International Journal of u- and e- Service, Science and Technology
Vol.8, No. 8 (2015), pp.371-386
http://dx.doi.org/10.14257/ijunesst.2015.8.8.38

A Cognition-inspired System for Data Stream Clustering

Zhaoyang Sun[1*], K. Z. Mao[2], Wenyin Tang[2], Lee-Onn Mak[3], Kuitong Xian[1] and Ying Liu[1]

[1]*China National Institute of Standardization, Beijing, China*
[2]*Nanyang Technological University, Singapore*
[3]*DSO National Laboratories, Singapore*
**sunzhy@cnis.gov.cn*

Abstract

In applications such as target detection, domain knowledge of sensed data is often available. In this paper, we incorporate the available domain knowledge into clustering process and develop a knowledge-driven Mahalanobis distance-based ART (adaptive resonance theory) clustering algorithm. The strength of the knowledge-driven algorithm is that it can automatically determine the number of clusters with improved clustering results. The validity of the new algorithm has been verified on four artificial datasets. In addition, the algorithm has been adopted in our cognition-inspired system for clustering data stream, where known target library and dispersion of feature or attributes are available. The basic idea of this system is to divide data stream into frames, and to incorporate knowledge learned in previous frames into clustering of the following ones. Experimental studies have demonstrated that the evolving learning mechanism leads to improved clustering results compared with conventional incremental clustering algorithm Fuzzy ART and batch-based clustering algorithm k-means.

Keywords: *Data stream, Knowledge-based clustering, evolving learning, Mahalanobis distance, Cognition-inspired, Target detection*

1. Introduction

Clustering is an unsupervised machine learning method for finding groups or clusters underlying data. Clustering is often used to reduce data volume/dimension for easier handling, or to filter noise or fringe data for focusing of core information. According to the manner of data entry, clustering algorithms can be divided into two categories including batch-based clustering and incremental clustering. The batch-based clustering, such as k-means and fuzzy c-means clustering, processes data in a batch manner. While having advantages of simplicity and ease of implementation, the batch-based clustering has some limitations. Firstly, clustering results depend greatly on the initial guess of centers and the pre-assumed number of clusters underlying the data. Secondly, the batch-based clustering cannot effectively handle data stream with time-varying characteristics. The batch-based clustering assumes stationary behavior of data, but in many practical applications such as target detection and classification, the behavior and the numbers of clusters (targets) often changes with time.

The incremental clustering algorithms process data one by one in a sequential manner, and are suitable for handling data stream. In the literature, many clustering approaches have been proposed for data stream clustering, see for example [1-3]. BIRCH [1] is proposed for mining very large data sets and could be applied for data stream clustering. Basically, BIRCH can find a good clustering with a single scan of the data, but the clustering quality can be improved by a few additional scan. STREAM [2] is a single-pass streaming data clustering algorithm based on k-median. It tries to find k clusters with minimum sum of squared distances between the points and the cluster center to which

ISSN: 2005-4246 IJUNESST

they are assigned. During the execution, the algorithm continues to maintain a number of cluster centers and improve them by making local search. CluStream [3] is a stream clustering framework which is composed of online and offline components. The online component performs incremental clustering computation and maintains statistical summary information of micro-clusters. The offline component is utilized by analyst for different inquiries such as time horizon or number of clusters by using the concept of pyramidal time frame. Although the clustering methods mentioned above addressed some issues of data stream clustering such as scalability and time window, they still have some limitations [4], for example, most of them need a pre-determined number of clusters. In real application, it is not easy for user to provide that information.

Fuzzy ART [5] is another incremental clustering algorithm which could be applied for data stream. It groups data using a minimum required similarity between patterns within one cluster. While the incremental clustering algorithm fuzzy ART does not demand pre-determined number of clusters, it has its own limitations. First, cluster quality depends on the choice of the vigilance. A small value of vigilance may generate a small number of categories with large neighborhood of similarity, while a large vigilance value may generate a large number of categories with small neighborhood. Second, it is sensitive to the order of incoming data. Different initial points may generate different clustering results. Third, as a single-pass approach, fuzzy ART may perform unwell for data with time-changing behavior.

To overcome the limitations of traditional clustering algorithms, we develop a cognition-driven system to cluster data stream based on knowledge. The system is based on a new clustering algorithm we propose that incorporates contextual knowledge of dispersion level of features or attributes into ART (adaptive resonance theory) clustering process. The paper is organized as follows. The knowledge-driven ART clustering algorithm and related experiment are introduced in Section 2. Section 3 presents the framework and functions of the system. The system could not only incorporate contextual knowledge of dispersion level of features or attributes into clustering process but also learn knowledge from experience and apply the knowledge into clustering of following data. Experimental studies on a multi-target classification task have demonstrated the effectiveness of the proposed method. Section 4 describes how the presented algorithm and system are applied. Finally, we summarize our main contributions in Section 5.

2. Mahalanobis distance-based ART clustering Algorithm based on dispersion Level

2.1. The Motivation

Determination of the “true” number of groups in a data is probably the most difficult problem in cluster analysis [6]. From previous research and experiments, it is generally acknowledged that once the correct number of clusters is known, algorithms such as k-means or fuzzy c-means could always produce valid clustering results [7]. However, in real applications, it is even difficult to determine what is meant by the “true” number of clusters underlying the data. For example, Figure 1 depicts a set of data from nine Gaussian distributions introduced by Sugar *et al.*, [8]. Although the dataset consists of nine groups of mixed data, it could be divided into either three large clusters or nine small clusters. Methods such as cluster validation [9] can be used to decide the number of clusters. However, overlapping of data or different points of view of observation scale often prevents these methods from finding the correct number of clusters. In such a situation, additional information such as contextual knowledge of cluster dispersion might be helpful for the determination of the number of clusters.

