CHAPTER

Temporal Data Clustering 3

CHAPTER OUTLINE

3.1 INTRODUCTION

Since temporal data have been dramatically increasing, temporal data mining has drawn much more attentions than ever. As one of important mining tasks, clustering provided underpinning techniques for discovering the intrinsic structure

Temporal Data Mining via Unsupervised Ensemble Learning. http://dx.doi.org/10.1016/B978-0-12-811654-8.00003-8

and condensing information over large amount of temporal data. In this chapter, we present a comprehensive survey on temporal data—clustering algorithms from different perspectives, which include partitional clustering, hierarchical clustering, density-based clustering, and model-based clustering. Their strengths and weakness are also discussed for temporal data clustering tasks. Moreover, based on the internal, external, and relative criteria, most common clustering validity indices are described for quantitative evaluation of clustering quality.

3.2 OVERVIEW OF CLUSTERING ALGORITHMS

Clustering can be considered the most important unsupervised learning problem: it deals with finding structure within a collection of unlabeled data. A cluster is therefore a collection of objects which are "similar" among themselves and "dissimilar" to objects belonging to other clusters. Data are called static if the feature values do not change, or change only negligibly, with time, and many clustering algorithms have been developed for static data-clustering analysis. Approached using different criteria, the taxonomies of static-clustering algorithms are many and various. However, a common framework (Jain et al., 1999) is still widely accepted for classifying clustering algorithms into partitioning clustering, hierarchical clustering, density-based clustering, and model-based clustering.

Although various algorithms have been developed to cluster different types of temporal data, they all try to modify the existing algorithms for clustering static data. This is done in such a way that temporal data can be handled or converted into the form of static data, meaning that existing algorithms for clustering static data can be directly applied. The former approach usually works directly with raw temporal data and is thus called the proximity-based approach. The major modification lies in replacing the distance/similarity measure for static data with one appropriate to temporal data. The latter approach first converts raw temporal data into either a feature vector with lower dimensions or a number of model parameters. It then applies a conventional clustering algorithm to the extracted feature vectors or model parameters. These are called the feature-based and model-based approaches, respectively. Three categorized temporal data clustering approaches are summarized in Table 3.1.

3.2.1 PARTITIONAL CLUSTERING

Partitioning clustering directly divides the data sets into several subsets, where each subset represents a cluster containing at least one data. In general, the partition is hard or crisp if each data belong to exactly one cluster or soft or fuzzy if one data are allowed to be in more than one cluster at a different degree, where each cluster is represented by a prototype and assigns the patterns to clusters according to most similar prototype. K-means algorithm (Forgy, 1965) and its modified version of K-medoids algorithm (Kaufman and Rousseeuw, 1990) are quite

Table 3.1 A Taxonomy on Temporal Data Clustering Algorithms

Methodology	Model-Based Clustering	Proximity-Based Clustering	Feature-Based Clustering
Working mechanism	• Clusters of temporal data are specified by a mixture of dynamic models • Identify the data dependency and regularity behind the dynamic behaviors of temporal data	• Works directly on temporal data • Similarity measure considering temporal relations	• A set of features are extracted from raw temporal data • All existing clustering algorithms can be applied directly on the feature space
Advantage	• Generally suitable for coping with data dependency among temporal data • Temporal data characterized with generative models	• Prevents loss of any information • A direct way to capture the dynamic behaviors • Flexible means to deal with variable length of temporal data	• Significantly reduces the computational cost • Compatible with existing static data clustering algorithms
Disadvantage	• Model selection problem • High computational complexity	• Sensitive to initialization • Model selection problem • High computational complexity	• Loss of useful information conveyed in the original temporal data • Model selection problem
Example	Hidden Markov Model—HMM (Smyth, 1999), dynamic Bayesian networks (Murphy, 2002), autoregressive moving average model (ARMA) (Xiong and Yeung, 2002)	Dynamic time warping—DTW (Keogh and Kasetty, 2003), temporal K-mean/hierarchical (Jain et al., 1999; Xu and Wunsch, 2005)	Polynomial/spline curve fitting (Liu and Brown, 2004), adaptive piecewise constant approximation (Sahouria and Zakhor, 1997), curvature-based PCA segments (Cheong et al., 2005), multiple-scaled i-k mean (Lin et al., 2004)

popular partitional-clustering algorithms, where each cluster is represented by either the mean value of the data points in the cluster or the most centrally located data points in a cluster. Two counterparts for fuzzy partitions are the fuzzy c-means algorithm (Bezdek, 1981; Hoeppner, 1999) and the fuzzy c-medoids algorithm (Krishnapuram et al., 2001). Actually there are many possible outputs obtained by partitioning the data sets into several groups, partitional-clustering algorithms

always attempt to achieve a desired result by optimizing a criterion function such as square error, which is defined either globally or locally. These heuristic algorithms work well for finding spherical-shaped clusters and small to medium data sets, but they always reveal the weakness of analyzing the complex structured data such as temporal data.

