A Rough Fuzzy Approach to Web Usage Categorization

Asharaf. S
Systems Science and Automation
Indian Institute of Science
Bangalore 560012
asharaf@csa.iisc.ernet.in

M. Narasimha Murty
Computer Science and Automation
Indian Institute of Science
Bangalore 560012
mnm@csa.iisc.ernet.in
(corresponding author)

Abstract

This paper introduces a novel clustering scheme employing a combination of Rough set theory and Fuzzy set theory to generate meaningful abstractions from web access logs. Our experimental results show that the proposed scheme is capable of capturing the semantics involved in web access logs at an acceptable computational expense.

Keywords : Roughness, Fuzziness, Leader, Supporting Leader, Upper_Bound, Lower_Bound.

1. INTRODUCTION

Web usage categorization is the non-trivial process of distinguishing implicit, previously unknown but potentially useful groups that may exist in any collection of web access logs. The required abstraction can be generated by clustering the web access logs based on some sort of similarity measure. Clustering is done such that the web access logs within the same group or cluster are more similar than data points from different clusters. Each of the generated clusters represents an access pattern of a group of people having similar behaviour.

The prime requirement of an algorithm performing clustering of web access logs is that the number of data set scans is less. Memory requirement turns out to be another bottle neck. Even though we have a good variant of k-means algorithm namely Rough k-means [3], it is a less attractive candidate due to its iterative nature. Work reported in the literature includes Web Usage Mining [5], Adaptive web sites [6], Mining Web access logs using relational competitive fuzzy clustering [7], Web personalization engine based on user transaction clustering [8] etc. , for generating useful abstractions from web access patterns. But an efficient and robust [9] Soft Computing approach for Web Usage Categorization is still not a reality.

The algorithm introduced in this paper is a Rough Fuzzy variant of the Leader [4] algorithm for clustering. It employs a cooperative combination of the soft computing approaches namely Fuzzy set theory [2] and Rough set theory [1] to cluster the web access logs. It is of incremental nature and hence scalable which is very critical in this kind of applications.

The rest of the paper is organized as follows. Section 2 introduces the rough k-means algorithm. Section 3 describes the conventional Leader algorithm. In section 4 the proposed algorithm is discussed. Section 5 deals with experimental results and section 6 with conclusions.

2.The Rough k-means Algorithm

Rough set is a mathematical tool used to deal with uncertainty. When we have insufficient knowledge to precisely define clusters as sets, we use rough sets; here, a cluster is represented by a rough set based on a lower_approximation and an upper approximation [1,3]. Some of the basic properties of rough sets are:

1)An object v can be part of at most one lower_approximation.

2)For a set X_i and object v, if $v \in$ lower_approximation(X_i), then $v \in$ upper_approximation(X_i).

3)If an object v not part of any lower_approximation, then v belongs to two or more upper_approximations.

In web usage categorization, similar web access logs are grouped using insufficient knowledge about the groupings and hence rough set theory fits suitably into this paradigm.

The Rough K-means algorithm provides a rough set theoretic flavour to the conventional K-means algorithm to deal with uncertainty involved in cluster analysis. The rough K-means algorithm can be stated as follows

1) Select an initial partition of n objects into k clusters.

2) Assign each pattern to the lower_bound ($\underline{A}(X)$) or upper_bound ($\tilde{A}(X)$) of cluster/clusters respectively as

For each object vector v, let $D(v,X_i)$ be the distance between itself and the centroid of cluster X_i. The difference $D(v,X_i) - D(v,X_j)$, $1 \le i,j \le k$ is used to determine the membership of v as follows

a) *If $D(v,X_i) - D(v,X_j) \leq threshold$, for any pair (i,j), then $v \in \tilde{A}(X_i)$ and $v \in \tilde{A}(X_j)$. Furthermore, v will not be a part of any lower_bound.*

b) *Otherwise, $v \in \underline{A}(X_i)$, such that $D(v,X_i)$ is the minimum for $1 \leq i \leq k$.*
In addition, $v \in \tilde{A}(X_i)$.