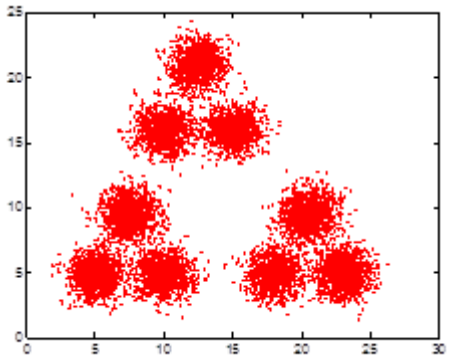


Figure 1. Dataset of Mixture Distribution

Dispersion level describes the expansion range of feature value of data. For example, it is assumed that the real height of a target is 180 cm. Precision of sensing device or observation error, and/or variability of the target may introduce noise to the measurement. The dispersion level just defines the extent to which the measured value could still be taken as the acceptable value under the introduced noise. That is, if the dispersion level is 1%, all values from the dispersion range (178.2~181.8) cm could be considered as the valid height of the target. If the dispersion level is 10%, all values from the dispersion range (162~198) cm are viewed as the valid value for the height of the target.

Motivated by the above analysis, we have developed an incremental clustering algorithm in this study by using prior knowledge concerning dispersion level of clusters. Our algorithm is based on the Euclidean ART neural network (EART) proposed by Kenaya *et al.*, [10]. EART replaces the fuzzy operations in Fuzzy ART neural network with Euclidean distance to measure pattern similarity and guide category learning. EART imposes the Euclidean distance threshold (*i.e.*, the radius of neighborhood) to decide the pattern membership. R is closely related to the number of clusters generated from the clustering process. If the given R is too small, proliferation of clusters could happen. On the other hand, if the given R is too large, dissimilar patterns might be grouped into the same cluster and this in turn decreases the purity of clusters. Our knowledge-driven clustering algorithm aims to determine the suitable value of R with the help of known knowledge so as to achieve improved clustering results. Details are described below.

2.2. The Algorithm

It is assumed that dispersion level is provided in the form of distribution variance or covariance, and is identical for all clusters. This assumption is reasonable in some applications. For example, when we measure the height of targets, although the values of height may be different, the noise brought into measurements might have identical statistics due to the similar conditions, such as the same tools, environment or observer. Figure 2 shows the flowchart of the knowledge-driven Mahalanobis distance-based ART clustering (KMART). Firstly, test pattern $\mathbf{x}$, known distribution variance or covariance $\mathbf{\Sigma}$ and R are presented to KMART. Then the Mahalanobis distance between $\mathbf{x}$ and center of different clusters is computed to find the closest cluster. If the minimal distance is less than or equal to R, the pattern is assigned to the closest cluster and the center of this cluster is updated. Otherwise a new cluster is generated. This learning process will be iterated until all patterns in the dataset are clustered.

In EART, Euclidean distance is used, which means spherical bound is generated for existing cluster and R could be considered as the radius of the sphere. If the distance between $\mathbf{x}$ and the spherical center is less than R, it is highly probable that $\mathbf{x}$ belongs to

the cluster. The further away it is from the center, the less likely that **x** belongs to the cluster. This approach assumes that the patterns are distributed around the center in a spherical manner, *i.e.*, no linear correlation exists between values of different features. In real applications, however, values of different features always have association, which means that data distributions are ellipsoidal. The probability of the test pattern belonging to the cluster depends not only on the distance from the center, but also on the direction. In those directions where the ellipsoid has a short axis the test point must be closer, while the axis is long the test pattern can be further away from the center. In order to fit the need of this situation, we employ Mahalanobis distance instead of the commonly used Euclidean distance in EART. Mahalanobis distance is defined as (1):

$$(\mathbf{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})' \le R^2 \quad (1)$$

where $\mathbf{x}$ is the test pattern, $\boldsymbol{\mu}$ is the center of existing cluster, $\boldsymbol{\Sigma}$ is the covariance matrix and R is the radius of the ellipsoid-like bound. If the covariance matrix $\boldsymbol{\Sigma}$ is an identity matrix, the Mahalanobis distance is reduced to the Euclidean distance. If the covariance matrix is diagonal, then the resulting distance measure is the normalized Euclidean distance.

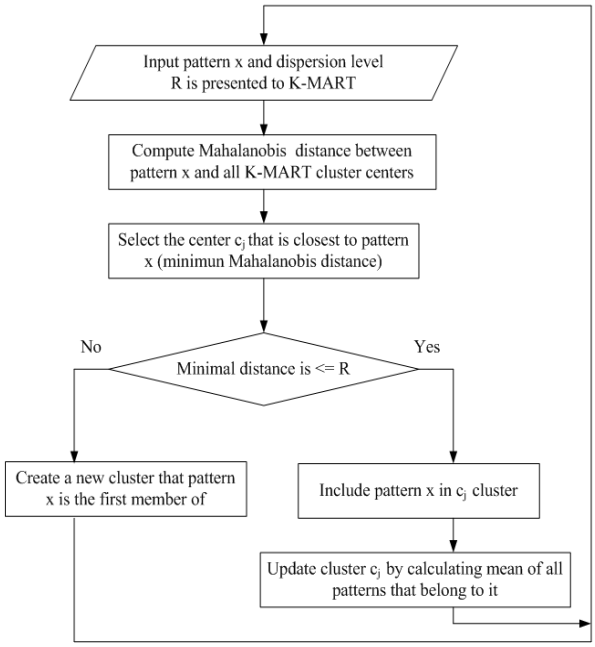


Figure 2. Algorithm Flowchart of KMART

2.3. Determination of Threshold *R*

Determination of *R* is the key of the KMART incremental clustering algorithm. Given the knowledge of dispersion level, how to determine a suitable value for *R*? As mentioned above, dispersion level is provided in the form of distribution variance or covariance. Therefore, we try to find the relationship between distribution variance (or covariance) and *R*. Take the univariate normal distribution shown in Figure 3 as an example. Probabilities are represented by area under the bell-shaped curve. If we choose $R = 3 * \sigma$, and then almost 99.73% data will be included in the cluster whose radius is *R*. Inversely, if we want 99.73% data to be included in the cluster, we should set $R = 3 * \sigma$. We try to find the similar relationship for multivariate normal distribution.

