K-MEANS-BASED FUZZY CLASSIFIER

Feng Zheng

Institute for Signal and Information Processing
Mississippi State University
Mississippi State, MS 39762 USA
email: zheng@isip.mstate.edu

ABSTRACT

In this paper, we propose a method based on the K-means algorithm to efficiently design a fuzzy classifier so that the training patterns can be correctly classified by the proposed approach. In this method, first we use the K-means algorithm to partition the training data for each class into several clusters, and the cluster center and the radius are calculated. Then, some fuzzy rules are proposed to construct a fuzzy classifier which can efficiently classify the given data to represent clusters belonging to the corresponding classes. The proposed method has the following features: (a) it does not need prior parameter definition; (b) it only needs a short training time; (c) it is simple. Finally, two data set are used for experiments to illustrate and examine the proposed method for the fuzzy classifier design.

1. INTRODUCTION

Pattern classification is an important element of many engineering problems such as speech recognition, handwriting recognition, diagnostic etc. Fuzzy classifiers provide a measure of the degree to which a pattern fits within a class. A fuzzy classifier with hyperbox regions has short training time for it only needs to calculate the maximum and minimum values of training data. But the classification accuracy rate of this method will decrease when there are identical data in different class. A fuzzy classifier based on genetic algorithm has long training time and the individual length becomes very long when the training data has high dimensions. A fuzzy classifier with ellipsoidal region has good performance in many classification problems, but it needs vast time to calculate the covariance matrix of the ellipsoidal clusters. In order to avoid the above drawbacks, a simple and systematic fuzzy classifier design method based on K-means algorithm is proposed to efficiently construct a fuzzy system [1] to correctly classify the training data.

Therefore, the rest of this paper is organized as follows: Section 2 will describe the K-means algorithm. Section 3 will propose a training procedure based on K-means algorithm. Section 4 will introduce two fuzzy rules to classify the data. In Section 5, some experiments on two data sets will be used to explain how to apply those fuzzy rules and training clusters to classify data. In Section 6, a summary of K-means based fuzzy classifier as well as future works is described.

2. K-MEANS ALGORITHM

The goal of clustering algorithms is to cluster the given data set into several groups so that the data within a group are more similar than those outside the group. Since most similar measures are sensitive to the ranges of elements in the input vectors, each of the input variables must be normalized to be within the unit interval. Therefore, we first normalize the given data set to be within the unit hypercube. K-means algorithm is one of the widely used clustering algorithms to find the c mean vectors. The K-means algorithm is described in the following.

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a given set of n vectors in a p-dimensional feature space. The K-means algorithm partitions the given data set into c clusters and calculates cluster centers $V = \{m_1, m_2, \ldots, m_n\}$. It is typical to let c samples randomly chosen from the data set serve as initial cluster centers. The procedures of the K-means algorithm are described as follows [2]:

Algorithm: (K-Means Clustering)

1 begin initialize n, c, $m_1, m_2, \ldots, m_c$
2 do classify n samples according to nearest m_i
3 recompute m_i
4 until no change in m_i
5 return $m_1, m_2, \ldots, m_c$
6 end

The computational complexity of the algorithm is O(npcT) where n is number of samples, p is the number of features and T is the number of iterations. In practice, the number of iterations is generally much less than the number of samples. So K-means algorithm is a simple and systematic algorithm for data clustering.

Further, for the step 2, according to the different shapes of the training data set (e.g. spherical or ellipsoidal) we can choose different distance measure criteria (e.g., Euclidean, Mahalanobis, Hamming, etc.) [2] to partition the training data set. In this paper, we use Euclidean distance as the dissimilarity measure.

3. K-MEANS-BASED FUZZY CLASSIFIER

In this section, a procedure based on K-means clustering algorithm is proposed to partition the training data into several clusters. It is required that the training data is normalized in a unit hypercube. The proposed procedure can be described by the following steps.

Step1: For class l ($l = 1, 2, \ldots, s$), we can get data subset $U_l = \{x_j | (x_j \in classl)\}$ and $U'_l = \{x_j | (x_j \in classl)\}$ ($U'_l = U_l$ in the initial state). Initialize parameters $l = 1$, i = 1, k = 1.

Step2: For class $l = 1$, set c = 1 for K-means algorithm.