3) *For each cluster X_i recompute new cluster center according to the following equation as the weighted combination of the data points in its lower_bound and its upper_bound*

$$x_i = \begin{cases} w_{lower} \times \frac{\sum_{v \in \underline{A}(X)} \{V_j\}}{|\underline{A}(X)|} + w_{upper} \times \frac{\sum_{v \in \tilde{A}(X) - \underline{A}(X)} \{V_j\}}{|\tilde{A}(X) - \underline{A}(X)|} & \text{if } |\tilde{A}(X) - \underline{A}(X)| \neq \emptyset \\ w_{lower} \times \frac{\sum_{v \in \underline{A}(X)} \{V_j\}}{|\underline{A}(X)|} & \text{otherwise} \end{cases}$$

where $1 \leq j \leq k$. The parameters w_{lower} and w_{upper} correspond to the relative importance of lower and upper bounds.

4) *If convergence criterion is met, i.e the centroid vectors from the previous iteration are identical to those generated in the current iteration, then stop; else go to step2.*

3. The Leader Algorithm

Leader clustering algorithm makes only a single scan of the data set and finds a set of leaders as the cluster representatives. It uses a user specified threshold and the algorithm can be stated as follows

1. Start with any of the patterns as the initial leader
2. For each pattern in the data set do
a) Find the nearest leader L_j for the current pattern CP_i from the set of all currently available Leaders
b) If the distance $D(CP_i,L_j) < threshold$
assign CP_i to the Cluster represented by L_j
Else add CP_i as new a leader

The found set of leaders acts as the prototype set representing the clusters and is used for further decision making. Due to the incremental nature and since the representative patterns from the data set itself (Leaders) form the prototype set representing the clusters, the algorithm generates cluster abstractions that are not biased by the existence of outliers. Thus the Leader algorithm exhibits robust behaviour.

4. Rough Fuzzy Clustering Approach

4.1 A Primer

The proposed method employs a flexible representation scheme where different categories are viewed as 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.

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 web access patterns as a starting leader. At any step in this phase, the algorithm uses two user specified parameters called **Lower_Threshold**(**L_T**) and an **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

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

Where $D(\ )$ is some measure of dissimilarity, **m** is a user specified fuzzy weighting factor and N_l is 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 a Cluster

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

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 $D(CP_i,L_r) < U\_T$. 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 ie. $\mathbf{U\_T(L_o)}$ is adapted as

$$\mathbf{mul = (1 - (}\boldsymbol{D}\mathbf{(CP_i,L_o)/ \sum_{r=1}^{N_o} \{ }\boldsymbol{D}\mathbf{(CP_i,L_r) \}))}$$

$$\mathbf{U\_T(L_o) = }\boldsymbol{MAX}\mathbf{ \{ mul * U\_T(L_o) , L\_T \}} \quad \mathbf{\ldots\ldots\ldots\ (2)}$$

where N_o is the number of overlapping clusters and the ***MAX*** function is defined as

$\boldsymbol{MAX}\{ A,B \}$ = A if A=B
= maximum of A and B otherwise

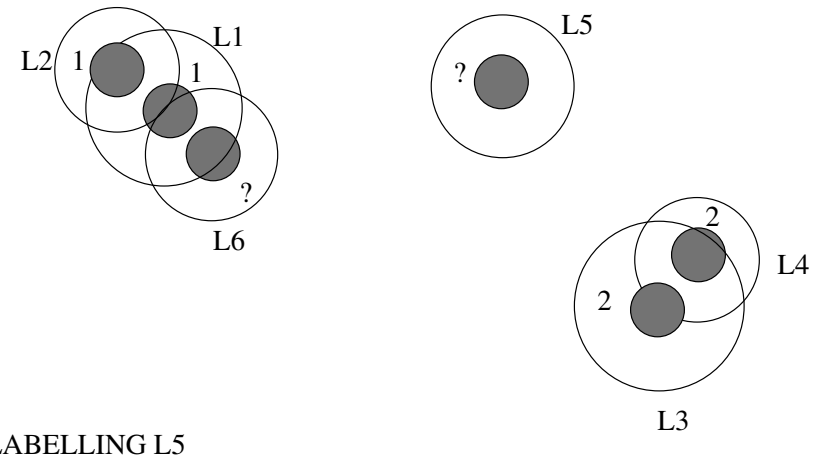
In this case CP_i gets added as a leader. When overlap on CP_i does not cross O_T the current pattern CP_i will get added as a supporting leader and no adaptation takes place in the Upper_Bound of the overlapping clusters.