K-means

It is one of the simplest partitional clustering algorithms and commonly used for solving temporal data clustering problem, for example, Liao et al. (2002) directly applied K-means as a proximity-based approach to multivariate battle simulation temporal data with the objective to form a discrete number of battle states or Vlachos et al. (2003) indirectly applied K-means as feature-based approach for analyzing time series on the wavelet-based representation. The procedure of K-means follows a simple way to classify a given temporal data set with a pre-defined number of clusters (assume K clusters), which it consists of the following steps:

1. Place K seed points into the representation space obtained from the data sets that are being clustered. These points represent initial groups.
2. Assign each data point to the group that has the closest seed point.
3. When all data points have been assigned, recalculate the positions of the K seed points.
4. Repeat steps 2 and 3 until the seed points no longer move. This produces a separation of the entire data set into groups known as clusters.

The entire process can be formulated by minimizing an objective function

$$\sum_{k=1}^{K} \sum_{x \in C_k} D(x, C_k) \tag{3.1}$$

where D is distance metric based on the meaningful objective (described in Section 2.3) to compute the distance between a data point x belonging to cluster k and representative point of the cluster C_k such as center of clustered data points.

Although the K-means can be proved that the learning procedure will always terminate at certain point, this algorithm does not necessarily find the most optimal solution, corresponding to the local minimum of objective function and sensitivity to the initialization and selection of number of seed points. Moreover, by directly applying K-means, it requires the temporal data with equal length because the concept of cluster centers would be ill defined when the individual one is represented in arbitrary length in the target data set.

Hidden Markov Model-Based K-Models Clustering

Essentially, Hidden Markov Model (HMM)-based K-models clustering is a K-means algorithm based on generative model−based representation, which has been commonly used for time series clustering task (Smyth, 1997; Zhong and Ghosh, 2003; Frossyniotis et al., 2004; Panuccio et al., 2009) due to its outstanding ability

of capturing the dynamic behavior of time series. For partitional clustering, the entire data set $X = \{x_1, x_2, \ldots x_N\}$ is represented as a set of K HMM $\{\lambda_1, \lambda_2, \ldots \lambda_K\}$ with M states. As mentioned in Section 2.2.3, the emission probability function is defined as a single Gaussian distribution for the HMM-based K-models clustering implemented in our simulations. HMM model would consist of the parameters $\lambda = \{\pi, A, B\}$, where $\pi = \{\pi_j\}$, $A = \{a_{ij}\}$, $B = \left\{\mu_j, \sigma_j^2\right\}$. Therefore, the entire process of the algorithm aims to estimate the optimal parameters of components models with maximum log-likelihood.

Given the number of clusters K and the similarity measure based on model-based distance metric such as log-likelihood, the procedure of HMM-based K-models follows these steps:

1. Randomly select K items for data sets without replacements.
2. Initialize K HMM with predefined number of states M, and estimate the parameters of each HMM on one of selected items by expectation-maximization (EM) algorithm.
3. Compute log-likelihood of each item under K HMM by the forward and backward algorithms.
4. Assign each time to the HMM with maximum log-likelihood.
5. Re-estimate the parameters of K HMM on the corresponding cluster of item by EM algorithm.
6. Repeat steps 3 to 5 until the cluster memberships no longer change.

Although HMM-based K-models are a simple and efficient approach that solve temporal data clustering problems, this algorithm does not necessarily find the most optimal model configurations and is also quite sensitive to the initialization and selection of seed points number.

