Clustering Web Access Patterns Based on learning Automata

Babak Anari [1], Mohammad Reza Meybodi [2] and Zohreh Anari[3]

**[1] Computer Engineering Department, Islamic Azad University
Shabestar, 5381915351, Iran**

**[2] Computer Engineering and Information Technology Department, Amirkabir University of Technology
Tehran, 5381915351, Iran**

**[3] Computer Engineering Department, Islamic Azad University
Shabestar, 5381915351, Iran**

Abstract

The interest of web users can be revealed by the visited web pages and time duration on these web pages during their surfing. In this paper a new method based on Learning Automata for clustering web access patterns is proposed. At the first step of the proposed algorithm, each web access pattern from web logs is transformed into a weight vector using the learning automata. In the second step a primitive clustering is performed to group weight vectors into a number of clusters. Finally, the weighted Fuzzy c-means approach is developed to deal with the results of the second step. Our experiments on a large real data set show that the method is efficient and practical for web mining applications.

Keywords: ***Web access patterns, Clustering, Learning automata, Distributed learning automata, Time duration.***

1. Introduction

The problem of clustering web access patterns is a part of a larger work of web usage mining which is the application of data mining techniques to discover usage patterns from Web data typically collected by web servers in large logs [9]. Clustering web access patterns is an important step in studying the characteristics of web surfers. The patterns from web data are non-numerical, thus Runkler and Beadek [7] proposed relational clustering method to group non-numerical web data. Some other researchers tried to explore clustering by another soft computing technique, rough theory. Wu [10] proposed a two-layer evolutionary clustering algorithm to group web access patterns from web logs. Shi [8] applied rough k-means clustering method in fuzzy environment to group web access patterns from web logs. De [3] tries to use rough approximation to cluster web transactions.

In this paper we propose a new algorithm based on learning automata to group web access patterns from web logs. First each pattern is converted into a weight vector using learning automata. Then the learning automaton is used to group all the weight vectors into a number of clusters. And the set of centers of these clusters is further clustered by the weighted Fuzzy c-means. The rest of the paper is organized as follows: In section 2 the learning automata and the distributed learning automata will be explained. In section 3 the proposed algorithm to cluster web access patterns is proposed. In section 4 the effectiveness of the method is demonstrated. Finally we conclude in section 5.

2. Learning Aautomata

An automaton can be regarded as an abstract model that has finite number of actions. This action is applied to the selected action of automata. The random environment evaluates the applied action and gives a grade to the selected action of automata. The response from environment (i.e. grade of action) is used by automata to select its next action. By continuing this process, the automaton learns to select an action with the best grade. The learning algorithm is used by automata to determine the selection of next action from the response of environment. Figure 1 shows the relationship between the environment and the learning automata [6].

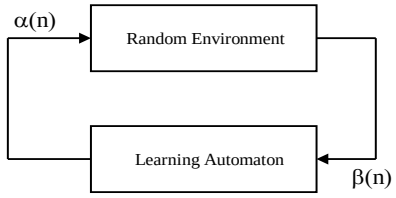

Figure1: The relationship between learning automata and the environment

2.1 Environment

The environment can be shown by E ≡{α, β, c} in which α ={α_1,α_2,….,α_r} represents a finite action / output set,

$\beta=\{\beta_1, \beta_2,\ldots, \beta_m\}$ represents an input / response set, and $c=\{c_1, c_2,\ldots,c_r\}$ is the set of penalty probabilities, where each element c_i corresponds to one action α_i of the set α. The output (action) α_n of the automaton belongs to the set α, and it is applied to the environment at time t = n.

2.2 Learning Automata with Variable Structure

Variable structure learning automata is represented by $< \beta, \alpha, T, p >$, where $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ is a set of actions. $\beta = \{0, 1\}$ is the set of inputs from the environment; where 0 represents a reward and 1 represents a penalty, $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ is learning algorithm and defines the method of updating the action probabilities on receiving an input from the random environment. $p = \{p_1(n), p_2(n)\ldots p_r(n)\}$ is the action probability vector, where $p_i(n)$ represents the probability of choosing action α_i at time n. In these kinds of automata, if the action of α_i is chosen in the n^{th} stage and receive the desirable response from the environment, the probability of $p_i(n)$ increases and the other probabilities decreases and in undesirable response, the - probability of $p_i(n)$ decreases and the other probabilities increase. The following algorithm is one of the simplest learning schemes for updating action probabilities, and is defined as follows:

