1995

A Neuro-Fuzzy Approach to Agglomerative Clustering

Anupam Joshi

Narendran Ramakrishnan

John R. Rice
Purdue University, jrr@cs.purdue.edu

Elias N. Houstis
Purdue University, enh@cs.purdue.edu

Report Number:
95-066

A NEURO-FUZZY APPROACH TO AGGLOMERATIVE CLUSTERING

Anupam Joshi
Narendran Ramakrishnan
John R. Rice
Elias N. Houstis

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD TR-95-066
October 1995

A Neuro–Fuzzy Approach to Agglomerative Clustering [*]

Anupam Joshi, Narendran Ramakrishnan,
John R. Rice and Elias N. Houstis
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907-1398, USA.
Phone: (317)-494-7821, Fax: (317)-494-0739
Email: {joshi,naren,jrr,enh}@cs.purdue.edu

Abstract

In this paper, we introduce a new agglomerative clustering algorithm in which each pattern cluster is represented by a collection of fuzzy hyperboxes. Initially, a number of such hyperboxes are calculated to represent the pattern samples. Then, the algorithm applies multi-resolution techniques to progressively "combine" these hyperboxes in a hierarchial manner. Such an agglomerative scheme has been found to yield encouraging results in real-world clustering problems.

1 Introduction

Clustering is a fundamental procedure in pattern recognition. It can be regarded as a form of unsupervised inductive learning that looks for regularities in training exemplars. It finds applications in unsupervised concept learning, human sensory processing, discovery, and provides a convenient representation of the input pattern domain. Clustering thus reveals associations between individuals of a population. The clustering problem [2, 5] can be formally described as follows:

Input A set of patterns $X = \{x_1, x_2, x_3, \ldots, x_n\}$

Output A c−partition of X that exhibits categorically homogenous subsets, where $2 \le c \le n$

Different clustering methods have been proposed that represent clusters in different ways - for example, using a representative exemplar of a cluster, a probability distribution over a space of attribute values, necessary and sufficient conditions for cluster membership etc [2]. To represent a cluster by a collection of training exemplars and to "assign" new samples to existing clusters, we use some form of a utility measure. This is normally based on some mathematical property such as distance, angle, curvature, symmetry, intensity that are exhibited by the members of X. It has been recognised [11] that *no* universal clustering criterion can exist and that selection of any such criterion is subjective and depends on the domain of application under question.

Clustering techniques can be divided into three categories [2, 11, 3]: (i) Hierarchial Methods, (ii) Graph–Theoretic Methods and (iii) Objective Function Methods. The hiearchial methods are based on agglomerative and divisive ideas to partition the pattern space and are characterized by their computational simplicity. The graph–theoretic methods represent the pattern exemplars by nodes on a weighted graph and edge weights are based on similarity measures between pairs of nodes. The most common of clustering methods are, however, the objective function clustering methods that attempt to minimize some function based on a distance measure to obtain the desired partition. The choice of a clustering criterion is, thus inherently

[*]This work was supported in part by NSF awards ASC 9404859 and CCR 9202536, AFOSR award F49620-92-J-0069 and ARPA ARO award DAAH04-94-G-0010

problematical and each algorithm provides a different partition of the pattern space. Also, few of these techniques take into account the uncertainty inherent in the pattern samples in the domain *i.e.,* the clusters formed are hard and don't address the problem of imprecisely defined categories.

In this paper, we describe a fuzzy agglomerative approach to the clustering problem. Section 2 details the use of fuzzy techniques to aid clustering. Section 3 outlines our algorithm in detail. We address the performance of this algorithm in Section 4 by applying it to real world problem sets. Section 5 concludes by summarizing the developments.