3.2.2 HIERARCHICAL CLUSTERING

Also known as a tree of clusters or a dendrogram, hierarchical clustering (Johnson, 1967) builds a cluster hierarchy in which every cluster node contains child clusters. These sibling clusters separate the points covered by their common parent allowing for the exploration of temporal data on different levels of granularity. In the early work (Van Wijk and Van Selow, 1999), they just performed an agglomerative hierarchical clustering of daily power consumption data based on the root mean-square distance. Hierarchical clustering methods are generally categorized as either agglomerative (bottom-up) or divisive (top-down). Agglomerative clustering begins with one-point (singleton) clusters and recursively merges two or more appropriate clusters. Divisive clustering begins with one cluster of all data points and recursively splits the most appropriate cluster. The process continues until a stopping criterion (frequently, the requested number K of clusters) is achieved. Theoretically, divisive hierarchical clustering is unfeasible because the possible divisions of data into two clusters at the first step of the algorithm are quite various. Therefore, in most

applications, divisive hierarchical clustering is rarely applied which generally restricts attention to agglomerative hierarchical clustering.

Single Linkage

The simplest agglomerative hierarchical clustering approach is single linkage or the "nearest neighbor" technique. Key to this method is that the distance between groups is defined as the distance between the closest pair of objects came from different groups. In the single-linkage method,

$$D(R, S) = \min(D(r_i, s_j)) \tag{3.2}$$

where object r_i is in cluster R and object s_j is in cluster S. The distance between every possible object pair (r_i, s_j) is computed, The minimum value of these distances is said to be the distance between clusters R and S. In other words, the distance between two clusters is given by the value of the shortest link between the clusters. At each stage of hierarchical clustering, the clusters R and S, for which $D(R, S)$ is minimum, are merged.

The single-linkage method works well on data sets containing nonisotropic clusters including well-separated, chain-like, and concentric clusters. Additionally, this method may be useful for the detection of outliers in the data set.

Complete Linkage

Complete linkage, or the "farthest neighbor" technique, is the opposite of single-linkage technique defining the distance between groups as the distance between the most distant pair of objects, one from each group. In the complete linkage method:

$$D(R, S) = \max(D(r_i, s_j)) \tag{3.3}$$

where object r_i is in cluster R and object s_j is in cluster S. The distance between every possible object pair (r_i, s_j) is computed. The maximum value of these distances is said to be the distance between clusters R and S. In other words, the distance between two clusters is given by the value of the longest link between the clusters. At each stage of hierarchical clustering, the clusters R and S, for which $D(R, S)$ is minimum, are merged.

The complete linkage method produces a clustering result with smaller, tighter, and more compact clusters. Most of time, it is difficult to deal with complex cluster structures, and may gives a poor clustering result for temporal data.

Average Linkage

Here, the distance between two clusters is defined as the average of distances between all pairs of objects, where each pair is made up of one object from each group. In the average linkage method:

$$D(R, S) = T_{RS}/(N_R * N_S) \tag{3.4}$$

where T_{RS} is the sum of all pairwise distances between cluster R and cluster S. N_R and N_S are the sizes of the clusters R and S, respectively. At each stage of hierarchical clustering, the clusters R and S, for which $D(R,S)$ is the minimum, are merged.

The average linkage method is a compromise between the single and complete linkage methods, which avoids the extremes of either large or tight compact clusters. Unlike other methods, the average linkage method has better performance on ball-shaped clusters in the feature space.

In general, the performance of an agglomerative hierarchical clustering method often suffers from its inability to adjust, once a merge decision has been executed. The same is true for divisive hierarchical clustering methods. Hierarchical clustering is not restricted to cluster time series with equal length. It is applicable to temporal data of unequal length as well if an appropriate distance measure such as dynamic time warping is used to compute the distance/similarity.

HMM-Based Agglomerative Clustering

HMM-based hierarchical agglomerative clustering was originally proposed by Smyth (1997). Initially, the N singleton HMM $\{\lambda_1, \lambda_2, \ldots \lambda_N\}$ is created and trained on each of items. Then the "closest" pair of clusters is recursively merged. It is quite important to choose the distance measure between the two clusters or models for hierarchical clustering. Conventional mergence of the two closest clusters (models) results in the largest log-likelihood $\log p(X|\lambda)$ (Ward, 1963; Fraley, 1999). Therefore, the distance can be defined as,

$$D(\lambda_i, \lambda_j) = \log p(X|\lambda_{before}) - \log p(X|\lambda_{after}) \tag{3.5}$$

where λ_{before} and λ_{after} are the entire set of parameters before and after merging two models (λ_i and λ_j), respectively.