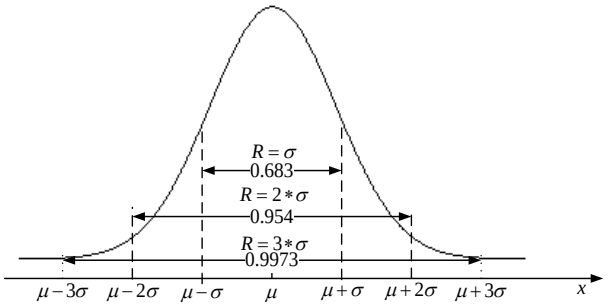


Figure 3. Relationship between R and σ in Univariate Normal Distribution

The univariate normal distribution with mean and variance has the following probability density function (2):

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-[(x-\mu)/\sigma]^2} \quad (2)$$

The term $\left[(x-\mu)/\sigma\right]^2 = (x-\mu)(\sigma^2)^{-1}(x-\mu)$ in the exponent of the univariate normal density function measures the square of the distance from x to μ in standard deviation units. It can be generalized for a p dimension normal distribution as $(\mathbf{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})'$. The $p \times p$ matrix $\boldsymbol{\Sigma}$ is the covariance matrix. We shall assume that the symmetric matrix $\boldsymbol{\Sigma}$ is positive definite, so the expression is the square of the generalized distance from $\mathbf{x}$ to $\boldsymbol{\mu}$, which is the so called Mahalanobis distance used in our algorithm. The multivariate normal density is obtained by replacing the univariate distance with the multivariate generalized distance in the density function (3) as follows:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}\left|\boldsymbol{\Sigma}\right|^{1/2}} e^{-(\mathbf{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})'/2} \quad (3)$$

where $\boldsymbol{\mu}$ is the mean vector and $\boldsymbol{\Sigma}$ is the covariance matrix. In the multivariate case, probabilities are represented by volumes under the surface over regions defined by intervals of the x_i values. The multivariate normal density is constant on surfaces when the square of the distance $(\mathbf{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})'$ is constant. This forms an ellipsoid centered at $\boldsymbol{\mu}$ whose surface is the probability density contour, where $\mathbf{x}$ satisfies (4)

$$(\mathbf{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})' = c^2 \quad (4)$$

In order to use the Mahalanobis distance to classify a test pattern as belonging to which clusters, one first estimates the covariance matrix of each cluster, usually based on patterns belonging to the cluster. However, in our algorithm, we assume covariance matrix for each cluster is identical and provided as knowledge (*i.e.*, dispersion level) beforehand. In multivariate normal distribution, the threshold for deciding the range of clusters is a $1 \times p$ vector *R*. Similar to univariate normal distribution, the value of *R* depends on covariance and probability. Because the distance has been normalized by known covariance in the left hand of (4), *R* should be normalized by the known covariance too, so the threshold is just equal to the constant c of the right hand of (4). Take univariate normal distribution for example, $c = R/\sigma$. If we want 99.73% data to be included in the cluster, we assign $c = 3$, given distance and *R* are both normalized by the known variance. For multivariate, $c^2 = \mathbf{R}\boldsymbol{\Sigma}^{-1}\mathbf{R}'$. In the following, c is used to represent the threshold for determining the membership of testing patterns, and its function is the same as *R* in the flowchart of Figure 2.

If the number of patterns is large enough, $c^2 = \chi_p^2(\alpha)$, where $\chi_p^2(\alpha)$ is the upper $(100\alpha)th$ percentile of a chi-square distribution with p degrees of freedom, leads to

contours that contain $(1-\alpha)\times 100\%$ of the probability. In general, if the number of samples are not very large, c is defined by the following equation[11]:

$$c^2 = \frac{p(n+1)(n-1)}{n(n-p)} F_{p,n-p}(\alpha) \tag{5}$$

where n is the number of samples, p is the number of dimensions, α is the confidence range defined by the user and $F_{n,n-p}(n, n-p, \alpha)$ is the F-distribution (cumulative distribution function).

Figure 4 is the example of relationship between c and confidence range for 100 patterns from 1 dimension to 10 dimensions. We can tell from the figure that the higher the dimensionality, the larger the radius so as to achieve the same confidence. For example, if we want 90% data to be included in the confidence ellipsoid, the radius for 2-dimensional data is 2.19, while the radius for 10-dimensional data is 4.31.

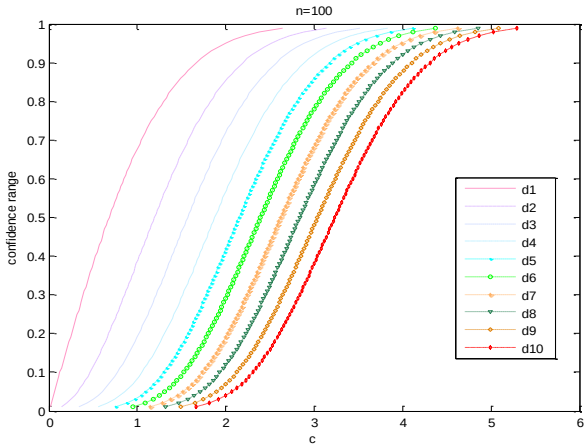


Figure 4. Relationship between Threshold c and Confidence Range for 100 Patterns from 1 Dimension to 10 Dimensions

2.4. Experiments

In experimental study, we tested the algorithm on four datasets, each of which is a mixture of nine 2-dimensional Gaussians but with different overlapping degree. To estimate the statistical power of the approach and find the best choice of c, we run the algorithm with different c, each for 100 times with randomly changed order of data each time. The goal of experimenting on artificial data is to test the adaptability of the proposed algorithm for data with different overlapping degree and to find the relationship between situations and choice of c.

We compare our method with the cluster validation method, Silhouette index (SI). SI is one of the most commonly used cluster validity indices for evaluating clustering results by using only features and information inherent in a dataset. It is used to estimate the optimal number of clusters in this experiment. The basic procedure for deciding number of clusters using SI has following steps [12].

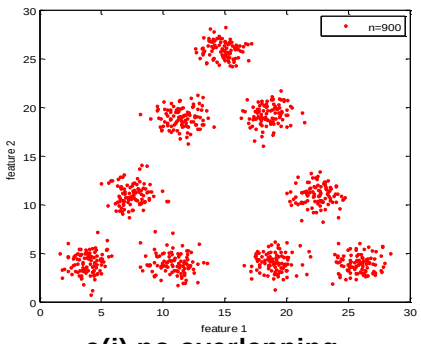
Run the clustering algorithm such as k-means or fuzzy c-means for different number of clusters (*i.e.*, $k = 2-10$), each for 100 times.

