PERGAMON

Pattern Recognition 36 (2003) 3015–3018

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Rapid and brief communication

An adaptive rough fuzzy single pass algorithm for clustering large data sets

S. Asharaf[a], M. Narasimha Murty[b,*]

[a] *Systems Science and Automation Indian Institute of Science, Bangalore 560012, India*
[b] *Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India*

Received 2 December 2002; accepted 26 December 2002

1. Introduction

Cluster analysis has been widely applied in many areas such as data mining, geographical data processing, medicine, classification of statistical findings in social studies and so on. Most of these domains deal with massive collections of data. Hence the methods to handle them must be efficient both in terms of the number of data set scans and memory usage.

Several algorithms have been proposed in the literature for clustering large data sets viz; CLARANS [1], DB-SCAN [1], CURE [1], K-Means [2], etc. Most of these require more than one pass through the data set to find the required abstraction. Hence they are computationally expensive for the clustering of large data sets. Even though we have a single pass clustering algorithm called BIRCH [1], it uses a memory expensive data structure called CF tree. In this scenario the Leader algorithm [3], which requires only a single data set scan and less memory, turns out to be a potential candidate.

This paper introduces an efficient variant of leader algorithm called Adaptive Rough Fuzzy Leader (ARFL) algorithm which out-performs the conventional leader algorithm. It employs a combination of Rough Set theory [4] and Fuzzy set theory [5] to capture the intrinsic uncertainty involved in cluster analysis. The paper is organized as follows. Section 2 discuss the conventional Leader algorithm, Section 3 introduces the proposed algorithm, in Section 4 a comparative study of the algorithm with conventional leader algorithm and single pass K-means algorithm is given and Section 5 deals with conclusions.

* Corresponding author. Tel.: +91-80-394-2779; fax: +91-80-360-2911.

E-mail addresses: asharaf@csa.iisc.ernet.in (S. Asharaf), mnm@csa.iisc.crnet.in (M.N. Murty).

2. Conventional leader algorithm for clustering

Leader clustering algorithm makes only a single pass through the data set and finds a set of leaders as the cluster representatives. It uses a user specified threshold and one of the patterns as the starting leader. At any step, the algorithm assigns the current pattern to the most similar cluster (leader) or the pattern itself may get added as a leader if its similarity with the current set of leaders does not qualify it to get added to any of the clusters based on a user specified threshold. The found set of leaders acts as the prototype set representing the clusters and is used for classifying test data.

3. Adaptive rough fuzzy leader (ARFL) clustering

The ARFL clustering scheme that we propose in this paper divides the data set into a set of overlapping clusters. To define the clusters it employs the Rough set theory and here each cluster is represented by a leader, a Lower_Bound and an Upper_Bound. The Lower_Bound of a cluster contains all the patterns that definitely belong to the cluster. There can be overlap in the Upper_Bounds of two or more clusters and the algorithm is adaptive in the sense that based on the nature of the overlap, the Upper_Bound may get adapted.

This is a two phase algorithm employing a single pass through the data set. In the first phase, the algorithm performs a pass through the data set and finds an abstraction of the clusters as some Leaders and Supporting Leaders. The *Supporting Leaders* are patterns with an intrinsic ambiguity in their assignment to some leaders and they themselves may provide a better level of abstraction in defining the clusters, if they get added as leaders.

The first phase starts with any of the patterns as the starting leader. At any step in this phase, the algorithm uses two user specified parameters called Lower_Threshold (L_T)

doi:10.1016/S0031-3203(03)00081-5

and Upper_Threshold (U_T) along with the fuzzy membership values of the pattern among the various leaders available to determine whether a pattern should get added to the Lower_Bound of some leader or Upper_ Bound of one/more leaders or the pattern itself should get added as a leader. The degree and nature of overlap in the Upper_Bound of different leaders on a candidate pattern and a user specified parameter called Overlap_Threshold (O_T) is used to determine (a) whether the addition of the current pattern (if it happens) is as a leader or supporting leader and (b) whether adaptation is needed in the Upper_Bound region of one/more clusters.

The fuzzy membership of a candidate pattern CP_i in a cluster represented by Leader L_k is found as

$$U_{ik} = \left(\sum (D(CP_i, L_k)/D(CP_i, L_j))^{2/(m-1)}\right)^{-1}, \quad (1)$$

where $D(\ )$ is some measure of dissimilarity and m is a user specified fuzzy weighting factor.

At any step r of the algorithm, let N_l be the number of currently available leaders. Depending on the value of U_{ik} and the user specified parameters, one of the three cases can arise for the assignment of the current pattern CP_i.

(1) It gets added to the Lower_Bound of any cluster.