Step3: Partition U'_l by K-means into c clusters and obtain the following information:

1. Data set $G_{li} = \{x_j \mid x_j \in U'_l, u_{ij} = 1\}$;

where,

$$u_{ij} = \begin{cases} 1.if(\|x_j - m_i\| \le \|x_j - m_k\|),\ for(k \ne i) \\ 0.therwise \end{cases}$$

2. Cluster center can be given as:

$$m_{li} = \frac{1}{|G_{li}|} \sum_{x_j \in G_{li}} x_j \quad (1)$$

3. Cluster radius:

$$r_{li} = \max_{x_j \in G_{li}} \|x_j - m_{li}\| \quad (2)$$

4. Distance from i-th cluster center of l-th class to the nearest different class datum:

$$d\_diff_{li} = \min_{x_j \notin U_l} \|x_j - m_{li}\| \quad (3)$$

In a special case, when only one datum belongs to cluster G_{li}, we define the radius r_{li} of the $cluster = 0.1 \times d\_diff_{li}$.

Step 4: Check every data set belonging to class l (G_{li} (i=1, 2, …, c)):
If $r_{li} < d\_\text{diff}_{li}$, then cluster G_{li} has been successfully partitioned, record G_{li} by assigning $Record_{lk} = G_{li}$, $Record\_Center_{lk} = m_l$, $Record\_Radius_{lk} = r_{li}$, $v_{lk} = |G_{li}|$, k = k + 1, and remove G_{li} from U'_l.
Else $r_{li} \ge d\_diff_{ii}$, then fail partition; do not record this cluster.

Step 5: If U'_l is empty, go to Step 6.
Else, check the conditions described as below:
If these are any successful partition in Step 4, we assume $c = l$ then go to Step 3.
Else c = c + 1, then go to Step 3.

Step 6: When U'_l is empty, the data belonging to class l has been partitioned successfully; then if $l < s$, we change class $l = l + 1$ and initialize parameters i = 1, k = 1, c = 1, and then go to Step 3. Else means all the classes have been considered (l = s), the cluster algorithm stops.

Till now, the training data in each class have been partitioned into several circular regions given by $Record_{lk}$, which has some advantages as follows:

1) Due to the check of r_{li} and d_diff_{li} in Step 3, the generated spherical regions will not contain the training data of the different class. It guarantees the training data can be 100% classified into the correct class, provided there are no identical data in different classes.

2) Each circular partition $Record_{lk}$ is associated with a value $v_{lk} = |Record_{lk}|$, which denotes the "total" number of data belonging to the partition. We can judge the importance of the partition by ordering v_{lk}, where the larger v_{lk} means the more important partition.

4. FUZZY RULES

After the cluster algorithm, the cluster centers and radius of the clusters $Record_{lk}$ are recorded as $Record\_Center_{lk}$ and $Record\_Radius_{lk}$. Then we can use different rules or combine these rules to classify the given data into the class.

Approach 1: For each cluster, a fuzzy IF-THEN rule is used to represent it and is described by

R_{lk}: *IF x is* A_{lk} *THEN x belongs to class l*

where A_{lk} is a fuzzy set described by the following membership function:

$$A_{lk}(x) = \exp\left(-\frac{\|x - Record\_Center_{lk}\|^2}{2Record\_Radius_{lk}}\right) \quad (4)$$

where x is a p-dimensional data. The membership value of input x for A_{lk} is denoted by $g_{lk} = A_{lk}(x)$, when x is the same as $Record\_Center_{lk}$, $g_{lk} = 1$ and while x moves away from $Record\_Center_{lk}$, g_{lk} will decrease to approximate 0.

After the rule extraction, we combine these rules to construct a fuzzy classifier. As an example, when the membership value to each rule is calculated corresponding to the input x, and the maximum value $\hat{g}_{lk}$ is determined, then x is classified into the class l^*.

Approach 2: We might change the decision criterion and we can get the different recognition performance. If we only measure the distance between each cluster and the input data which is outside of all the clusters, we simply take the nearest cluster to represent it. In the following experiment, we will show this criterion works better than approach 1 on both given data sets.

$$A_{lk}(x) = \|x - Record\_Center_{lk}\| - Record\_Radius_{lk} \quad (5)$$

This equation indicates that we will represent input data x to the class represented by the nearest cluster, so the minimum value $\hat{g}_{lk}$ is determined.

5. EXPERIMENTS

In this section, we first use a data set to give an example to illustrate the proposed method and then examine the performance of the proposed method. Further we will discuss some issues to enhance the performance.