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)] \qquad \forall j \quad j \neq i$$
$$p_j(n+1) = (1-a)\,p_j(n) \qquad (1)$$

a) Desirable response

$$p_i(n+1) = (1-b)\,p_i(n)$$
$$\forall j \quad j \neq i \quad p_j(n+1) = \frac{b}{r-1} + (1-b)\,p_j(n) \qquad (2)$$

b) Undesirable response

As seen from the definition, the parameter a is associated with reward response, and the parameter b with penalty response. According to the values of a and b we can consider three scheme. If the learning parameters a and b are equals, the scheme called reward penalty (L_{R-P})When b is less than a, we call it linear reward epsilon penalty ($L_{R\varepsilon P}$) scheme. When b equals to zero, we call it as linear reward inaction (L_{R-I}) scheme. For more information about the theory and applications of learning automata, refer to [2, 7, 8].

2.3 Distributed Learning Automata (DLA)

A distributed learning automaton (DLA) is a network of learning automata which collectively cooperates to solve a particular problem. In DLA, the number of actions for any automaton in the network is equal to the number of outgoing edges from that automaton. When an automaton selects one of its actions, another automaton on the other end of edge corresponding to the selected action will be activated. At any time only one automaton in the network will be active. Formally, a DLA with n learning automata can be defined by a graph (V, E).Where $V= \{LA_1, LA_2\ldots LA_n\}$ is the set of automata and $E \subset V \times V$ is the set of edges in the graph in which an edge (LA_i, LA_j) corresponds to action α_j of automata LA_i. Figure 2 shows the network of distributed learning automata. The action probability vector for automaton LA_i is shown by $p^i = \{p_1^i, p_2^i, \ldots, p_m^i\}$ where p_m^i denotes the probability of selecting action α_m , that is, edge (LA_i, LA_m). The choice of action α_m^i by LA_i activates LA_m. r_i shows the number of actions done by LA_i automata.

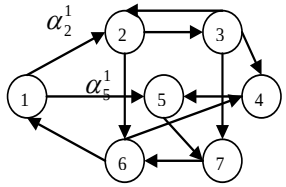

Figure 2: Network of distributed learning automata

3. Clustering Web Access Patterns based on Learning Automata

The proposed algorithm includes three steps: At the first step of the proposed algorithm, each web access pattern from web logs is transformed into a weight vector using the learning automata. In the second step we put each weight vector in the nearest cluster using the learning automata. By doing this, a primitive clustering is performed on the web access patterns and the primitive centers of clusters are determined. Finally, these primitive clusters which have no or several access patterns are used by weighted c-means clustering algorithm and on the basis of the weight of each cluster which has been determined according to the number of web access patterns in each cluster and also the centers of clusters which have been determined by the learning automata are clustered. Finally, the final clusters are determined from the primitive clusters.

3.1 Characterizing Web Access Pattern as a Weight Vector

Suppose there are m users and user transactions where $s_i (1 \le i \le m)$ representing the unique surfing behavior of the i^{th} user. Let $W = \{Url_1, Url_2, \ldots, Url_n\}$ is the union set of distinct n web pages visited by users, $U = \{(Url_1, t_{1_1}), \ldots, (Url_1, t_{1_g}), \ldots, (Url_n, t_{n_1}), \ldots, (Url_n, t_{n_h})\}$ be the union set of $s_i (1 \le i \le m)$, where g is the number of time

duration on web page Url_1, h is the number of all time durations on web page. Url_n, n is the number of all different web pages visited by users. Let T_{max} be the maximum time duration on web pages. Web pages and users play the role of stochastic environment for the existing learning automata in DLA. In the proposed method for each web page like P_k the learning automata of LA_k is considered. Suppose $p^k = \{p_1^k, p_2^k, \dots, p_n^k\}$ is the action probability vector for LA_k automata has been assigned to web page of P_k and $p_m^k(n)$ is the probability of choosing action α_m in learning automata of LA_k at the n^{th} time. If the user moves from page P_k to page P_m ($P_k \rightarrow P_m$) learning automata of LA_k updates its action probability vector based on learning algorithm. We can represent each web access pattern $s_i \in S(1 \le i \le m)$ by a weight vector as $W_i =< w_{i1}, w_{i2}, \dots, w_{ik}, \dots, w_{in} >$ where w_{ik} shows that the i^{th} user has been visited the web page of k. The main steps of the algorithm for determining the action probability of each automaton are provided as follows:

Step1. Create a DLA according to web pages structure.
Step2. Do Eq.3 for each learning automata such as LA_k where$(1 \le k \le n)$,

$$p_i^k(n) = \begin{cases} \frac{1}{r} & i \ne k \\ 0 & i = k \end{cases} \qquad (1 \le i \le n) \qquad (3)$$

