Neurocomputing 188 (2016) 233–238

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Kernel-based fuzzy c-means clustering algorithm based on genetic algorithm

Yi Ding, Xian Fu*

Department of Computer Science and Technology, Hubei Normal University, Huangshi 435002, China

ARTICLE INFO

Article history:
Received 23 October 2014
Received in revised form
31 December 2014
Accepted 4 January 2015
Available online 2 December 2015

Keywords:
Fuzzy clustering
Fuzzy c-means clustering
Kernel-based fuzzy c-means
Genetic algorithm

ABSTRACT

Fuzzy c-means clustering algorithm (FCM) is a method that is frequently used in pattern recognition. It has the advantage of giving good modeling results in many cases, although, it is not capable of specifying the number of clusters by itself. Aimed at the problems existed in the FCM clustering algorithm, a kernel-based fuzzy c-means (KFCM) is clustering algorithm is proposed to optimize fuzzy c-means clustering, based on the Genetic Algorithm (GA) optimization which is combined of the improved genetic algorithm and the kernel technique (GAKFCM). In this algorithm, the improved adaptive genetic algorithm is used to optimize the initial clustering center firstly, and then the KFCM algorithm is availed to guide the categorization, so as to improve the clustering performance of the FCM algorithm. In the paper, Matlab is used to realize the simulation, and the performance of FCM algorithm, KFCM algorithm and GAKFCM algorithm is testified by test datasets. The results proved that the GAKFCM algorithm proposed overcomes FCM's defects efficiently and improves the clustering performance greatly.

1. Introduction

Traditional pattern recognition involves two tasks: unsupervised clustering and supervised classification [1,2]. In unsupervised clustering, samples without class labels are grouped into meaningful clusters. These clusters can be utilized to describe the underlying structure in data, which is helpful for better understanding of data. In supervised classification, samples with class labels are used to build the classification mechanism, through which class labels can be provided for new samples.

When class information is available, most traditional classifiers are designed in a direct way by employing supervised information to determine their decision functions. Such classifiers usually provide only class labels for new samples, but rarely care about the revelation of data distribution. For example, multilayered perception (MLP) [3] and support vector machines (SVM) [4,5] successfully utilize the class information of samples to achieve high classification accuracies; however, they emphasize more the classification of the data than the revelation of the data distribution, thus fail to interpret the obtained classification results well.

In contrast to these classifiers, another type of classifiers is designed in an indirect way by incorporating structural information into their classification schemes. Since clustering analysis is appropriate for exploring the data distribution [1,2], these classifiers usually first perform clustering to uncover the underlying structure in data, and then design classification rules based on the obtained structural information. In this way, these classifiers fuse the advantages of both clustering learning and classification learning together to some extent. On the other hand, clustering methods can be roughly categorized into unsupervised ones and supervised ones, depending on whether using class labels or not.

1.1. Clustering algorithm background

Radial basis function neural network (RBFNN) [6] is a classical algorithm belonging to the first category, i.e., unsupervised-clustering plus classifier-design. To determine the parameters of the hidden layer in RBFNN, training samples are clustered in an unsupervised way by using c-means or FCM [7]. Then, the connection weights between the hidden and output layers are optimized by minimizing the mean squared error (MSE) criterion between the target and actual outputs. Here, clustering makes RBFNN yield good generalization [3], but its function is just to help determine the parameters of the neural network, rather than explore the underlying structure of the input space. In fact, RBFNN cannot really inherit the merits of both clustering learning and classification learning as shown below.

Recently, some fuzzy relation based methods are proposed to bridge clustering and classification [8,9], which also belong to the first category. Setnes et al. proposed relational classifier trained by fuzzy clustering (FRC) to represent a transparent alternative to

* Corresponding author.
E-mail addresses: teacher.dingyi@live.com (Y. Ding), teacher.fu@live.com (X. Fu).

http://dx.doi.org/10.1016/j.neucom.2015.01.106
0925-2312/

conventional black-box techniques such as neural networks. To enhance FRC's robustness by replacing FCM and hard class labels with KFCM [10,11] and soft labels, respectively. The training of both algorithms includes two steps. First, unsupervised clustering is performed on training samples to discover the natural structure in data. Then, a relation matrix R between the obtained clusters and given class labels is established to reflect the logical relationship between clusters and classes. Here this matrix R plays a role of a connection weight matrix as in RBFNN. However, such relationship in both FRC and RFRC is directly constructed by the logical composite operator rather than by optimizing some defined criterion function. As a result, the clustering and classification results cannot be simultaneously optimal. In addition, the entries in the relation matrix R lack the statistical characteristic, and thus fail to indicate the relative reliability of the obtained relationship. Moreover, it is difficult to optimize these entries by defining an objective function due to in the differentiability of the composite operators.