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

Once the cluster abstractions are found as Leaders and Supporting Leaders, the algorithm tries to add some of the Supporting Leaders to the set of Leaders if they can contribute to the quality of the prototype set generated. The strategy adopted here is, the algorithm will add any Supporting Leader to the set of Leaders if its dissimilarity with atleast one Leader from the set of currently available set of Leaders is not within the Lower_Threshold. The output of the first phase of the algorithm is the set of Leaders which can act as the prototype set representing the data set which is being clustered.

The second phase of the algorithm generates meaningful abstractions for the given data set by clustering the leaders obtained in phase1 and then performing syntactic labelling. Here for clustering, we use the same clustering scheme that is used in phase1 with the Leaders obtained in phase 1 as the input. The strategy adopted for syntactic labelling is that well separated clusters are given distinct labels. We start labelling from an arbitrary leader obtained in phase 2 and at any step assign the farthest leader from the leaders labeled so far with a new label if it does not qualify itself to get added to a cluster represented by an already labelled leader. If the current pattern does fall within the proximity (within the Upper_Threshold) region of an already labelled pattern it is labelled with the same label. The syntactic labelling procedure is illustrated in Fig 1. The syntactic label of L1 and L2 is 1 and that of L3 and L4 is 2.

The labelled set of leaders obtained after phase 2 act as the prototype set representing the distinct categories that exist in the data set.



LABELLING L5
==========

Since D(Li,L5) > Upper_Threshold(Li) give a new label say '3' to L5
i = 1,2,3,4

LABELLING L6
==========

Since D(L1,L6) < Upper_Threshold(L1) label L6 with the label of L1

Note:

● : marks the Lower_Threshold of the cluster

○ : marks the Upper_Threshold of the cluster

FIG 1: Illustrates the Syntactic labelling procedure

4.2 The Algorithm

Data Structures used

Uses two sets

{Leader} :- the set of all leaders.

{Supporting Leader} :- to maintain the supporting leaders.

Algorithm

Procedure RFClustering (L_T,U_T,m,O_T)

{

Initialize {Leader} with any one pattern from the Data Set to be Clustered

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 equation (1).

Find the Leader L_j with the maximum Fuzzy membership value among the currently available set of leaders ie. {Leader}

If D(CP_i , L_j) < 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 **$D(CP_i,L_k) < U\_T(L_k)$** Do*

{

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

Overlap = Overlap +1

}

If (Overlap > O_T)

{

/ since there is much overlap on CP_i, add CPi to the set of leaders and adjust the Upper_Bound of all the overlapping clusters by modifying the Upper_Threshold of the corresponding leaders */*

Add CP_i to {Leader}

For all the overlapping Leaders L_o do
Adjust the $U\_T(L_o)$ as per equation (2)

}

Else

// since there is not much overlap add CPi as a Supporting_Leader.

add CP_i to {Supporting_Leader}

}

//when CP_i is outside the region defined by all the currently available set of leaders

If the current pattern 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}

}

/* add some of the Supporting Leaders to the set of Leaders if it improve the quality of the prototype set generated */

For each Supporting Leader SL_i from {Supporting Leader} do

If ***there does not exist atleast one L_j*** from {Leader}
such that $D(L_j,SL_i) < L\_T$

add SL_i to the the set of leaders **{Leader}**

Return the set {Leader} as the set of prototypes representing the clusters
}

Procedure WebUsageCategorization(Data Set)

{

1. *Cluster the given data set using the procedure RFClustering()*
2. *Generate refined abstractions of the data set by clustering the set of leaders obtained in step 1 and performing syntactic labelling*

   Perform RFClustering() with the leaders obtained in step 1 as the input

   Use the syntactic labelling scheme to generate meaningful abstractions of the given data set

3. *Return the labelled set of Leaders as the set of prototypes representing the different categories*

}

4.3 Analysis

Let n be the number of data points in the given data set and let k_{1L} and k_{1S} be the leaders and supporting leaders generated in the first phase of the algorithm also $k_1 = k_{1L} + k_{1S}$. Now the time complexity of the first phase is $O(n\, k_1) + O(k_{1L} k_{1S})$. Since $k_1 <<< n$. it is $O(n\, k_1)$.
The second phase of the algorithm starts with the set of leaders generated in phase 1 as the input. There can be a maximum of k_1 leaders. Hence the time complexity of clustering the leaders is $O(k_1 k_2)$ where k_2 is the number of clusters generated by clustering the leaders obtained from phase 1. These k_2 clusters act as the input to the syntactic labelling procedure. This can have a complexity of $O(k_2^2)$.