2 Fuzzy Clustering

Fuzzy clustering techniques have been shown to have considerable advantages over conventional methods. They serve as a natural basis for pattern recognition in cases where an object belongs to a cluster not in a binary fashion, but to a certain "degree". The first use of fuzzy sets as a basis for clustering was in [1]. A systematic and extensive analysis of the use of fuzzy sets in clustering was made by Ruspini [11]. Further clustering techniques are dealt with in detail in [2]. Each technique differs in the kind of method it uses to partition the pattern space, the type of partitions generated and the shapes of clusters that it can detect. But many of these techniques are also similar in that they iteratively minimize an objective function to obtain the final clustering. The Fuzzy C-Shells (FCS) algorithm [3] uses n-dimensional hyperspherical shells to represent cluster prototypes. A variant of this is the Fuzzy C Spherical Shells Algorithm (FCSS) [9] that first identifies good clusters, merges compatible clusters and eliminates spurious ones to obtain the final partition. The Adaptive Fuzzy C-Shells algorithm (AFCS) [4] extends the FCS algorithm to detect clusters that can be hyperellipsoidal shells. All these techniques assume that the underlying cluster shapes are hyperellipsoidal shells.

A more general technique for pattern clustering is Simpson's fuzzy min-max neural network [12] that uses groups of fuzzy hyperboxes to represent pattern clusters. We briefly describe this algorithm next. A hyperbox is used to represent a region in pattern space and all patterns completely contained within the hyperbox have full cluster membership in it. Clustering in this approach occurs by placing and adjusting these hyperboxes. For each new pattern, it looks for a "close enough" hyperbox and if it finds one, adjusts the hyperbox to contain the given pattern. If there exists no hyperbox "close enough", it creates a new hyperbox for the pattern. Further, a limit θ is imposed on the size to which a hyperbox can expand. If the size of the hyperbox extends beyond θ, a new hyperbox is created to accomodate the given pattern. This algorithm has been shown to perform very well on classical pattern recognition problems.

Each hyperbox fuzzy set B_j is described as the 3-tuple

$$B_j = \{V_j, W_j, c\} \tag{1}$$

where V_j denotes the min-point in n-space, W_j represents the max point in n-space and c denotes the class to which the fuzzy set B_j corresponds. The membership function m_j for the j^{th} hyperbox satisfies

$$0 \leq m_j(x_k) \leq 1$$

and measures the degree to which the k^{th} pattern, x_k, falls inside the j^{th} hyperbox. Simpson choses this function to be

$$m_j(x_k) = 1 - \frac{1}{n}\sum_{i=1}^{n}[f(x_{ki} - w_{ji}) + f(v_{ji} - x_{ki})] \tag{2a}$$

where

$$f(x) = max(0, min(\gamma x, 1)) \tag{2b}$$

$f(x)$ is a squashing function and the sensitivity parameter γ represents the rate at which the membership function varies as the distance between A_k and B_j varies. Thus hyperboxes, defined by pairs of min-max points, and their membership functions are used to define fuzzy subsets of the n-dimensional pattern space. The bulk of this processing involves the finding and fine-tuning of the boundaries of the clusters.

Simpson's clustering algorithm, however, results in a large number of hyperboxes (clusters) to represent the given data adequately. Also, the clustering performance depends to a large extent on the maximum allowed size of a hyperbox. In other words, θ, the maximum hyperbox size influences the number of clusters formed, and in turn, the clustering accuracy. Simpson also desires the clusters to be "compact" and hence, performs a compaction procedure that eliminates overlap between hyperboxes in *all* dimensions. The disadvantage of this is that the algorithm requires more than one run through the data in order to achieve "cluster stability" and hence discourages single–pass clustering.

3 Agglomerative Fuzzy Clustering

We propose a multi-resolution scheme, similar to computer vision [7], to partition the data into clusters. For clustering at the base of the multi-level pyramid, we use Simpson's algorithm. Then, we operate at different "zoom"/resolution levels to obtain the final clusters. This approach has led to encouraging results from clustering real world data sets.

The parameters into this algorithm are θ – the maximum hyperbox size and Z - the zoom factor. The user specifies the zoom factor as the extent to which the algorithm should "focus" on the data in the pattern space. We also enhance the fuzzy hyperbox data structure as follows:

$$B_j = \{V_j, W_j, c, M_j, \delta\} \tag{3}$$

V_j, W_j and c are the same variables as defined in eqn. (1). M_j denotes the "center–of–mass" of the pattern samples represented by the hyperbox and δ represents the number of pattern samples represented by the hyperbox.