Step3. Do step3-1 for every user access pattern in the log file

Step3-1 If the user moves from page D_k to page D_m $(k \ne m)$ and t_{value} is the time duration on D_m then update the action probability vector for LA_k automata based on Equations of 4, 5, 6 and 7 respectively.

$$a_m^k(n+1) = \frac{t_{value} \times p_m^k(n)}{T_{\max}} \qquad (4)$$

$$p_m^k(n+1) = p_m^k(n) + a_m^k[1 - p_m^k(n)] \qquad (5)$$

$$p_j^k(n+1) = (1 - a_m^k)\, p_j^k(n) \; j \ne m \forall j \qquad (6)$$

$$w_{ik} = \begin{cases} p_m^k & \text{If the user moves from page } D_k \text{ to page } D_m, (k \ne m) \\ 0 & \text{Otherwise,} \quad (1 \le k \le n) \end{cases} \qquad (7)$$

By increasing the number of access frequency to each web page and spending more time duration on each web page the amount of probability increases.
For example let $S = \{s_1, s_2, \dots, s_6\}$ be the set of user transactions and W= {A, B, C, D, E, F, G, H} be the union set of distinct web pages visited by all users, then 10 learning automata are considered (for each web page one automata and also one automata for the exit and start pages). Suppose that start and exit are the two pages that the user has entered and left from the site respectively. Also we consider 10 actions for each automaton according to equation 1. If the browsing sequence of a user is as $Start \rightarrow A \rightarrow B \rightarrow C \rightarrow D \rightarrow Exit$ and the time duration for visited pages as $(A, 30), (B, 42), (C, 118), (D, 91)$ and $T_{max} = 120sec$ then the action vector of each automata that is $(Start, A, B, C, D, E, F, G, H, Exit)$ is done based on equation (3) ,(4) and (5) is updated and the probability vector for each web page which is computed by learning automata as follows:

Start=(0,0.136,0.108,0.108,0.108,0.108,0.108,0.108,0.108,0.108)

A=(0.10625,0,0.15,0.10625,0.10625,0.10625,0.10625,0.10625,0.10625, 0.10625,0.10625)

B=(0.09875,0.09875,0,0.21,0.09875,0.09875,0.09875,0.09875,0.09875, 0.09875)

C=(0.101625,0.101625,0.101625,0,0.187,0.101625,0.101625,0.101625, 0.101625,0.101625)

D=(1/9, 1/9, 1/9, 1/9,0, 1/9, 1/9, 1/9, 1/9, 1/9)

Exit=(1/9, 1/9, 1/9, 1/9, 1/9, 1/9, 1/9, 1/9, 1/9,0)

p_k^j is the probability of choosing action α_k in learning automata LA_j . The value of p_k^j is computed in the first step of algorithm. For example, if the user access pattern for s_1 like is $Start \rightarrow A \rightarrow B \rightarrow C \rightarrow D \rightarrow Exit$ and the action probability values are like the previous example then $W_1 =< 0.136, 0.15, 0.21, 0.187, 0, 0, 0, 0, 0 >$.

3.2 Clustering Web Access Patterns by DLA

By increasing the number of access frequency to each web page and spending more time duration on each web page the amount of probability increases. In the second step of algorithm, we use DLA to group web access patterns into a number of clusters. In the proposed method for each web page like P_i $(1 \le i \le N)$ the learning automata LA_i is considered. Also we consider N be the number of clusters. The center of each cluster like $L(1 \le L \le N)$, can be denoted as $P_L = [p_L^i, p_L^{i+1}, p_L^{i+2}, \dots, p_L^m]$ $(1 \le L \le N)$. We suppose P_L^i is the probability of choosing action α_L in the learning automata of LA_i. The main steps of the algorithm are provided as follows:

Input: Web access patterns S
Output: N clustering centers
Step1. Create a DLA according to web pages structure and initialize action probability vector of each LA based on Eq.3
Step2.
Repeat
for L=1 to number of clusters **do**

// P_L^i is the probability of choosing action α_L in the learning automata of LA_i

// P_L is the cluster vector($1 \le L \le N$)

$$P_L = [p_L^i, p_L^{i+1}, p_L^{i+2}, \dots, p_L^m]$$

end for
for k =1 to number of users **do**
for L=1 to number of cluster vector

Compare each user weight vector W_k with vector P_L so that this Equation is satisfied.

$$\|W_k - P_L\|^2 =_L^{min} \|W_k - P_L\|^2$$

end for
end for

for i=1 to m //m is the number of automata

Enable action ith of LA_i according to the following equation:

$$p_L^i = p_L^i + a[1 - p_L]$$
$$p_m^i = (1-a)p_m^i \; \forall m \;\; L \neq m$$

end for
Until (no noticeable changes in the cluster vector)

The above algorithm classifies all user access patterns into N clusters, where the lth clusters is defined as $U^L = s_k \in S: \|W_k - P_L\|^2 ==_L^{min} \|E[W_k] - P_L\|^2$ where $(1 \le k \le N)$ and $(1 \le L \le N)$.