It is worth noting that all the above algorithms have a common point: sequentially optimizing the clustering and classification objective functions respectively. That is, the clustering learning obtains a description of the underlying data distribution, and then the classification learning uses the obtained information to train the classification rules. In these algorithms, although the clustering learning and classification learning separately optimize their own criteria, such kind of sequential learning manner cannot always guarantee simultaneous optimality for both clustering and classification learning. In fact, the clustering learning here just aids the classification learning and does not benefit from the classification learning.

1.2. Genetic algorithm background

Over the last decade, GA have been extensively used as search and optimization tools in various problem domains, including sciences, commerce, and engineering. The primary reasons for their success are their broad applicability, ease of use, and global perspective [12].

The concept of a genetic algorithm was first conceived by John Holland of the University of Michigan, Ann Arbor. Thereafter, he and his students have contributed much to the development of the field. Most of the initial research works can be found in several conference proceedings. However, now there exist several text books on GAs [12–16]. A more comprehensive description of GAs along with other evolutionary algorithms can be found in the recently compiled “Handbook on Evolutionary Computation” published by Oxford University Press [17]. Two journals entitled “Evolutionary Computation” published by MIT Press and IEEE are now dedicated to promote the research in the field. Besides, most GAs applications can also be found in domain-specific journals.

Meng et al. [18] studied the encoding techniques of GA since GA encoding has significant influence on GA systems performance when solving problems with high complexity. A sufficient convergence condition on genetic encoding in Genetic Algorithms has been presented, such as Bias Code, Uniform Code, Trisector Code and Symmetric Code. Angelov [19] proposed a new approach for on-line design of fuzzy controllers of Takagi–Sugeno type (TS type); fuzzy rules are generated based on data collected during the process of control using newly introduced technique for on-line identification of TS type fuzzy models. Output of the plant under control and the respective control signal has been memorized and stored in on-line mode, and used to train in a noniterative, recursive way the fuzzy controller. Gacognne [20] has used the GA to find a set of nondominated solutions in the sense of Pareto instead of a unique solution with a unique fitness function. Gaccognne first began with a small random population of points in the space of research and setting a maximal size, then he used a family of genetic operators in relation with each specific problem, and he made a control on that family to give reinforcement for the best of them. Magdalena and Monasterio [21] proposed a new way to apply GAs to fuzzy logic controllers (FLC), and applies it to a FLC designed to control the Synthesis of biped walk of a simulated. A new approach adapted to systems with a larger number of variables has been proposed and tested over a FLC controlling a complex problem the locomotion of a simulated six links biped robot. Lee and Takagi [22] proposed a method for automatically designing complete triangular fuzzy systems using a genetic algorithm and a penalty strategy to determine membership function shape and position, number of fuzzy rules, and consequent parameters. Experimental results demonstrated the practicality of the method comparably to a system produced by another method, in Lee and Takagi work they have used triangular and trapezoidal membership functions for the fuzzy controller, and experimental score function. Several papers have proposed automatic design methods. Much of the work has focused on tuning membership functions [23–25] Takagi and Hayashi [26] used neural networks as a membership values generator and in [27] they treated fuzzy systems as networks and used back propagation techniques to adjust membership functions. Alata and Demirli [28] investigated the influence of the shape, the distribution of the membership functions and the order of the functional consequent of Takagi–Sugeno controller on the interpolation function of the fuzzy system. Number of inputs, conjunction operator, the order of consequent, and complementary or non-complementary triangular membership functions will determine the shape of the output.

2. Clustering algorithm

FCM algorithm is a clustering algorithm based on partitioning, which makes the idea to be divided into the biggest similarity between objects on the same cluster, while the minimum similarity between different clusters. Since the introduction of the fuzzy set theory in 1965 by Zadeh, it has been applied in a variety of fields. FCM is an improvement of common c-means algorithm for data classification that is rigid, while the FCM is a flexible fuzzy partition.

2.1. Fuzzy clustering algorithm

The fuzzy clusters are generated by the partition of training samples in accordance with the membership functions matrix $\mathbf{U}=[\mu_{ki}]$. ν_i is the degree of membership of x_k in the cluster i, x_k is the kth of d-dimensional measured data. The standard FCM uses the Euclidean distance as a cost function to be minimized and expressed as the following equation:

$$J_{FCM}(U,V)=\sum_{k=1}^{c}\sum_{i=1}^{n}\mu_{ki}^{m}|x_k-\nu_i|^2 \tag{1}$$

where $|\cdot|$ is any norm expressing the similarity between any measured data and the center.

As the FCM objective function is minimized, each pixel is assigned a high membership in a class whose center is close to the intensity of the pixel. A low membership is given when the pixel intensity is far from the class centroid. The FCM is minimized when the first derivatives of Eq. (1) with respect to μ_{ki} and ν_i are zero. The final classes and their centers are computed iteratively through these two necessary conditions. In the end, a hard classification is reached by assigning each pixel solely to the class with the highest membership value.