The current pattern CP_i gets added to the Lower_Bound of the cluster represented by L_c if $\mathrm{MAX}\{U_{ik}/k{=}1 \ldots N_l\} {=} U_{ic}$ and $D(CP_i, L_c) < \mathrm{L\_T}$.

(2) It gets added to the the upper bound of one/more cluster/clusters

CP_i falls in to the Upper_Bound of all the clusters L_r for which $D(CP_i, L_r) < \mathrm{U\_T}$. If the number of clusters that are overlapping in CP_i is more than O_T, the Upper_Bound of each overlapping cluster L_o is adapted by modifying its U_T value as

$$\mathrm{mul} = \left(1 - (D(CP_i, L_o) \Big/ \sum D(CP_i, L_r))\right)$$

$$\mathrm{U\_T}(L_o) = \mathrm{MAX}\{\mathrm{mul}^* \mathrm{U\_T}(L_o), \mathrm{L\_T}\} \quad (2)$$

where r takes value from 1 to N_o, which is the number of overlapping clusters.

In this case CP_i gets added as a leader if it does not fall in the class of the majority of the overlapping clusters. It gets added as a supporting leader if all the available classes have equal membership among the overlapping leaders. Another case that can arise is when overlap on CP_i does not cross O_T. In this case it will get added as a supporting leader and no adaptation takes place in the Upper_Bound of the overlapping clusters.

(3) Gets added as Leader since it is outside the region defined by any of the existing clusters.

The second phase of the algorithm adds some of the Supporting Leaders as leaders to the existing set of leaders if it improves the quality of the prototype set.

In this phase the algorithm tries to classify the supporting leaders using the available leaders. If they are incorrectly classified they get added as leaders since they themselves contribute to the quality of the prototype set due to this addition.

The Algorithm

Data Structure used

Uses two sets

{Leader}:- keeps the set of all leaders.

{Supporting_Leader}:- to maintain the supporting leaders.

Algorithm

1. Initialize {Leader} with any one pattern from the Data Set to be Clustered
2. For all the patterns in the Data Set Do

{

Find the fuzzy membership value of the current pattern CP_i for the set of available Leaders as per Eq. (1).

Find the Leader L_j with the maximum Fuzzy membership value among {Leader}

If $D(CP_i, L_j) < \mathrm{L\_T}$

add CP_i to the lower_bound of the cluster represented by L_j

Else

{

For all the Leaders L_k from {Leader} such that $D(CP_i, L_k) < \mathrm{U\_T}$ Do

{

// adding current pattern CP_i to the Upper_Bound of one or more Leaders

Overlap = Overlap + 1;

Record the Class label of L_k

}

If (Overlap > O_T)

{

For all all the over lapping Leaders L_o do

Adjust the $\mathrm{U\_T}(L_o)$ as per Eq. (2)

Find the class Max_Class in which the maximum number of Overlapping Leaders L_o are

falling in

If Max_Class ⟨ ⟩ *Class label* (CP_i) add current pattern CP_i to {Leader}

Else

If all Classes have equal membership values among the set of overlapping Leaders $\{L_o\}$

add CP_i to {Supporting_Leader}

}

Else add CP_i to {Supporting_Leader}

}

If the current pattern CP_i *does not belong to either the Lower_Bound or the Upper_Bound* of any of the Leaders from {Leader}

add pattern CP_i to {Leader}

}

Table 1
Shows the performance of SPKM, L, FL, RL and ARFL algorithms

SPKM		LEADER		FL		RL		ARFL	
NP	CA	NP	CA	NP	CA	NP	CA	NP	CA
1123	27.90	1110	79.99	1138	81.28	1112	82.78	1083	85.81
1307	28.56	1319	81.97	1348	83.68	1317	83.56	1307	86.11
1470	31.50	1437	83.68	1414	84.39	1484	84.46	1470	87.13
1588	31.08	1544	84.25	1588	85.30	1555	85.18	1588	88.66
2201	38.70	1983	87.13	1877	87.25	1965	87.94	2077	89.35