3.3 Clustering Centers of Clusters by Weighted c-means

In this step, the set of clustering centers $P_L(1 \le L \le N)$ generated in algorithm3.2 is further clustered based on the weighted c-means. Since $U^L(1 \le L \le N)$ includes different number of web access patterns, different weight is assigned to different $U^L(1 \le L \le N)$. The weight of $U^L(1 \le L \le N)$ is defined as follows:

$$w_i = \frac{\aleph(U^i)}{\sum_{j=1}^N \aleph(U^i)} \qquad (8)$$

Weighted c-means is applied to group $P_L(1 \le L \le N)$ into c different nonempty subsets. The main steps are described as follows:

Input: clustering center $P_L(1 \le L \le N)$ generated by the second step of algorithm
Output: Clustering results $(c\ clusters)$
Step1. Assign initial means $v_i(1 \le i \le c)$
Step2. According to the following membership function, assign each pattern P_L into the nearest cluster

(9)

$$u_{il} = \frac{1}{\sum_{j=1}^c \left(\frac{d(P_L, v_i)}{d(P_L, v_j)}\right)^{\frac{2}{m-1}}}$$

in which u_{il} is the membership degree of the P_L belonging to ith, $m \in (1, \infty)$ is a fuzzy parameter.
Step3. Recompute $v_i(1 \le i \le c)$ according to the following equation:

$$v_i = \frac{\sum_{L=1}^N w_L (u_{il})^m v_l}{\sum_{j=1}^N w_L (u_{il})^m} \qquad (10)$$

Step4. Repeat step 2 to step 3 until the following objective function convergence, i., there are no more new assignment.

$$J_m(M, V) = \sum_{l=1}^N w_L (u_{il})^m \sum_{i=1}^c d(P_L, v_i) \qquad (11)$$

In which $M = \{[u_{il}], 1 \le i \le c, 1 \le l \le N\}$ is a clustering matrix, $V = \{[v_i], 1 \le i \le c\}$ is the set of final clustering centers. After the algorithm 3.3 is executed, each $P_L(1 \le L \le N)$ belongs to c clusters according to different degrees. Each cluster center $v_i(1 \le i \le c)$. Every web access pattern s_i in U^l belongs to c clusters by the same degree with P_L.

4. An Experiment

Several preprocessing tasks have to be performed prior to clustering web access patterns from web logs. In this experiment, we just need data cleaning and simple session identification. 15,534 web access patterns are extracted from an information resource after a web log is downloaded [11]. Assume these web access patterns are grouped into 5 clusters, the clustering results are shown in Table 1. The same data is clustered into groups by other algorithms. In this paper, we compute the optimal number of clusters in terms of the Davies-Bouldin cluster validity index [2], which is a function of the ratio of the sum of within-cluster distance to between-cluster separation. The optimal clustering method for c clusters minimizes

$$DB = \frac{1}{c}\sum_{k=1}^c \max_{l \neq k} \left\{\frac{S(C_k) + S(C_l)}{d(C_k, C_l)}\right\}, \qquad (12)$$

Where $S(C_k)$ is the within-cluster and $d(C_k, C_l)$ is the between-cluster separation.

Table 1. Clustering result

Cluster number	The included web access patterns
Cluster 1	4567
Cluster 2	3478
Cluster 3	3589

Cluster 4	1897
Cluster 5	2003

In order to evaluate the results of clustering algorithms, the experiments have been done and the results of the proposed algorithm are compared to the different algorithms. Table 2 shows the Davies-Bouldin cluster validity index between our approach with other algorithms.

Table 2. Comparison Davies-Bouldin Index (DB) with Other Algorithms

Clustering algorithm	Davies-Bouldin Index
LVQ	0.697
DLA based approach	0.574
Fuzzy c-means	0.411
LVQ+fuzzyc-means[10]	0.335
Our Approach	0.232

Table 2 shows the DB criterion comparison between the proposed algorithms to different algorithms. As it is in the table the factor of DB in the proposed approach is lower than other algorithms. This shows that the proposed algorithm in the clustering of web access patterns has a higher proficiency. If only the first step of proposed algorithm (DLA based approach) is used, it won't have appropriate proficiency.