Compute SI for each runs and its average value of 100 times for different number of cluster.

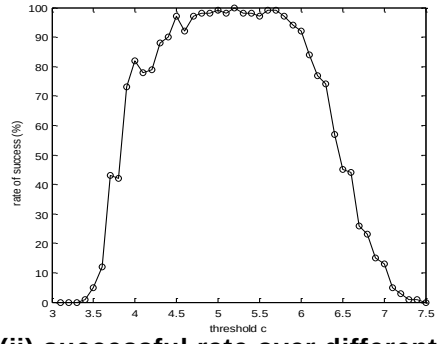
Estimate the number of clusters in the dataset by comparing the average value of SI. A larger SI value indicates a better quality of a clustering result and corresponding number of clusters is suggested.

The dataset illustrated in Figure 5a (i) contains well-separated clusters. The results of our algorithm are shown in Figure 5a (ii). It is observed that if we choose c from range

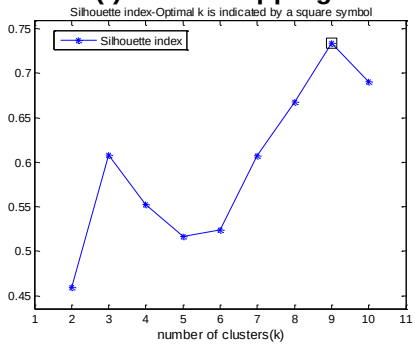
(3.95~6.4), the correct number of clusters could be found with chances of more than 50%. Especially, when $c \geq 4.5$, the chance of finding the correct number of clusters is high. Figure 5a (iii) shows that the optimal k suggested by SI is also correct.



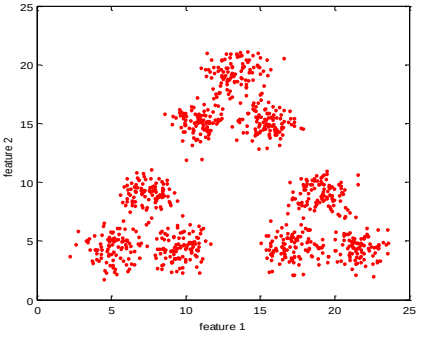
a(i) no overlapping



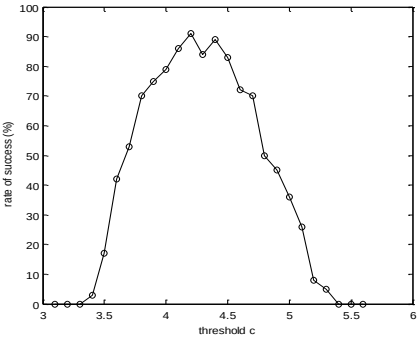
a(ii) successful rate over different c



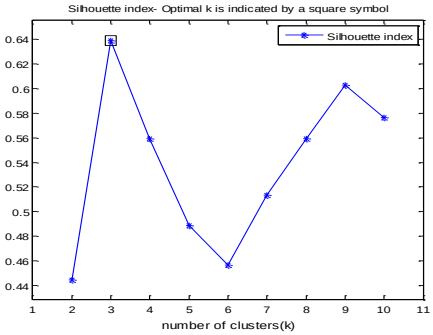
a(iii) optimal k suggested by SI



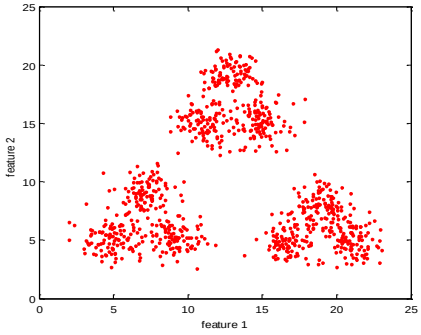
b(i) slight overlapping



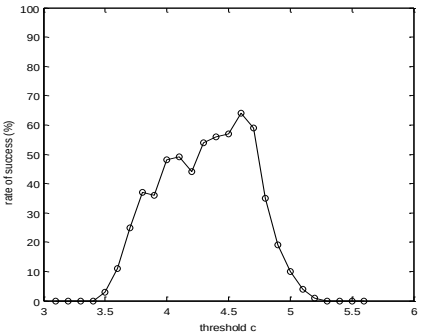
b(ii) successful rate over different c



b(iii) optimal k suggested by SI



c(i) moderate overlapping



c(ii) successful rate over different c

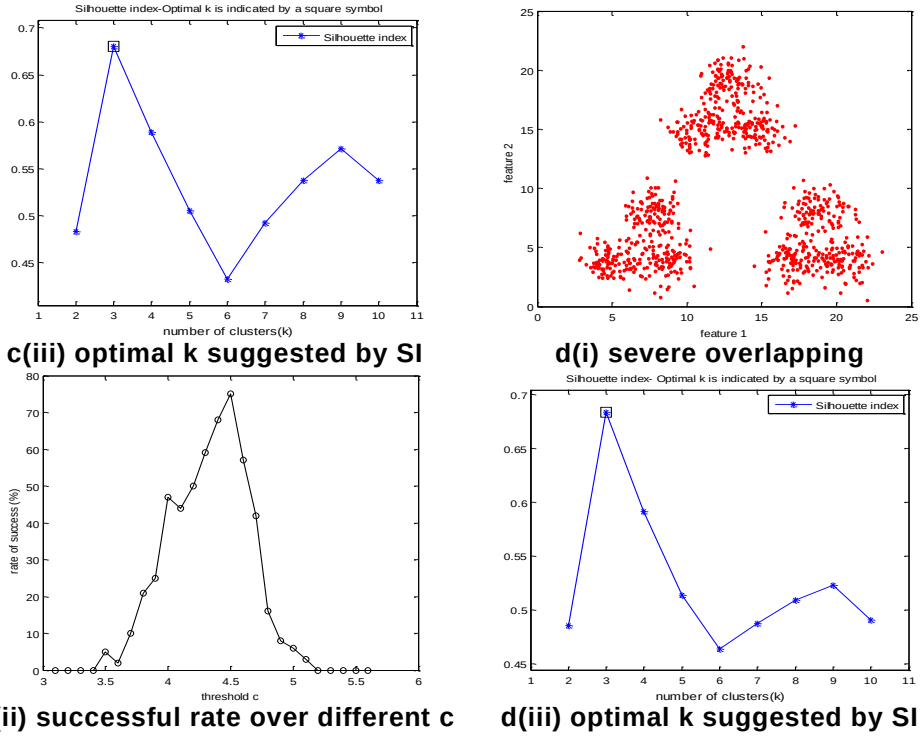


Figure 5. (i): Four Simulated Datasets, each with nine Gaussian clusters. (ii): Rate of Success using Proposed Clustering Algorithm for Each Dataset. (iii): the Optimal Number of Clusters (k) Suggested by SI