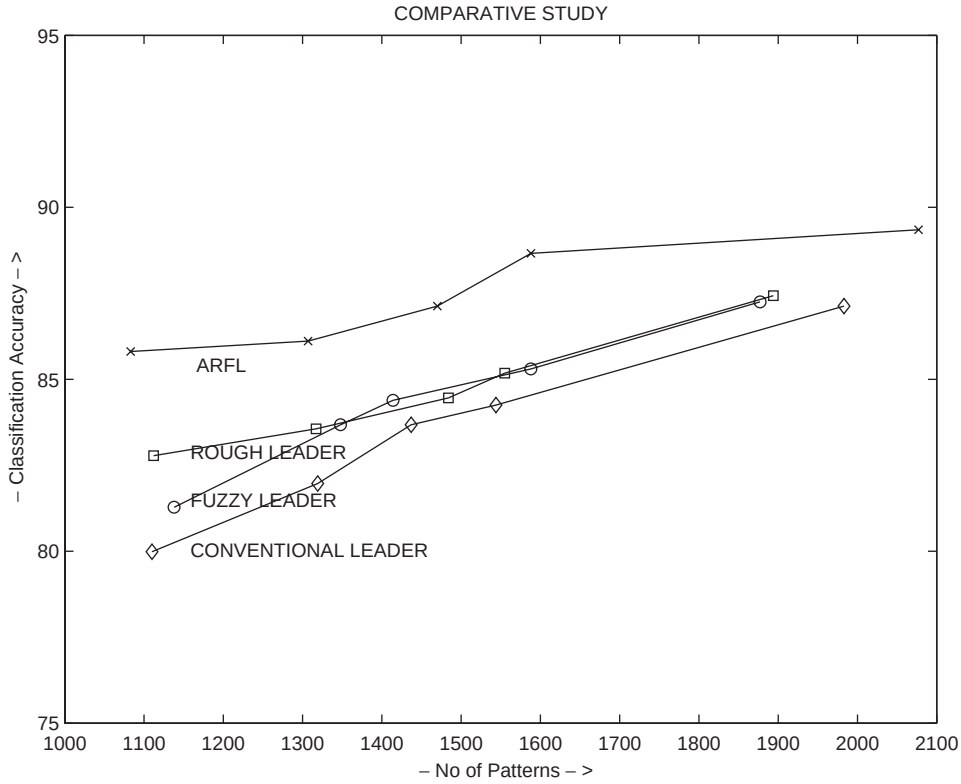

Fig. 1. Shows a comparison of SD, S_Dbw and hybrid validity indices when used for the evaluation of ARFL algorithm.

3. For all the Supporting_Leader SL_j from {support_Leader} do
   Classify SL_j using {Leader}
   If SL_j is incorrectly classified add SL_j to {Leader}
4. Return the set {Leader} as the set of prototypes representing the clusters

4. Experimental results

Single Pass K-Means(SPKM), Conventional Leader (L) algorithm, Fuzzy Leader (FL) algorithm, Rough Leader (RL) algorithm and ARFL clustering algorithm are implemented and used for generating the prototypes.

4.1. Prototype selection

Let $X = \{X_i/i = 1 \ldots n\}$ be the data set. Let the cluster abstraction generated by a clustering algorithm be $C = \{C_i/i = 1 \ldots m\}$. Let the corresponding cluster descriptions be given by the prototypes $R = \{R_i/i = 1 \ldots m\}$. In the case of the conventional Leader algorithm and the variants of Leader algorithm, the obtained leaders form the prototypes. For the Single Pass K-Means algorithm the centroid of each cluster C_i forms the prototype representing that cluster.

4.2. Data set used

The data set used is handwritten digit data. The training data consist of 667 patterns for each digit from 0 to 9, to-

tally 6670 patterns. The test data consists of 3333 patterns [3]. Each of this patterns is of 193 dimensions and the last dimension shows the class label.

4.3. Comparative studies

Table 1 and Fig. 1 show the Number of Prototypes (NP) generated and the corresponding Classification Accuracy (CA) obtained using Nearest Neighbour Classifier, for the different algorithms whose performance is compared.

Among the different algorithms we have considered, it can be seen from the table and figure that, the ARFL algorithm gives the best performance in terms of both memory usage and classification accuracy. The proposed ARFL algorithm is pragmatic in the sense that it selects and maintains only the most appropriate patterns as the representatives of the clusters.

5. Conclusion

A novel variant of the conventional Leader algorithm for clustering of large data set is proposed. The advantages of the proposed scheme are: (a) It generates cluster abstraction in a single data set scan, (b) the clusters generated by this new algorithm can be of arbitrary shape, (c) the quality of the cluster abstraction generated by ARFL outweighs the other single pass clustering schemes, (d) this is a scalable algorithm and (e) The memory requirement of the algorithm is limited to the space required for two simple set implementations.

References

[1] A.K. Pujari, Data Mining Techniques, Universities Press, Nancy, 2001.
[2] F. Farnstrom, J. Lewis, C. Elkan, Scalability for clustering algorithms revisited, SIGKDD Explor. 2 (1) (2000) 51–57.
[3] T. R. Babu, M.N. Murty, Comparison of Genetic algorithm based prototype selection scheme, Pattern Recognition 34 (2) (2001) 523–525.
[4] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (1982) 341–356.
[5] L. Zadeh, Fuzzy sets, Inf. Control 8 (1965) 338–353.