=j<=k), recalculate the centroids Repeat•

Algorithm B 1. For each data-point di Compute its distance from the centroid of• the present nearest cluster If this distance is less than or equal to the• present nearest distance, the data-point stays in the cluster Else ; For every centroid cj (1•<=j<=k) Compute the distance (di, cj); End for Assign the data-point di to the cluster with the nearest centroid Cj Set Clustered[i] =j• Set Nearest_Dist[i] = d (di, cj); End for

The m_k -means Algorithm Input: a set D of d-dimensional data and an integer K. Output: K clusters begin randomly pick K points ∈D to be initial means; while measure M is not stable do begin compute distance dkj = ||xj – zk||2 for each k, j where $1 \leq k \leq K$ and $1 \leq j \leq N$, and determine members of new K subsets based upon minimum distance to zk for $1 \leq k \leq K$; compute new center zk for $1 \leq k \leq K$ using (3); compute M; end end The above algorithm reveals that the new clustering scheme is exactly similar to the original k-means algorithm except the only difference at the center computation step. In the following subsection, we shall try to prove that the m_k-means algorithm converges to kmeans centers and the rate of convergence is almost equal to that of the original k-means algorithm.



The K-means algorithm finds the predefined number of clusters. In the practical scenario, it is very much essential to find the number of clusters for unknown dataset on the runtime. The fixing of number of clusters may lead to poor quality clustering. The proposed method finds the number of clusters on the run based on the cluster quality output. This method works for both the cases i.e. for known number of clusters in advance as well as unknown number of clusters. The user has the flexibility either to fix the number of clusters or by input the minimum number of clusters required. In the former case it works same as K-means algorithm. In the latter case the algorithm computes the new clusters by incrementing the cluster counter by one in each iteration until it satisfies the validity of cluster quality threshold. The modified algorithm is as follows: Input: k: number of clusters (for dynamic clustering initialize k=2) Fixed number of clusters = yes or no (Boolean). D: a data set containing n objects. Output: A set of k clusters. Method:

1. Arbitrarily choose k objects from D as the initial cluster centers.

2. Repeat.

3. (re)assign each object to the cluster to which the object is most similar, based on the mean value of the objects in the cluster

4. Update the cluster means, i.e. calculate the mean value of the objects for each cluster.

5. until no change.

6. If fixed_no_of_clusters =yes goto 12.

7. Compute inter-cluster distance using Eq.2

8. Compute intra-cluster distance using Eq. 3.

9. If new intra-cluster distance < old_intra_cluster distance and new_inter- cluster >old_inter_cluster distance goto 10 else goto 11

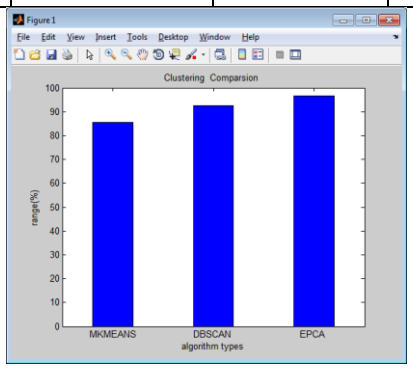
10. k= k + 1 goto step 1. k= 1 , 2 , ……K-1 and kk = k+1, ……,K

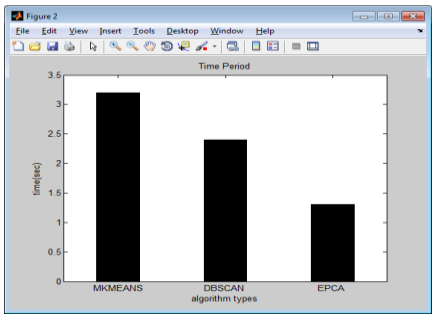
11. STOP dynamic clustering of data with modified K-means Algorithm.

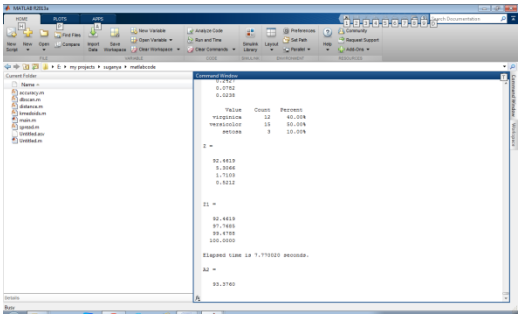
4. EXPERIMENTAL RESULTS

We evaluated the proposed algorithm on the data sets from UCI machine learning repository [9]. We compared clustering results achieved by the k-means, PCA+Mk-means with random initialization and initial centers derived by the proposed algorithm.

S.No	Algorithm	Accuracy	Time period
1	MKMEANS	88.4	3.2
2	DBSCAN	92.5	2.6
3	EPCA	95.6	1.8







5. CONCLUSIONS

The main objective of applying EPCA on original data before clustering is to obtain accurate results. But the clustering results depend on the initialization of centroid. In this paper, we have proposed a new approach to initialize the centroid and reducing the dimension using principal component analysis to improve the accuracy of the cluster results and the standard Mk-means algorithm also modified to improve the efficiency by reducing the computation complexity of the algorithm. The experiment results show that the substantial improvement in running time and accuracy of the clustering results by reducing the dimension and initial centroid selection using EPCA. Though the proposed method gave better quality results in all cases,

over random initialization methods, still there is a limitation associated with this, i.e. the number of clusters (k) is required to be given as input. Evolving some statistical methods to compute the value of k, depending on the data distribution is suggested for future research. In the future, we plan to apply this method to microarray cancer datasets.

REFERENCES

[1] Mishra, S. Et al. : Pattern Discovery in Hydrological timeseries data mining during the monsoon period of the high flood years inBrahmaputra River basin, IJCA, vol. 67, no. 6., 2013.
[2] Rodrigues, P. et al.: ―Hierarchical Clustering of Tim e Series Data Stream s,‖ IEEE Transactions on Knowledge and Data Engineering,vol. 20, no. 5, pp. 615-627, May 2008.
[3] Ramoni, M. et al.: Bayesian clustering by dynamics, Mach.Learning 47 (1) (2002) 91–121.
[4] Ramoni, M. et al.: Multivariate clustering by dynamics,Proceedings of the 2000 National Conference on Artificial Intelligence(AAAI-2000), San Francisco, CA, 2000, pp. 633–638.
[5] Wijk, J. J. van and Selow, E. R. Van: Cluster and calendar based visualization of time series data, Proceedings of IEEE Symposiumon Information Visualization, San Francisco, CA, October 25–26, 1999.
[6] Kumar, M. et al. :Clustering seasonality patterns in thepresence of errors, Proceedings of KDD ’02, Edm onton, Alberta,Canada.
[7] Vlachos, M. et al.: A wavelet based anytime algorithm for kmeans clustering of time series, Proceedings of the Third SIAMInternational Conference on Data Mining, San Francisco, CA, 2003.
[8] Li, C. And Biswas, G.: Temporal pattern generation usinghidden Markov model based unsupervised classification, Lecture Notes inComputer Science, vol. 164, IDA ’99, Springer, Berlin, 1999, pp. 245–256.
[9] Bicego, M. et al.: Similarity Based Clustering of Sequences Using Hidden Markov Models, Springer-Verlag Berlin Heidelberg, pp.86–95, 2003.