The dataset of Figure 5b (i) has slight overlapping. Figure 5b (ii) shows that if we choose c from range (3.76~4.88), our algorithm also finds the correct number of clusters with the probability higher than 50%. However, the probability will not be higher than 90%. The best probability is obtained when c is around 4.4. Figure 5b (iii) shows that the optimal k suggested by SI is 3. Meanwhile the SI for k=9 is the second high value, which means that the SI is confused about the situation when no additional information is accessible.

The dataset of Figure 5c (i) has moderate overlapping. Figure 5c (ii) shows that if we choose c from range (4.4~4.6), our algorithm can also find the correct number of clusters with the probability more than 50%. However, the probability will not be more than 70%. The best probability is achieved when c is around 4.6. Similarly, Figure 5c (iii) shows that the optimal k suggested by SI is 3 and the SI for k=9 is much lower than that of k=3.

The dataset of Figure 5d (i) has severe overlapping, which is almost indistinguishable from three clusters. As shown in Figure 5d (ii), only when we choose c from range (4.4~4.6), the probability of finding correct number of clusters is over 50%. Figure 5d (iii) shows that the optimal k suggested by SI is 3 and the SI for k=9 is much lower than that of k=3.

The experiment results show that our algorithm works well even when overlapping exists. As the overlapping degree of datasets increases, the range of acceptable c is shrunk. For two dimensional data, c=4.4~4.6 is suggested, and this means that $sqrt(\mathbf{R}\boldsymbol{\Sigma}^{-1}\mathbf{R}') = 4.4 \sim 4.6$.

3. The Cognition-driven System

The algorithm mentioned above is core of our human cognition-inspired target detection and classification system as shown in Figure 6. In this study, we propose a cognition-inspired system for evolving clustering using available knowledge. The system could not only incorporate contextual knowledge of dispersion level of features or

attributes into clustering process but also learn knowledge from data and use the learned knowledge to progressively improve clustering of following data.

3.1. Motivation

Human processes mass of information every day. Pattern recognition by human is fast, accurate and adaptive to changing environment. Human perform well because we learn, store and use knowledge to progressively improve our intelligence. The knowledge may be taught by teachers, which is called principle knowledge, or learned from experience. This learned knowledge could also be used for improvement of future behavior. It is a natural choice to enable knowledge integration in the clustering algorithm to improve the performance of clustering, just like human do.

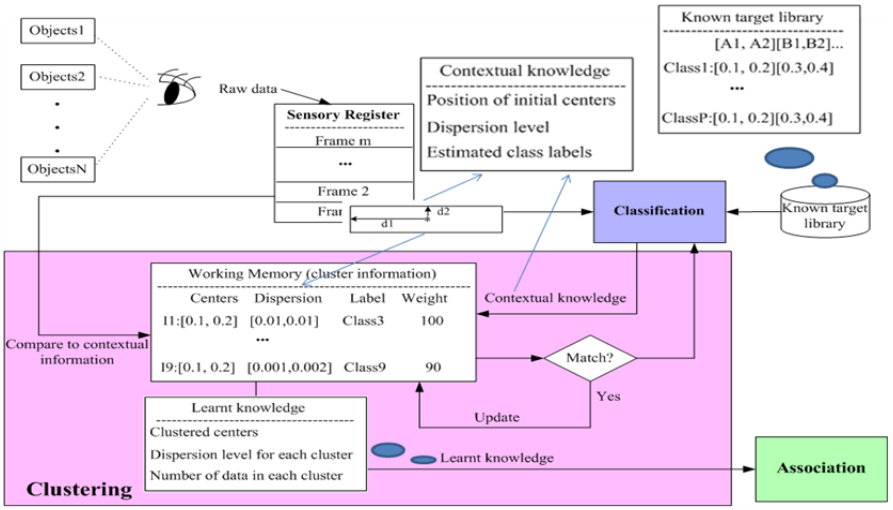


Figure 6. Block Diagram of the Cognition-driven Target Classification System

In general, clustering algorithms work in an unsupervised setting. Except for feature values describing data, no additional information (*e.g.*, labels) is presented to a clustering algorithm. But in some applications, domain knowledge owned by domain experts is often available. If used properly, the domain knowledge has the potential of facilitating clustering process and improving clustering results. However, traditional clustering algorithms do not have a mechanism to incorporate the domain knowledge into the clustering process. Knowledge-driven clustering has received attentions only in recent years. Domain knowledge available for clustering is often in the form of "true" similarity between pairs of objects, class labels for a small set of data, or supplementary information of the clusters such as the minimal or maximal size *etc*. Depending on the nature of available knowledge, a variety of knowledge-driven clustering algorithms have been developed in the literature. In Wagstaff and Cardie [13-14], knowledge is expressed as must-link and cannot-link, and pair-wise constraints are therefore imposed on cluster membership. Forestier *et al.*, [15] used class label and link-based constraints as background knowledge to facilitate clustering collaboration. Basu *et al.*, [16] used a set of labeled samples to initialize clusters for the k-means algorithm. Pedrycz [17] presented a semi-supervised clustering to collaborative and knowledge-based fuzzy clustering. Bradly *et al.*, [18] adopted cluster-level constraints to help clustering process. Because of the imposed constraint of minimum size of cluster, the modified k-means clustering algorithm could avoid empty clusters. Compared with traditional clustering, the research in knowledge-driven clustering is relative rare. This is because knowledge is not always available, and when available, the knowledge is often application specific.