However, this traditional model-based distance measure for model-based clustering is inefficient due to the fact of searching the closest pair of clusters needs to train the merged model for every pair and evaluate the resulting log-likelihood. Therefore, the Kullback-Leibler (KL) distance measure described in Section 2.3.3 is used for the single linkage, complete linkage, and average linkage methods of HMM-based hierarchical clustering. Although hierarchical agglomerative clustering avoids the problem of initialization, the distance comparison, mergence, and model estimation require high computation cost for temporal data clustering.

HMM-Based Divisive Clustering

HMM-based divisive clustering (Butler, 2003) is a "reverse" approach of HMM-agglomerative clustering, starting with one cluster or model of all data points and recursively splitting the most appropriate cluster. The process continues until a stopping criterion (pre-defined number K of clusters or models) is achieved. Initially, an HMM is trained on the whole data set. During each iteration of division, the "poorest-fit" cluster whose HMM gives the lowest likelihood to the items in this cluster would be split. This process is repeated until a stop criterion is reached or all the clusters become singletons.

Like HMM-based agglomerative clustering, this still suffers from high computational costs and model selection problems. Moreover, it is quite sensitive to initialization, due to the possible divisions of data into two clusters at the first step.

3.2.3 DENSITY-BASED CLUSTERING

Density-based clustering algorithms are designed to find the arbitrary-shaped clusters in data sets, a cluster is defined as a high-density region, which exceeds a threshold, separated by low-density regions in data space. Density-Based Spatial Clustering of Applications with Noise, DBSCAN (Ester et al., 1996) is a typical density-based clustering algorithm. The basic idea of DBSCAN is to iteratively grow a cluster of data points as long as the density in the "neighborhood" exceeds some threshold. Rather than producing a clustering explicitly, Ordering Points To Identify the Clustering Structure (Ankerst et al., 1999) computes an augmented cluster ordering for automatic and interactive cluster analysis. The ordering contains information that is equivalent to density-based clustering obtained from a wide range of parameter settings, thus overcoming the difficulty of selecting parameter values. For analyzing time series—contained significant noise, density-based clustering has been typically applied by Denton (2005) to identify and remove this noise by only considering clusters rising above a preset threshold in the density landscape, while Jiang et al. (2003) proposed a density-based hierarchical clustering to tackle the problem of effectively clustering time series gene expression data, where all objects in a data set are organized into an attraction tree according to the density-based connectivity, the clusters are then identified by dense areas.

Density-Based Spatial Clustering of Applications with Noise

The basic idea of DBSCAN is to iteratively construct a new cluster from a selected data point by absorbing all data points in its neighborhood. It is based on two main concepts: density reachability and density connectability (Ester et al., 1996). Both of these concepts depend on two input parameters of the DBSCAN clustering: the size of neighborhood ε defined as a semidiameter of neighborhood based on Euclidean distance and the minimum number of data points in a cluster $N^{\min Pts}$. The entire process of DBSCAN involves the following steps:

1. It starts with an arbitrary starting point that has not been visited. It then finds all the neighbor points within distance ε from the starting point.
2. If the number of neighbors is greater than or equal to $N^{\min Pts}$, a cluster is formed. The starting point called core point and its neighbors are added to this cluster, and the starting point is marked as visited. Otherwise the point is marked as noise.
3. For each neighbor of core points in the current cluster, it is to mark it as visited. If the number of its neighbors is greater than or equal to $N^{\min Pts}$ and its neighbors which are not already contained in the current cluster are added to the current cluster, this process continues until the cluster cannot be expanded.

4. Then the algorithm proceeds to iterate through the remaining unvisited points in the data set.

In general, DBSCAN is beneficial from automatically detecting the number of clusters, ability to find arbitrarily shaped clusters, and relatively insensitive to noise and ordering of points in the database. However, it has the limitation to dealing with the high-dimensional data such as time series and has a difficulty to cluster data sets with large differences in densities.

3.2.4 MODEL-BASED CLUSTERING

In model-based clustering, we use certain models for describing groups of data points, and each cluster can be mathematically represented by a parametric model, such as HMM or autoregressive moving average model. The entire data set is therefore modeled by a mixture of these component models. An component model is used to represent a specific cluster that is often referred to a probability distribution. A large amount of literature (Picone, 1990; Smyth, 1997; Beran and Mazzola, 1999; Oates et al., 1999; Fraley and Raftery, 2002; Ramoni et al., 2002; Xiong and Yeung, 2002; Bagnall and Janacek, 2004; Pavlovic, 2004; Panuccio et al., 2009) have shown that the model-based approaches have been widely used for time series clustering analysis.