For example, when a hyperbox B_j is first created for pattern x_i, $V_j = W_j = x_i$ (i.e., the min and the max point both correspond to the pattern sample). Now, M_j is set to x_i as x_i is the only pattern "represented" by B_j and δ is set to 1. When B_j is expanded to represent an additional pattern sample x_{i+1}, in addition to V_j and W_j getting updated by Simpson's algorithm, we update M_j and δ as follows:

$$M_j = \frac{\delta M_j + x_{i+1}}{\delta + 1} \tag{4a}$$

$$\delta = \delta + 1 \tag{4b}$$

In other words, M_j is updated to reflect the new "center–of–mass" of the pattern samples represented by B_j.

Our proposed algorithm operates as follows:

1. Initial clusters are formed from the pattern data by placing and adjusting the hyperboxes. At this stage, the number of clusters equals the number of hyperboxes. In our implementation, we have used Simpson's fuzzy min-max neural network, but any similar technique for such clustering can be used.

2. The bounding box formed by the patterns is calculated and we partition this region based on the zoom factor. In effect, this partitions the total pattern space into several levels of windows/regions. A zoom factor of Z implies that there exist Z levels above the bottom of the pyramid. The i^th level above the base level partitions the total region into $4^{(Z-i+1)}$ sub-regions. For example, if we choose a zoom factor of 2, the first level above the base has 16 sub-regions and the next level has 4 sub-regions.

3. We then assume the highest zoom factor (*i.e.*, which causes the window regions to assume the smallest size) and then examine the hyperboxes inside each window. If they are "sufficiently close by", we relabel them so that they indicate the same pattern cluster. The criterion for such combination is a function that depends on θ, Z, the size of the bounding box and the actual distance between the hyperboxes. A good choice for such a heuristic (after empirical trial and error) was found to be $d < D/2 + \theta$, where d is the actual distance between the hyperboxes and D is the diagonal of the current bounding box. Thus D represents the effect of the zoom factor Z on the pattern clustering. d, the distance between hyperboxes is defined as the distance between the centers of masses of the two hyperboxes. In other words, if hyperboxes B_i and B_j are candidates for such "combination", then

$$d = \|M_i - M_j\|_2$$

The hyperboxes are combined if the distance condition is satisifed.

4. After we are done with all regions of a zoom factor, we zoom up and view these newly grouped hyperboxes at a higher level. The same procedure is repeated till no more hyperboxes can be relabeled.

5. In effect, we form the hyperbox clusters initially, then we zoom into the pattern space and progressively relabel them, zoom out accordingly and repeat the algorithm.

Another subtle point is deciding on the method to relabel clusters - Does hyperbox B_i take on the class of B_j or vice versa? The δ parameter of the hyperboxes aid us in this decision. If the δ of B_i is greater than that of B_j, then B_j assumes the class of B_i and vice versa.

4 Experimental Results

We have implemented this algorithm and used it on some classical pattern recognition problems like the Iris data set and the famous soybean disease database (from the machine learning repository at the University of California, Irvine [10]), as well as on some new data sets from the domain of scientific computation, which we describe below.

4.1 PYTHIA

PYTHIA is an intelligent computational assistant [8] that we have developed to automate selecting methods to solve Partial Differential Equations (PDEs). PYTHIA attempts to solve the problem of determining an optimal strategy (i.e., a solution method and its parameters) for solving a given problem within user specified resource (i.e., limits on execution time and memory usage) and accuracy requirements (i.e., level of error). While the techniques involved are general, our current implementation of PYTHIA operates in conjuction with systems which solve (elliptic) partial differential equations (//ELLPACK [6]). The strategy of PYTHIA is to compare a given problem to the ones it has seen before, and then use its knowledge about the performance characteristics of prior problems to estimate those of the given one. For the class of linear second order elliptic PDEs, PYTHIA's population consists of fifty–six linear, two–dimensional elliptic PDEs defined on rectangular domains. Forty–two of the problems are parameterized which leads to an actual problem space of more than two–hundred and fifty problems. Many of the PDEs were artificially created so as to exhibit various mathematical behaviors of interest; the others are taken from "real world" problems in various ways. The population has been structured by introducing measures of complexity of the operator, boundary conditions, solution and problem. The database created using this problem population contains information about the properties of the problems plus performance data for about $15,000$ computations to solve one of the PDEs. From this population, five non–exclusive classes are defined.