3.2. Data Frame Clustering

Different from traditional clustering algorithms, the proposed evolving clustering algorithm KMART processes data in the form of frame. The incoming data stream is divided into frames with a pre-specified time window. A data frame is a set of all points that fall into a time window. As shown in Figure 6, these frames are stored in a sensory register and processed by the clustering algorithm one by one. The data stream of large volume arrives at the clustering component continuously, and it takes time to cluster them. A buffer is introduced to maintain the information until the clustering process is done. The buffer is the so-called register memory, whose name is borrowed from cognition psychology. In cognition psychology, the register memory is the initial memory system used to avoid the loss of present information while processing information that has just occurred. Compared to the batch-based and incremental clustering algorithms, the frame-based clustering method has two advantages: (1) Large amount of data are divided into small frames and this helps to reduce the complexity of computation and storage burden and hence could improve efficiency of the algorithm. (2) Frame-based data processing facilitates evolving learning of knowledge from the previous data and use of the learned knowledge for the clustering of the future data.

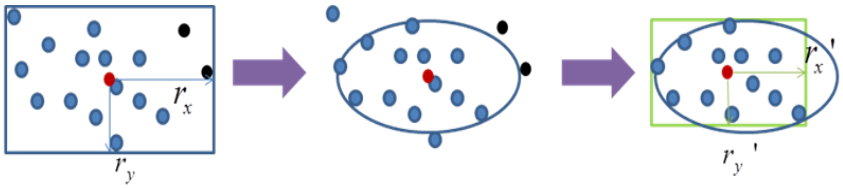
3.2.1. Clustering of the First Frame of Data: When the first frame of data is input to the system, the clustering algorithm KMART accesses the known target library (KTL) through the classification component. The KTL consists of known characteristics of targets acquired from long time observation or default parameters provided by the manufacturer. KTL records the value bounds (lower and upper bounds) of every feature for each target (class). Data in the first frame will match to the classes in KTL and be assigned to the most similar classes. This is similar to the k-means clustering, but the initial centers are generated from known KTL rather than picked randomly. The initial guesses of cluster centers are thus dramatically narrowed down by the knowledge of KTL. At the end of this process, all data in the first frame are clustered into corresponding groups. The center of a cluster is calculated by the average of data belong to this cluster. These centers are stored as the initial settings for the next frame. Besides the center information, the dispersion level of each cluster is also calculated and stored. After calculating the number of samples included in each cluster, the clusters with small number of samples will be deleted to avoid the effect of outlier. The variance of each class is then calculated and knowledge of dispersion level is acquired into the KTL. When computing the dispersion level, we may exclude noisy samples or outliers. In this study, we only choose 75% of data near to center to compute the dispersion level. Knowledge learned in this process is called contextual knowledge. Except for center and dispersion level of each cluster, the contextual knowledge also includes the number of samples and variance of each cluster. This contextual knowledge is used as the initial condition for the next frame of data and stored in the working memory (WM).

3.2.2. Clustering of the Subsequent Frame of Data: When the second frame of data enters the clustering algorithm, there is no need to visit KTL again. It is assumed that most clusters (or objects) appeared in the last frame of data will probably appear in the current data frame. Therefore, the distance between data and centers of last frame will be calculated first. If the dissimilarity between current data and one of the best fitting clusters does not exceed the dispersion level of that cluster, the data should be absorbed into that cluster and center of that cluster is updated accordingly. If the data cannot be absorbed by any existing cluster, a new cluster will be created and added to the working memory. In such a case, the new cluster is classified using KTL which is similar to the first frame. Again, knowledge

about current frame of data, *i.e.*, number of samples, variation, centers and dispersion level of each cluster, will be leaned and stored for the further use.

3.3. Outlier Handling

Outliers are handled by knowledge accumulation in our clustering system as shown in left part of Figure 7, where class 2 has 16 data and 2 of them are outliers. After clustering data in one frame, system will record some statistics information as learned knowledge, *i.e.*, number of samples, variation, center and dispersion level of each cluster. In order to handle outliers, as shown in middle part of Figure 7, only 75% data that are the closest to the current center are used to learn the knowledge. Based on normalization by deviations, an ellipse is generated to decide the similarity of data and the cluster. Data outside of the ellipse are considered as outliers. As shown in right part of Figure 7, the knowledge is re-calculated after outliers are excluded. Then new dispersion level is computed, which is represented by the rectangle as shown in Figure 7.



After clustering data in one frame, system will record some statistics information as learnt knowledge:
Centers: (x_0, y_0)
Dispersion: (σ_x, σ_y) (r_x, r_y)
Label: Class 2
Weight: 16

In order to handle outlier, only consider 75% data which are closest to the current center

$$d_i = \sqrt{\left(\frac{x_i - x_0}{\sigma_x}\right)^2 + \left(\frac{(y_i - y_0)}{\sigma_y}\right)^2}$$

Re-calculate the center, dispersion level and weight to validate the knowledge:
Centers: (x_0', y_0')
Dispersion: (σ_x', σ_y') (r_x', r_y')
Label: Class 2
Weight: 12

Figure 7. Outlier Handling

3.4. Overlapping Handling

Overlapping problems could be solved by learning from experience in our clustering system. As shown in Figure 8, class 2 and class 3 in frame 2 are overlapped. For traditional clustering algorithm, it's difficult to separate these two clusters without more information. In our system, it is possible because the dispersion level of class 2 (represented by the rectangle in Figure 8) has been learned in frame 1, therefore, most data of class 2 and class 3 in frame 2 will not be grouped to one cluster.