Hence the overall time complexity of the algorithm is $O(n\, k_1) + O(k_1 k_2) + O(k_2^2)$.
Since $k_2 <<< k_1 <<< n$, we can say that the complexity of the algorithm is $O(n\, k_1)$.

Regarding the space complexity, at any step the algorithm has to maintain at most k_{1L} leaders and k_{1S} supporting leaders each having a dimensionality of d. Hence the space complexity is $O(dk_1)$ where $k_1 = k_{1L} + k_{1S}$.

Also note that the Rough K-means is an O(nk) algorithm. But the algorithm proposed in this paper requires only one data set scan.

5. Experimental Results

The Rough Fuzzy Clustering approach to web usage categorization is implemented and applied to three distinct data sets. The dissimilarity measure used in all the experiments is the squared Euclidean distance.

5.1 Prototype Selection

Let $X = \{X_i / i = 1...n\}$ be the data set. Let the cluster abstraction generated by a clustering algorithm be $C = \{ C_i / i=1...m\}$. Let the corresponding cluster descriptions be given by the prototypes $R = \{R_i / i = 1...m\}$. In the case of the proposed approach the obtained leaders form the prototypes representing different categories.

5.2 Data Set Used

The data sets used are the web access logs of university students doing a course in Computer Science. The students are from different educational backgrounds and hence their attitude towards the course vary in a great deal. The class notes and assignments for the course was put on the web and the access pattern of the students is recorded.
Based on the attitude towards the course it can be expected that the students generally fall into three broad classes

1. Studious : These students do their work in a regular manner. Hence they always download the current set of nodes regularly.

2.Crammers: These set of students stay away from class notes and assignments for long period of time and download the bulk just before the exam for a pre-test cramming.

3.Workers: These group visit the website but are more interested in doing class and lab assignments than downloading the class notes.

The data collected for each web access consist of six features. They are

1) The first field is index.
2) Second field is Campus access, (access on campus or off campus)
3) Third field is Day/Night Time, (daytime or night)
4) Fourth field is Lab Day, (Lab time or non lab time, lab time is Tuesday and Thursday)
5) Fifth field is Hits
6) Last field is Document Requests.

The values for second, third, fourth fields are either 0 or 1.
Three such distinct data sets having a size of 1287, 6056 and 7673 are used for the empirical studies.

5.3 The User Specified Parameters

The algorithm uses four user specified parameters viz; Upper_Threshold(U_T), Lower_Threshold(L_T), m and Overlap_Threshold(O_T).

Upper_Threshold(U_T), Lower_Threshold(L_T): The L_T of the cluster define the hard core region of the cluster and U_T define the soft core region of the cluster. Their selection is highly depend on the nature of the clusters that exist in the data set. A low value for them may result in over fitting where a large cluster will get represented by

more than one subclusters(represented by corresponding leaders). But this sort of a representation is able to capture the abstraction for any arbitrarily shaped cluster. A high value for L_T and U_T may result in under fitting where more then one nearly spaced small sized clusters will get represented by by a single leader which is un acceptable.

m : This is fuzzy weighting factor. The extent of fuzzification lies in the selection of the value of m. Two extreme cases possible are

m = ∞: result in total randomness. ie. irrespective of the value of the distance the current pattern will have equal fuzzy membership in all the candidate clusters.

m = 1 : result in boolean. ie. the fuzzy membership of the current pattern in the nearest neighbour cluster will be 1 and for all the other clusters it will be zero.

Depending on the scenario appropriate fuzzy weight should be determined.

O_T : The extend of overlap that can be permitted between the clusters is determined by the value of O_T. A low value of O_T does not permit much overlap.

5.4 Results

The experiments are done for the three data sets mentioned above. It is seen that in all the three cases the proposed scheme resulted in generation of three well defined categories from the web access logs representing the three different classes of students as expected. Figures 2,3 and 4 show the different categories obtained using the Rough Fuzzy Approach to web usage categorization for the three different data sets used.