For a data set of N objects $x = \{x_n\}_{n=1}^{N}$, a mixture of models is defined as a linear combination of K component distributions and formulated as,

$$p(x|\theta) = \sum_{k=1}^{K} w_k \, p(x|\theta_k) \tag{3.6}$$

where $\theta = \{\theta_k\}_{k=1}^{K}$ is the unknown model parameter, w_k is the prior probability (also known as mixing or weighting coefficient), and satisfies the requirements $0 \leq w_k \leq 1$, and $\sum_{k=1}^{K} w_k = 1$. K is the number of component models representing the entire data set. In the model-based clustering, finding the clusters of a given data set is equivalent to estimating the parameters θ_k of each component distribution. The maximum likelihood (ML) estimation (Bilmes, 1998) is a useful statistical approach for parameter estimation, which finds the optimal parameters by maximizing a likelihood function derived from the observed data.

$$p(\{x_1, \ldots, x_N\}|\theta) = \prod_{n=1}^{N} \left(\sum_{k=1}^{K} w_k \, p(x_n|\theta_k) \right) \tag{3.7}$$

or in a logarithm form

$$l(\theta) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{k} w_k \, p(x_n|\theta_k) \right) \tag{3.8}$$

However, the computation of maximizing the log-likelihood is quite complex in most circumstances. Therefore, a powerful technique called EM algorithm (Dempster et al., 1977; Bilmes, 1998; Fraley and Raftery, 2002) is introduced in order to optimize such computation involved.

EM Algorithm

The EM algorithm is an iterative approach for ML parameter estimation from the observable data $\{x_n\}$ as incomplete data. Then the cluster label $\{y_n\}$ would be treated as missing data. Both observable data $\{x_n\}$ and cluster label $\{y_n\}$ construct the complete data. Thus, the complete data log-likelihood is formulated as,

$$l(\theta) = \sum_{n=1}^{N} \sum_{k=1}^{k} p(y_n|x_n) \log(w_k p(x_n|\theta_k)) \tag{3.9}$$

The posterior probability $p(y_n|x_n)$ represents the probability that a component model generating the data point x_n is signed by label y_n, which is defined as

$$p(y_n|x_n) = \frac{w_{y_n}\, p\left(x_n|\theta_{y_n}\right)}{\sum\limits_{k} w_k\, p(x_n|\theta_k)} \tag{3.10}$$

In producing a series of parameter estimations $\theta^t = \left\{\theta^0, \theta^1, \ldots, \theta^T\right\}$ until the convergence criterion is met, EM algorithm iteratively invokes the following steps:

1. Initialize $\theta^0 = \left\{\theta_k^0\right\}_{k=1}^{K}$ and set $t = 0$.
2. E-step: estimation of the posterior probability of missing data.

$$p(y_n = k|x_n) = \frac{w_{y_n}\, p\left(x_n\middle|\theta_{y_n}^t\right)}{\sum_k w_k\, p(x_n|\theta_k)} \tag{3.11}$$

3. M-step: re-estimation of model parameters.

$$\theta_{y_n}^{t+1} = \arg\max_{\theta} \sum_{n=1}^{N} p(y_n|x_n) p(x_n|\theta) \tag{3.12}$$

For model-based clustering, EM algorithm has the advantage (Bilmes, 1998) of conceptual simplicity and ease of implementation of estimating the model parameters. It iteratively optimizes the log-likelihood-based objective function, and has good rate of convergence on the first few steps. However, this approach cannot always guarantee to provide the best solution, and sometimes converges to a local optima. Moreover, its process always becomes quite slow for dealing with the temporal data with large volume and high dimensionality.

HMM-Based Hybrid Partitional-Hierarchical Clustering

As mentioned in the early sections, both HMM-based K-models and HMM-based hierarchical clustering have their own characters. In order to combine the strengths of model-based partitional and hierarchical approaches, Zhong and Ghosh (2003) proposed an HMM-based hybrid partitioning-hierarchical clustering.