The clustering problem in the PYTHIA domain is further complicated by the fact that the classes are not mutually exclusive. For example, a given PDE can be both analytic and oscillatory. Ideally, to detect such clusters, we require that the fuzzy min-max neural network allow some hyperbox clusters to overlap so that each data point gets associated with only one cluster.

4.2 IRIS

The next data set comprised the Fisher iris data [5] . This is a very popular data set and there is a wide range of classification results that can be used to provide a good measure of realtive performance. The data set consists of 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2 but the latter are *not* linearly separable from each other. 25 randomly selected patterns from each class were used for training and the remaining 75 patterns were used for testing. In this case, each pattern belonged to an unique class and hence the clustering algorithm did not run into the problems as in the case of the PYTHIA domain.

4.3 Soybean Disease Database

The final data set that we used to test the algorithm was Michalski's famous Soybean Disease Database [10].

4.4 Observations

1. We obtained encouraging results for all the data sets considered in this experiment. The number of samples that were clustered with other samples (that were known to belong to the same class) was taken to be a measure of performance. While performance on the Iris and soybean databases was of the order of 90-97%, that on the PYTHIA data set was only 85%. This can be attributed to the fact that the PYTHIA classes do not confirm to a neat clustering as the PDE pattern classes are not mutually exclusive. The fuzzy min-max neural network algorithm does not allow hyperbox clusters to overlap and hence, each data point gets associated with only one cluster. This causes the accuracy to drop down.

2. It was observed that we need not zoom down to less than two levels below the main region. However, this could be an artifact of the data sets chosen. More empirical evidence is required to support this belief.

3. The clustering accuracy did not vary with the hyperbox size θ as much as in the case of Simpson's original fuzzy min-max clustering algorithm. However, a greater accuracy was observed at small values of θ.

4. We generate confusion matrices for the clustering wherein we compare the physical clusters formed with the actual clusters that exist in the pattern space. The rows of the confusion matrix represent the clusters detected by the algorithm. The columns denote the actual clusters known to exist in the data. An entry in the (i, j) th position of the table represents the degree to which cluster i faithfully represents the actual data in cluster j. Such a matrix for the PYTHIA domain is given in the table below ($\theta = 0.0125$):

	Singular	Analytic	Oscillatory	Boundary Layer	Boundary Conditions Mixed
Cluster 1	100%				
Cluster 2		94.28%	5.7%		
Cluster 3		8.8%	91.17%		
Cluster 4				84.37%	15.6%
Cluster 5				20.2%	78.37%

In the above table, the clusters 1–5 represent the clusters identified by the fuzzy min-max algorithm. (Each cluster is a group of hyperboxes at the top level suitably relabeled so that they represent the same class.) Class information was not provided until after the clusters were formed. A routine analyzes these hyperboxes and decides what physical clusters they actually represent. In other words, each cluster (hyperbox group) detected by the algorithm is analyzed as to which physical cluster is maximally represented by it. Thus, Cluster 1 has been identified to represent the Singular class of PDEs in its entirety. We realize that there exist some errors associated with the recognition of other clusters. Nevertheless, this shows that there indeed exists some kind of ordering about the PYTHIA data. It can be easily seen that the entries in the matrix sum up to 100% computed row–wise.