FIG 2: Shows the different categories obtained for data set of size 1287

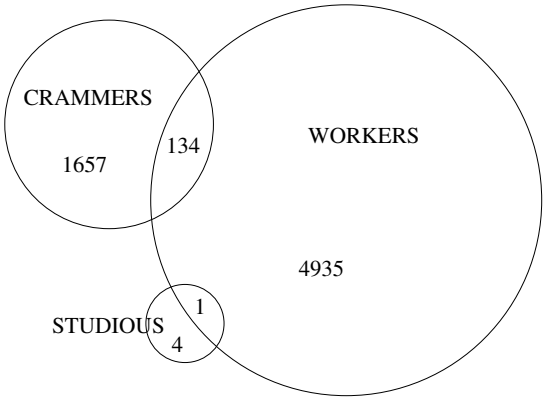


FIG 3: Shows the different categories obtained for data set of size 6056

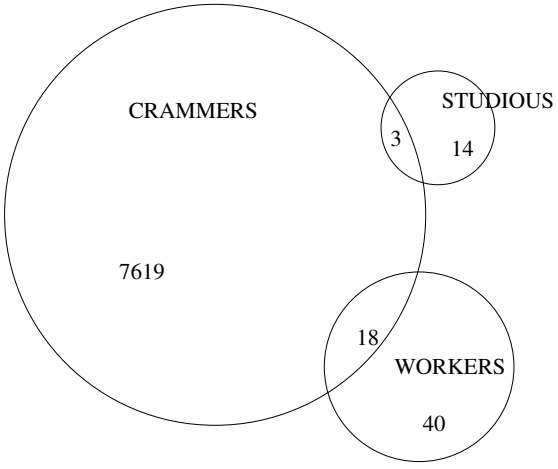


FIG 4: Shows the different categories obtained for data set of size 7673

6 Conclusion

A novel variant of the conventional Leader algorithm for categorization of web users is proposed. The advantages of the proposed scheme are

a) It generates abstractions representing different categories in a single data set scan

b) It is a scalable algorithm

c) The memory requirement of the algorithm is limited to the space required for maintaining the leaders and supporting leaders

d) Due to the incremental nature and since the representative patterns from the data set itself (Leaders) form the prototype set representing the clusters, the algorithm generates cluster abstractions that are not biased by the existence of outliers. But a centroid kind of representation for the prototypes is not robust since it can be affected by the outliers. Hence the algorithm is robust unlike the rough k-Means approach which uses rough centroids to represent the clusters.

References

[1] Pawlak . Z. Rough Sets . International Journal of Computer and Information Sciences, 11 (1982), 341-356.

[2] Zadeh .L. Fuzzy Sets, Information and Control,1965, 338-353.

[3] Pawan Lingras, Chad West. Interval set Clustering of Web users with Rough k-Means, submitted to the Journal of Intelligent Information System in May 2002, http://cs.stmarys.ca/pawan/research/pubindx.htm#submit.

[4] Spath H. Cluster Analysis Algorithms for Data Reduction and Classification of Objects, Ellis Horwood Limited, West Sussex, U.K, 1980.

[5] R. Cooley, and J. Srivastava. Web Usage mining : Discovery and applications of interesting patterns from web data, PhD thesis, Graduate School of the University of Minnesota, University of Minnesota, 2000.

[6] B. Mobasher, R. Cooley, and J. Srivastava. Creating adaptive web sites through usage-based clustering of urls. In IEEE Knowledge and Data Engineering Workshop (KDEX'99), November 1999.

[7] O. Nasraoui, H. Frigui, A. Joshi, R. Krishnapuram. Mining Web access logs using relational competitive fuzzy clustering. In Proceedings of the Eight International Fuzzy Systems Association World Congress, August 1999.

[8] B. Mobasher. A Web personalization engine based on user transaction clustering. In Proceedings of the 9th Workshop on Information Technologies and Systems (WITS'99), December 1999.

[9] R.N Dave, R Krishnapuram. Robust Clustering Methods: A Unified View IEEE Trans. On Fuzzy Systems, Vol5, No2, pp 270-293,1997.