5. Conclusion

The visited web page and the time duration on it reflect the interest of web users. In this paper, each web access pattern from web logs is transformed into a weight vector using the learning automata then we put each weight vector in the nearest cluster using the learning automata. By doing this, a primitive clustering is performed on the web access patterns and the primitive centers of clusters are determined. Finally, these primitive clusters which have no or several access patterns are used by Fuzzy weighted c-means clustering algorithm and on the basis of the weight of each cluster which has been determined according to the number of web access patterns in each cluster and also the centers of clusters which have been determined by the learning automata are clustered. In order to avoid disadvantage of single DLA based approach or LVQ or Fuzzy weighted c-means, a hybrid approach based on DLA and Fuzzy weighted c-means is proposed to cluster web access patterns from web logs. Using this approach, the surfing behaviors of web users can be more quickly and exactly disclosed which is useful to build adaptive web server and design personalized service according to users' surfing behaviors.

Acknowledgement

This work has been supported by a grant from the Islamic Azad University, Shabestar Branch with Number: 5195489020400

References

[1] H. Beigy, and M. R. Meybodi , "A Learning Automata based Algorithms for Determination of Minimum Number of Hidden Unites for Three Layers Neural Networks", Journal of Amirkabir , Vol. 48, No. 4, 2002, pp. 957-974.
[2] J. Bezdek, and N. Pal, "Some New Indexes for Cluster Validity", IEEE Transactions on Systems, Man, and Cybernetics, Part-B, 1998, Vol. 28, pp. 957-974.
[3] S. De, and P. Krishna, "Clustering Web Transactions Using Rough Approximation", Fuzzy Sets and Systems, 2004 , Vol. 148, pp. 131-138.
[4] M. R. Meybodi, and H. Beigy, "A Note on Learning Automata based Schemes for Adaptation of BP Parameters", Journal of Neuro Computing, Vol. 48, 2002, pp. 957-974.
[5] M. R. Meybodi, and S. Lakshmivarahan, "On A class of Learning Algorithms Which have Symmetric Behavior under Success and Failure", Lecture Notes in Statistics, Berlin: Springer Verlag, 1984, pp. 145-155.
[6] K. S. Narendra, and M. A. L., Thathachar, Learning Automata: An introduction, Prentice Hall, 1989.
[7] T. Runkler, and J. Beadek, "Web Mining with Relational Clustering". International Journal of Approximate Reasoning, Vol. 32, 2003, pp. 217-236.
[8] P. Shi, "An Efficient Approach for Clustering Web Access Patterns from Web Logs", International Journal of Advanced Science and Technology, Vol. 5, 2009, pp. 1- 13.
[9] J. Srivastava, R. Cooley, M. Deshpande, and P.-N., Tan,. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, ACMSIGKDD Explorations, 2000.
[10] R. Wu, "Clustering Web Access Patterns based on Hybrid Approach", Fifth International Conference on Fuzzy Systems and Knowledge Discovery, 2008, Vol. 5, pp.52-56.
[11] http://ita.ee.lbl.gov/html/traces.html.

Babak Anari received the B.Sc. degrees in computer engineering from Azad University of Shabestar, Iran in 2002, and M.Sc. degrees from Azad University of Arak, Iran in 2007, respectively. He has some research papers in web mining and learning automata field.

Mohammad Reza Meybodi received the B.Sc. and M.Sc. degrees in economics from Shahid Beheshti University, Tehran, Iran, in 1973 and 1977, respectively, and the M.S. and Ph.D. degrees incomputer science from Oklahoma University, Norman, in 1980 and 1983, respectively. He is currently a Full Professor with Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran. Prior to his current position, he was an Assistant Professor with Western Michigan University, Kalamazoo, from 1983 to 1985 and an Associate Professor with Ohio University, Athens, from 1985 to 1991. He has many research papers in learning systems and in International Proceedings or in International Journals. He has many research papers in learning systems in International Proceedings or in International Journals.His research interests include learning systems, parallel algorithms, soft computing, and software development.

Zohreh Anari received the B.Sc. and M.Sc. degrees in computer engineering from Azad University of Shabestar, Iran in 2003 and

2009, respectively. She has some research papers in fuzzy web mining field. Her current research interests include learning automata, web mining and soft computing.