Preliminary experiments were conducted on two data sets: set I is a 10-dimensional, 11-class classification problem, given 528 training data, and 379 test data; set II is a 39-dimensional, 5-class, sequence (by 5 continuous vectors belonging to the same class) classification problem (see the system descriptions in [3] for details). We want to construct a fuzzy classifier to correctly classify these classes. First, we use the clustering algorithm proposed in Section 3 to partition the training data. where the training data were partitioned into 97 clusters. Using those two fuzzy rules, the training data can be guaranteed 100% classified by the fuzzy classifier.

The experiments shown in Table 1 clearly show the performance of *K-means-based Fuzzy Classifier* on both data sets. Comparing the results of the two fuzzy rules, we found rule 2 yields better performance than rule 1 in both data sets.

Rule	Set I	Set II
Fuzzy I	62.01%	31.43%
Fuzzy II	53.56%	28.57%

Table 1: Comparison of performance in token error rate between Fuzzy rule 1 and rule 2.

As this is a simple algorithm, we can easily to tune some parameters or combine some other criterion to achieve better performance without significantly degrading the efficiency of the system. First, in the clustering algorithm, according to Wong et al. we will record the radius of successful partition as the

distance nearest from the different class datum. We think that only recording the nearest distance might leave many unclustering spaces in the whole data region, so it might be better to loose this limit, i.e. we propose to enlarge the radius up to the median between the original radius and the nearest hetero-class datum point. By doing this we found the performances achieve great improvements on both rules as shown in Table [2].

	Set I (%)		Set II (%)	
	Fuzzy 1	**Fuzzy 2**	**Fuzzy 1**	**Fuzzy 2**
Baseline	62.01	53.56	31.43	25.71
BR	52.24	49.34	32.86	21.43
Outlier	55.67	51.45	28.57	20.00
BR+OTL	51.98	**48.81**	25.71	**20.00**

Table 2: Comparison of enhancement experiments, 'BR' denotes the experiments enlarging the radius of each cluster; 'Outlier' denotes the experiments removing the outliers and retraining; BR+OTL denotes the experiments doing both above.

Second, as we know, in the real world, there are some outliers in the training data, which make the boundary more blurry and decision hard. By removing these data, we can get more accurate decision boundary and achieve better performance. For example in Section 3, we have described a parameter v_{ik}, which represents the cluster size. In general, the cluster that contains more data is more important. Thus, we might simply the classifier architecture by removing the clusters that have smaller v_{ik}. In our experiments, we found there are some clusters that include only single one point which has very short radius, i.e. we might think it has quite few probabilities that test data will fall into this region. Therefore, we might have two solutions to this problem: (1) merge these points to other clusters if possible; (2) remove these points as outliers. After doing this, we need to retrain the clusters, which make run-time n-iteration times. However, first several iterations might help improve the performance, and more iterations will not improve the performance anymore or even hurt the system. As shown in Table 2, we achieve a better performance by removing outliers.

Finally, Combining these two enhancement methods results in our best performance on Fuzzy 2 system.

6. CONCLUSION

In this paper, a K-means based fuzzy classifier is proposed, which has several advantages such as simple and systematic structure. In the cluster generation process, we have also proposed a clustering procedure based on K-means and two fuzzy rules to classify the test data. Our future work will focus on the following four folds: 1) Fuzzy rule, we need to find more efficient rule to represent the given data using this procedure. 2) Performance improvement. We propose two methods to improve the error rate in this paper, but we don't mention too much fuzzy rule tuning. Therefore, our aim in the future will adjust the rules to improve the classification performance. 3) Computation optimization, as the computational complexity of the algorithm increases exponentially with the data size and the number of data dimension. 4) Iterative optimization, different starting points can lead to different solutions, and one never knows whether or not the best solution has been found. The basic idea is to find some reasonable initial partition and to "move" samples from one group to another if such a move will improve the value of the criterion function, which guarantee local but not global optimization.

REFERENCES

[1] C. Wong, C. Chen, and S. Yeh, "K-Means-Based Fuzzy Classifier Design," *The Ninth IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 48-52, 2000.

[2] R. Duda, R. Hart, and D. Stork, Pattern Classification, Wiley-Interscience Publication, New York, NewYork, 2000.

[3] J. Picone, "Common Evaluation," *http://www.isip.msstate.edu/publications/courses/ece_8990_pr/exams/2001/*, Institute for Signal and Information Processing, Mississippi State University, Mississippi State, Mississippi, USA, May 2001.