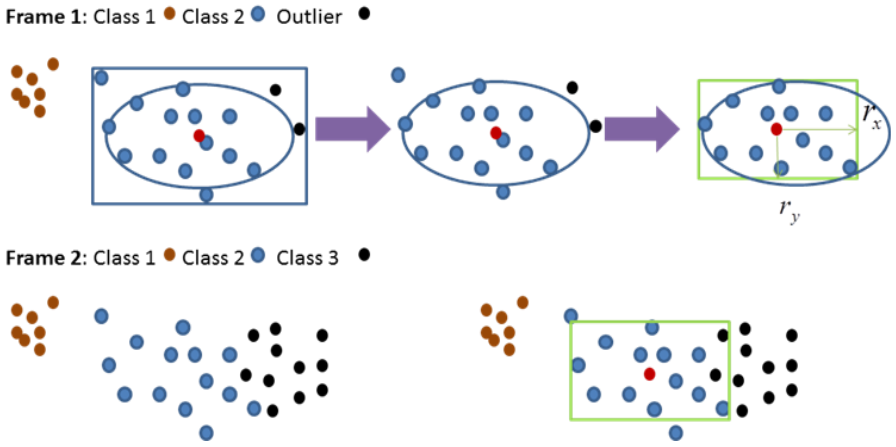


Figure 8. Overlapping Handling

3.5. Experimental Studies

In this section, we conduct experimental studies to evaluate the effectiveness of the proposed method for clustering of data stream. The data, the clustering evaluation criterion and the results are described below.

3.5.1. Data: The data is from a target detection and classification task. The data stream contains 47000 records from 16 targets. Each record consists of ten attributes including class label for validation. The data is divided into 47 frames, where each frame has 1000 data.

3.5.2. Rand Statistics as a Clustering Evaluation Criterion: Assuming $C = \{C_1 \cdots C_k\}$ is a resulting clustering structure of a dataset C, $P = \{P_1 \cdots P_n\}$ is the true partition of the data.

- *SS*- defined as the number of pairs of points that belong to the same cluster of the clustering structure *C* and to the same group of partition *P*.
- *SD*- the number of pairs of points that belong to the same cluster of *C* and to different groups of *P*.
- *DS*- the number of pairs of points that belong to different clusters of *C* and to the same group of *P*.
- *DD*- defined as the number of pairs of points that belong to different clusters of *C* and to different groups of *P*. The Rand statistics[19] is defined as(6):

$$R = \frac{SS + DD}{SS + SD + DS + DD} \tag{6}$$

where the value of *R* is between 0 and 1. High values of *R* indicate great similarity between *C* and *P*.

3.5.3. Results: Firstly, **the** evolving clustering algorithm is compared to Fuzzy ART clustering using the criteria mentioned above. The result is given in Figure 9.

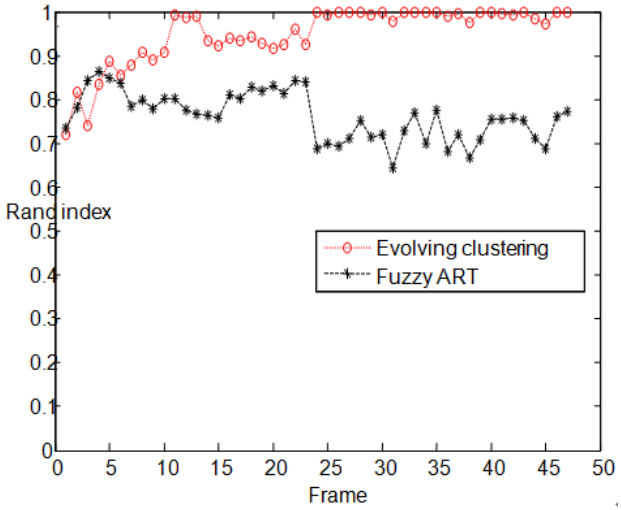


Figure 9. Rand Statistics Comparison between Evolving Clustering and Fuzzy ART

It is observed that at the early stage of the clustering process, both algorithm produces unsatisfactory performance. With more frames of data coming, evolving clustering algorithm performs better and better, while the results of fuzzy ART are not improved. The continuing improvement of the evolving algorithm is due to knowledge accumulation as well as the usage of accumulated knowledge for the coming data.

Secondly, the evolving clustering algorithm is compared to k-means clustering. Before clustering with k-means, the number of clusters needs to be defined beforehand. The real number of clusters for each frame is different, which are from 5 to 9, so k was selected in the range of 5 to 9. Due to the random nature of the k-means algorithm, the number of tests for each k was set as 100. The result is shown in Figure 10.

It is observed that even we narrowed down the range of k to 5-9, the performance of k-means is not comparable to the evolving clustering. The possible reasons may be as follows:

1) Searching the appropriate number of clusters for a given data set is generally a trial-and-error process. An inappropriate choice of k may produce poor results. It is more difficult when try to decide the 'correct' clusters conforming to the subjective nature. Knowledge-based evolving clustering, on the contrary, does not fixed on any predefined number of clusters. The "correct" number of clusters is neither required nor necessary. During the evolving learning process, new cluster is automatically added whenever feasible. This "add as it is needed" fashion could be more flexible, adaptive and more human-like.
2) Once the number of clusters is specified for k-means clustering, it remains fixed, even when new-coming data is evolving. While knowledge-based evolving clustering is able to learn from the knowledge of last frame and adapt its results accordingly, which makes it more suitable for time-varying data stream.

Using Euclidean distance as a measure of cluster scatter, incremental k-means clustering tends to yield spherical clusters. It means all the clusters are almost the same size, which may not accord with the true story. As shown in Figure 7, knowledge-based evolving clustering could produce different size of clusters by learning dispersion level of each cluster.