A similar matrix for the IRIS data set is shown below ($\theta = 0.1$):

	Class 1	Class 2	Class 3
Cluster 1	100%		
Cluster 2		96%	4%
Cluster 3		10%	90%

Thus, for the IRIS dataset, the diagonal entries in the confusion matrix were nearly 100% and other off-diagonal entries were negligible. For want of space, we only provide the main-diagonal elements for the soybean dataset:

$$(100\%, 100\%, 80\%, 96\%, 85\%, 100\%, 80\%, 100\%, 80\%, 75\%, 80\%, 75\%, 70\%, 94\%, 98\%)$$

Considering that we "combine" (and hence, refine) previously discovered clusters, our algorithm works out to be better than Simpson's original algorithm that didn't use the multi-resolution technique and counted on an iterative scheme to achieve cluster stability. Also, the other fuzzy clustering algorithms such as FCS, FCCS and AFCS cannot cater to the situation because they recognise only clusters that are hyperelliposidal and hyperspherical in shape. Our agglomerative clustering algorithm performs as well as other conventional clustering algorithms.

5. We were able to achieve single–pass clustering and the clusters formed were found to be compact.

6. Our multi–resolution formulation can exploit parallelism where available, since the different regions of a zoom level can be processed concurrently.

5 Conclusion

In this paper, we have introduced an agglomerative approach to pattern clustering. Initial clusters are formed by Simpson's fuzzy min-max hyperbox algorithm. Then, our algorithm borrows ideas from computer vision to progressively relabel existing pattern clusters. This results in the refinement of the previously discovered clusters. The performance of the algorithm was also found to not vary drastically with the variations in the values of the controlling parameters. Such an algorithm was found to be more useful for real world pattern recognition problems than traditional approaches.

References

[1] R. Bellman, R. Kalaba, and L. Zadeh. Abstraction and Pattern Classification. *J. Math. Anal. Appl.*, vol.13:pp.1–7, 1966.

[2] Bezdek, J. Pattern Recognition with Fuzzy Objective Function Algorithms. 1981.

[3] R.N. Dave. Fuzzy Shell-Clustering and Applications to Circle Detection in Digital Images. *International J. General Systems,*, vol.16:pp.343–355, 1990.

[4] R.N. Dave and K. Bhaswan. Adaptive Fuzzy C-Shells Clustering and Detection of Ellipses. *IEEE Transactions on Neural Networks*, vol.3(5):pp.643–662, 1992.

[5] R. Fisher. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics,*, vol.7(2):pp.179–188, 1936.

[6] E. N. Houstis, J. R. Rice, N. P. Chrisochoides, H. C. Karathanasis, P. N. Papachiou, M. K. Samartzis, E. A. Vavalis, Ko-Yang Wang, and S. Weerawarana. //ELLPACK: A numerical simulation programming environment for parallel MIMD machines. In J. Sopka, editor, *Proceedings of Supercomputing '90*, pages 96–107. ACM Press, 1990.

[7] Jolion, Jean-Michel. A Pyramid Framework for Early Vision. *Kluwer International Series in Engineering and Computer Science*, 1994.

[8] A. Joshi, S. Weerawarana, and E.N. Houstis. The Use of Neural Networks to Support "Intelligent" Scientific Computing. In *Proceedings Int. Conf. Neural Networks, World Congress on Computational Intelligence*, volume IV, pages 411–416, 1994. (Orlando, Florida).

[9] R. Krishnapuram, O. Nasraoui, and H. Frigui. The Fuzzy C Spherical Shells Algorithm: A New Approach. *IEEE Transactions on Neural Networks*, vol.3(5):pp.663–671, 1992.

[10] P.M. Murphy and D.W. Aha. University of California, Irvine, Repository of machine learning databases [http://www.ics.uci.edu/ mlearn/MLRepository.html]. 1994. (Irvine, CA).

[11] E. Ruspini. A New Approach to Clustering. *Inf. Control*, vol.15:pp.22–32, 1969.

[12] P.K. Simpson. Fuzzy Min-Max Neural Networks–Part 2: Clustering. *IEEE Transactions on Fuzzy Systems,*, vol.1(1):pp.32–45, 1993.