For the hybrid approach, the whole data set is initially partitioned into K_0 clusters (K_0 is greater than the intrinsic number of clusters K) by HMM-based K-models. Then the flat K_0 clusters are used as the input of HMM-based agglomerative clustering and the closed clusters based on a specified distance measure are iteratively merged until the stop criterion is reached. The entire procedure is presented by following steps:

1. Flat partitional clustering: partition data objects into K_0 clusters by using HMM-based K-models clustering.
2. Distance calculation: compute pairwise intercluster distances using distance measures based on log-likelihood and identify the closest cluster pair.
3. Cluster merging: merge the two closest clusters and re-estimate a model for the merged data objects.
4. Stop if all data objects have been merged into one cluster or if a user-specified number of clusters is reached. Otherwise, go back to second step.

Although the HMM-based hybrid partitional-hierarchical clustering is able to reduce complexity of applying agglomerative clustering by using a preprocess based on partitional clustering, this hybrid clustering approach still suffers from the initialization problem and model selection problem of detecting the intrinsic number of clusters.

HMM-Based Hierarchical Metaclustering

The HHM-based hierarchical metaclustering (Zhong and Ghosh, 2003) is a modified version of hybrid clustering described previously. It treats each initial cluster obtained from the flat partitional clustering as metadata and applies standard HMM-based agglomerative clustering to group the meta data. Therefore, the composite model $\lambda_{i,j}$ obtained from the pair of merged clusters λ_i and λ_j is defined as combined parameters of children $\lambda_{i,j} = \{\lambda_i, \lambda_j\}$, and the log-likelihood of the composite model on the data associated with both merged clusters is defined as $\log p(X_{i,j}|\lambda_{i,j}) = \log p(X_i|\lambda_i) + \log p(X_j|\lambda_j)$, where $X_{i,j} = X_i \cup X_j$. For agglomerative clustering based on the KL distance measure (e.g., *Min*KL, *Max*KL, *Boundary*KL) detailed in Section 2.3.3, the distance between two composite models is defined as $D(\lambda_a, \lambda_b) = D^{KL}_{\lambda \in \lambda_a \& \lambda' \in \lambda_b}(\lambda, \lambda')$, where $\lambda_a = \{\lambda_{a1}, \lambda_{a2} \ldots\}$ *and* $\lambda_b = \{\lambda_{b1}, \lambda_{b2} \ldots\}$. This approach is implemented as following steps:

1. Flat partitional clustering: partition data objects into K_0 clusters using one of the model-based partitional clustering algorithms such as HMM-based K-models clustering.

2. Distance calculation: compute pairwise intercluster distances using distance measures based on log-likelihood and identify the closest cluster pair,
3. Cluster merging: merge the two closest clusters to form a composite model $\lambda_{i,j} = \{\lambda_i, \lambda_j\}$. The distance between two composite models $\lambda_a = \{\lambda_{a1}, \lambda_{a2}\ldots\}$ and $\lambda_b = \{\lambda_{b1}, \lambda_{b2}\ldots\}$ is defined as KL distance measure $D(\lambda_a, \lambda_b) = D^{KL}_{\lambda\in\lambda_a \& \lambda'\in\lambda_b}(\lambda, \lambda')$.
4. Stop if all data objects have been merged into one cluster or if a user-specified number of clusters is reached. Otherwise, go back to second step.

In comparison with original version, this approach further reduces computational cost due to the fact of that no re-estimation of merged models is required and well captures the character of complex structure of cluster than single model that is difficult to define and train by using a composite model represented by the parameter of its children models. However, the initial clustering analysis still causes the initialization problem by using HMM-based K-models and also requires a predefined number of clusters as an important input to the algorithm.

3.3 CLUSTERING VALIDATION

How to evaluate clustering results obtained by different clustering algorithms is the main subject of clustering validation. For a 2-3D clustering output space, the clustering result can be evaluated by the subjective visual inspection, which is often difficult and expensive for a large multidimensional data set such as temporal data. Other solutions are concerned with the clustering validation technique based on two criteria (Halkidi et al., 2001), namely the compactness and separation of cluster membership. For compactness, the members of each cluster should be as close to each other as possible. For separation, the clusters should be widely placed.