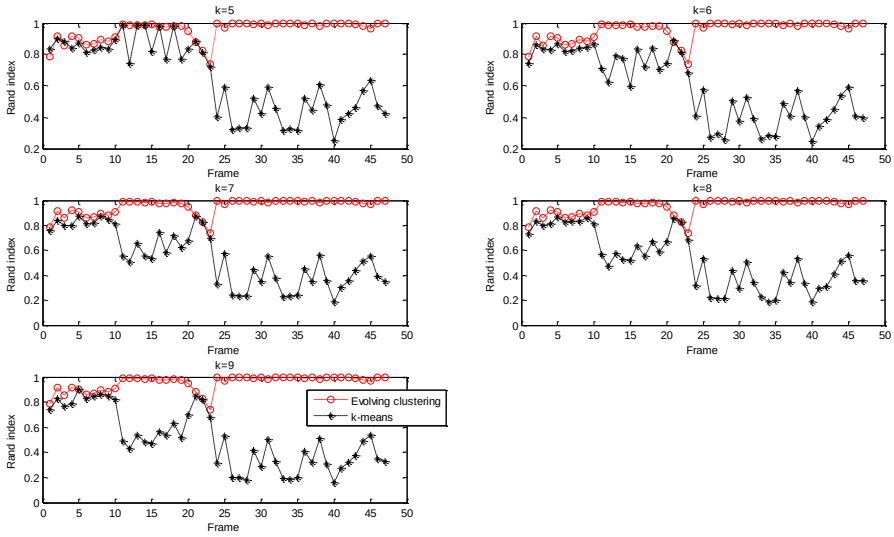


Figure 10. Rand Statistics Comparison between Evolving Clustering and K-means

4. Application

In the target detection system, data clustering is used for preprocessing [20]. As shown in Figure 11, data clustering in the system could access two types of domain knowledge: known target library (KTL) and feature dispersion information (FDI). The KTL consists of known characteristics of targets acquired from long period of observation or default parameters provided by the manufacturer [21]. It records the value bounds (lower and upper value) of every feature for each class in the KTL. Different class corresponds to different combination of FDI.

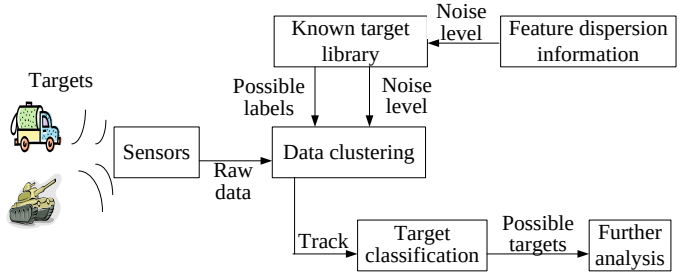


Figure 11. Target Detection System with Available Knowledge

In our application, the massive raw data collected from sensors are clustered first. Representatives with compressed size and typical value are output for further classification and identification. The goal of the knowledge-driven clustering algorithm in the above system is to integrate existing domain knowledge (*i.e.*, information in the KTL and attribute or feature) to produce accurate clusters of data so as to improve performance of target identification.

5. Conclusion

In this study, a knowledge-driven ART clustering algorithm has been developed. The experiment results show that with the help of contextual knowledge (*i.e.*, dispersion level), the proposed algorithm outperforms the methods without using contextual knowledge, even when overlapping exists.

In addition, the algorithm has been applied in our cognition-inspired target detection system, where known target library and feature dispersion information are employed to facilitate the clustering process.

Compared with fuzzy ART and k-means clustering algorithm, the proposed new method has two outstanding features:

1) The proposed algorithm is able to integrate expert knowledge to enforce a good starting point for clustering.
2) The proposed algorithm adopts an evolving learning mechanism that accumulates learned knowledge for continuing improvement of clustering performance.

ACKNOWLEDGEMENTS

This work was supported in part by the Chinese National Science and Technology Support Program (Grant No. 2012BAH35F01) and DSO grant (Grant No. DSOCLO9179).

References

[1] T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases", Proceedings of the 1996 ACM SIGMOD, **(1996)**.
[2] L. O'Callaghan, A. Meyerson, R. Motwani, N. Mishra and S. Guha, "Streaming-Data Algorithms for High-Quality Clustering", 18th International Conference on Data Engineering, **(2002)**.
[3] C. C. Aggarwal, T. J. Watson, R. Ctr, J. Han, J. Wang and P. S. Yu, "A Framework for Clustering Evolving Data Streams", Proceedings of the 29th International Conference on Very Large Data Bases, **(2003)**.
[4] A. R. Mahdiraji, "Clustering Data Stream: A Survey of Algorithms", International Journal of Knowledge-based and Intelligent Engineering Systems, vol. 13, **(2009)**.
[5] G. A. Carpenter, S. Grossberg and D. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system", Neural Network, vol. 4, **(1991)**.
[6] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms", 16th IEEE International Conference on Tools with Artificial Intelligence, **(2004)**.
[7] P. Berkhin, "Survey of Clustering Data Mining Techniques", Accrue Software, **(2002)**.
[8] C. A. Sugar and M. J. Gareth, "Finding the number of clusters in a data set: An information theoretic approach", Journal of the American Statistical Association, vol. 98, **(2003)**.
[9] M. Halkidi, "On clustering validation techniques", Journal of Intelligent Information Systems, vol. 17, **(2001)**.
[10] R. Kenaya and K. C. Cheok, "Euclidean ART neural networks", Proceedings of the World Congress on Engineering and Computer Science, **(2008)**.
[11] R. A. Johnson and D. W. Wichern, "Applied multivariate statistical analysis", Springer, **(2007)**.
[12] K. Wang and L. Peng, "CVAP: Validation for cluster analyses", Data science journal, vol. 8, **(2009)**.
[13] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints", ICML '00 Proceedings of the Seventeenth International Conference on Machine Learning, **(2000)**.
[14] K. Wagstaff, C. Cardie, S. Rogers and S. Schrödl, "Constrained k-means clustering with background knowledge", ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning, **(2001)**.
[15] G. Forestier, P. Gançarski and C. Wemmert, "Collaborative clustering with background knowledge", Data & Knowledge Engineering, vol. 69, **(2010)**.
[16] S. Basu, A. Banerjee and R. Mooney, "Semi-supervised clustering by seeding", Proceedings of 19th International Conference on Machine Learning, **(2002)**.
[17] W. Pedrycz, "Fuzzy clustering with a knowledge-based guidance", Pattern Recognition Letters, vol. 25, **(2004)**.
[18] P. S. Bradley, K. P. Bennett and A. Demiriz, "Constrained k-means clustering", Microsoft Research, **(2000)**.
[19] W. M. Rand, "Objective criteria for the evaluation of clustering methods", Journal of the American Statistical Association, vol. 66, **(1971)**.
[20] G. W Ng, "Intelligent systems-fusion, tracking and control", Research studies, **(2003)**.
[21] W. Tang, K. Z. Mao, L. O. Mak and G. W. Ng, "Classification for Overlapping Classes Using Optimized Overlapping Region Detection and Soft Decision", Proceedings of the 13th International Conference on Information Fusion, **(2010)**.