In order to access clustering quality based on the two criteria mentioned previously, three groups of cluster validity indices (Dunn, 1974; Davies and Bouldin, 1979; Sharma, 1995; Halkidi et al., 2001; Halkidi and Vazirgiannis, 2001) have been developed for quantitative evaluation of the clustering results based on external, internal, and relative measures (Theodoridis et al., 1999), respectively. The external measures require the class label (ground truth) to be known, where the clustering result generated by a clustering algorithm is compared to the prespecified partition of a data set based on the ground truth or the proximity matrix is compared to the prespecified partition. The internal measures are to verify whether the cluster structure produced by a clustering algorithm fits the data set, by using only information inherent to the data set. Both external and internal validation methods are based on statistical tests requiring high computational costs. However, the principal idea of the third approach, based on the relative criteria, is to determine the best clustering results generated from the same clustering algorithm but with different parameterization.

3.3.1 CLASSIFICATION ACCURACY

Classification accuracy as the simplest clustering quality measure was proposed by Gavrilov et al. (2000) to evaluate clustering results associated with the ground truth. Given the partition of the data set based on the ground truth $P^* = \{C_1^*, \ldots C_K^*\}$ and clustering results generated by clustering algorithm $P = \{C_1, \ldots C_K\}$, the similarity between them is formulated as

$$CA(P^*, P) = \left(\sum_{i=1}^{K} \max_j Sim(C_i^*, C_j) \right) \Big/ K \tag{3.13}$$

where $Sim(C_i^*, C_j) = 2\frac{|C_i^* \cap C_j|}{|C_i^*|+|C_j|}$ and K is the number of clusters.

3.3.2 ADJUSTED RAND INDEX

The Adjusted Rand Index (ARI) (Halkidi et al., 2002) is defined by

$$ARI(P^*, P) = \frac{\sum_{i,j}\binom{N_{ij}}{2} - \left[\sum_i \binom{N_i}{2}\sum_j \binom{N_j}{2}\right] \Big/ \binom{N}{2}}{\frac{1}{2}\left[\sum_i \binom{N_i}{2} + \sum_j \binom{N_j}{2}\right] - \left[\sum_i \binom{N_i}{2}\sum_j \binom{N_j}{2}\right] \Big/ \binom{N}{2}}. \tag{3.14}$$

Here, N is the number of data points in a given data set and N_{ij} is the number of data points of the class label $C_j^* \in P^*$ assigned to cluster C_i in partition P. N_i is the number of data points in cluster C_i of partition P, and N_j is the number of data points in class C_j^*. In general, an ARI value lies between 0 and 1. The index value is equal to 1 only if a partition is completely identical to the intrinsic structure and close to 0 for a random partition.

3.3.3 JACCARD INDEX

The Jaccard Index (Halkidi et al., 2002), also known as the Jaccard similarity coefficient is defined by

$$J(P^*, P) = \frac{\sum_{i,j}\binom{N_{ij}}{2}}{\sum_i \binom{N_i}{2} + \sum_j \binom{N_j}{2} - \sum_i \binom{N_{ij}}{2}}. \tag{3.15}$$

Here, N_{ij} is the number of data points of the class label $C_j^* \in P^*$ assigned to clusters C_i in partition P. N_i is the number of data points is in cluster C_i of partition P, and N_j is the number of data points in class C_j^*.

3.3.4 MODIFIED HUBERT'S Γ INDEX

Modified Hubert's Γ Index (MHΓ) (Theodoridis et al., 1999) is given by the following equation

$$MHT(P) = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} PM_{ij} Q_{ij} \tag{3.16}$$

where PM_{ij} is the proximity matrix, and Q is $N \times N$ cluster distance-based matrix based on partition P, where Q_{ij} is the distance between the centers of clusters to which x_i and x_j belong. In the MHΓ, the high value represents a compact and well-separated clustering structure of the partition.

3.3.5 DUNN'S VALIDITY INDEX

The Dunn's Validity Index (DVI) (Dunn, 1974) is given by the following equation

$$DVI\ (P) = \min_{i,j} \left\{ \frac{d(c_i, c_j)}{\max\limits_{k=1 \ldots K_m} \left\{ diam(c_k) \right\}} \right\} \tag{3.17}$$

where $[c_i, c_j, c_k] \in P$, $d(c_i, c_j)$ is the single-linkage dissimilarity function between two clusters and $diam(c_k)$ is the diameter of cluster, c_k, based on assessed partition, P. Similar to the MHΓ, DVI also assesses the clustering quality based on compactness and separation of clusters in the partition.

3.3.6 DAVIES-BOULDIN VALIDITY INDEX

The Davis-Bouldin Validity index (Davies and Bouldin, 1979) is a function of the ratio of the sum of within-cluster scatter and between-cluster separation

$$DB(P) = \frac{1}{K} \sum_{i,j=1}^{K} \max_{i \neq j} \left\{ \frac{Dist(Q_i) + Dist(Q_j)}{Dist(Q_i, Q_j)} \right\} \tag{3.18}$$

where K is the number of clusters. $Dist(Q_i)$ is the average distance of all objects from the cluster to their cluster center Q_i in partition PP and $Dist(Q_i, Q_j)$ is the distance between cluster centers (Q_i, Q_j). Hence, the ratio is small if the clusters are compact and far from each other. Consequently, the Davies-Bouldin index will have a small value for good clustering.

3.3.7 NORMALIZED MUTUAL INFORMATION

Normalized mutual information (NMI) (Vinh et al., 2009) is proposed to measure the consistency between any two partitions, which indicates the amount of information (common structured objects) shared between two partitions. Given a set of partitions $\{P_t\}_{t=1}^{T}$ obtained from a target data set, the NMI-based clustering validity criteria of

assessed partition P_a are determined by summation of the NMI between the assessed partition P_a and each individual partition P_m. Therefore, the high-valued NMI represents a well-accepted partition and indicates the intrinsic structure of the target data set. However, this approach always shows bias toward highly correlated partitions and favors the balanced structure of the data set. The NMI is calculated as following

$$NMI(P_a, P_b) = \frac{\sum_{i=1}^{K_a} \sum_{j=1}^{K_b} N_{ij}^{ab} \log\left(\frac{NN_{ij}^{ab}}{N_i^a N_j^b}\right)}{\sum_{i=1}^{K_a} N_i^a \log\left(\frac{N_i^a}{N}\right) + \sum_{j=1}^{K_b} N_j^b \log\left(\frac{N_j^b}{N}\right)} \tag{3.19}$$

$$NMI(P) = \sum_{t=1}^{T} NMI(P, P_t) \tag{3.20}$$

Here P_a and P_b are labelings for two partitions that divide a data set of N objects into K_a and K_b clusters, respectively. N_{ij}^{ab} is the number of shared objects between clusters $C_i^a \in P_a$ and $C_j^b \in P_b$, where there are N_i^a and N_j^b objects in C_i^a and C_j^b.

3.4 SUMMARY

Temporal data clustering is to partition an unlabeled temporal data set into groups or clusters, where all the sequences grouped in the same cluster should be coherent or homogeneous. Although various algorithms have been developed to cluster different types of temporal data, they all try to modify the existing clustering algorithms for processing temporal information.

Such approaches usually work directly with raw temporal data and are thus called the proximity-based approach. The major modification lies in the fact that the distance/similarity measure for static data is replaced by the one appropriate to temporal data. By directly applying the clustering algorithms to temporal data, most partitioning clustering suffer from model-selection problems and sensitivity to initialization, and hierarchical clustering has difficulties in properly grouping temporal data with high dimensionality. Density-based clustering is beneficial from automatically detecting the number of clusters, and in turn finding arbitrarily shaped clusters, but it still has the limitation of dealing with temporal data with high dimensionality and large differences in densities.

In contrast, other approaches are to convert the raw temporal data into either a number of statistical models or a feature vector with lower dimensions firstly and then apply a conventional clustering algorithm to the representative models or extracted feature vectors. These methods are often called the model-based and feature-based approaches. Although model-based clustering algorithms are able to model the dynamic behaviors of temporal data, there are also several drawbacks

such as computational costs and sensitivity to model configuration, which critically determines its performance. By indirectly applying the clustering algorithms to temporal data based on their representations, feature extraction as a vital process in feature-based clustering algorithms will always imperfectly represent all differently structured temporal data as each feature extraction accommodates a limited amount of characters obtained from temporal data. Therefore, we cannot solely rely on these algorithms for robust clustering analysis on temporal data. This, in turn, means that another advanced technique, the ensemble learning described in the later chapter, is required to improve the performance of temporal data clustering.

At the end of the chapter, most common clustering validity indices are described for quantitative evaluation of clustering quality based on the internal, external, and relative criteria. In addition, three clustering validity indices, including MHΓ, DVI, and NMI detailed in this chapter, would be used to constitute the basic concept of a partition weighting scheme for the proposed Weighted clustering ensemble with multiple representations presented in Chapter 7.