International Journal on Artificial Intelligence Tools
© World Scientific Publishing Company

EXTRACTING WEB USER PROFILES USING RELATIONAL COMPETITIVE FUZZY CLUSTERING

OLFA NASRAOUI, HICHEM FRIGUI
Department of Electrical and Computer Engineering
University of Memphis,
206 Engineering Science Bldg., Memphis, TN, 38152-3180
onasraou@memphis.edu, hfrigui@memphis.edu

RAGHU KRISHNAPURAM
Department of Mathematical and Computer Science
Colorado School of Mines, Golden, CO 80401
rkrishna@vindhya.mines.edu

ANUPAM JOSHI
Department of Computer Science and Electrical Engineering,
University of Maryland, Baltimore County,
Baltimore, MD, 21250 USA
joshi@cs.umbc.edu

The proliferation of information on the World Wide Web has made the personalization of this information space a necessity. An important component of Web personalization is to mine typical user profiles from the vast amount of historical data stored in access logs. In the absence of any *a priori* knowledge, unsupervised classification or clustering methods seem to be ideally suited to analyze the semi-structured log data of user accesses. In this paper, we define the notion of a "user session" as being a temporally compact sequence of Web accesses by a user. We also define a new distance measure between two Web sessions that captures the organization of a Web site. The Competitive Agglomeration clustering algorithm which can automatically cluster data into the optimal number of components is extended so that it can work on relational data. The resulting Competitive Agglomeration for Relational Data (CARD) algorithm can deal with complex, non-Euclidean, distance/similarity measures. This algorithm was used to analyze Web server access logs successfully and obtain typical session profiles of users.

Keywords: Web mining, User profile, Session similarity measure, Relational clustering, Competitive agglomeration, Fuzzy clustering.

1. Introduction

The proliferation of information on the World Wide Web has made the personalization of this information space a necessity. This need for personalization will have a significant impact on the way Web sites are designed and organized as well as the way documents are searched for and delivered.

Personaliztion is a recent and informally-articulated notion, and deals with tailoring a user's interaction with the Web information space based on information about him/her. For example, a person in Switzerland searching for ski resorts is more likely to be interested in the Alps, whereas a person in Colorado would likely be interested in the Rockies. Personalization can either be done via search engines such as Lycos, or by making Web sites adaptive. Initial work in this area has basically focused on creating recommender systems. One of the earliest such systems was the Firefly system[1] which attempted to provide CDs that best match a user's professed interests. More recently, systems such as W^3IQ[2] and PHOAKS[3] have sought to use cooperative information retrieval techniques for personalization. The Webwatcher project[4] at CMU highlights hyperlinks in a page based on the declared interests and the path traversal of a user as well as the path traversals of previous users with similar interests.

Mining typical user profiles from the vast amount of historical data stored in server or access logs is a possible approach to personalization that has been recently proposed. The standard K-Means algorithm was used to cluster users's traversal paths.[5] However, it is not clear how the similarity measure was devised and whether the clusters are meaningful. The WEBMINER system[6] discovers associations and sequential patterns between Web transactions based on Apriori algorithm.[7] It is important to mention that so far, most efforts have relied on relatively simple non-fuzzy techniques which can be inadequate for real user profile data. Such data, as we elaborate later, are usually fuzzy in nature. In the absence of any *a priori* knowledge, unsupervised classification or clustering methods seem to be ideally suited to analyze the semi-structured log data of user accesses by categorizing them into classes of user session profiles. We define the notion of a "user session" as being a temporally compact sequence of Web accesses by a user. The goal of our Web mining is to categorize these sessions. In this light, Web mining can be viewed as a special case of the more general problem of knowledge discovery in databases.[8] We define a new distance measure between two Web sessions that captures the organization of a Web site. This organizational information is inferred directly from the URLs.

Categories in most data mining tasks are rarely well separated. The class partition is best described by fuzzy memberships,[9] particularly along the overlapping borders. Specifically, in Web mining, certain access patterns may be regarded as belonging to more than one class with different degrees instead of belonging to only one class. Moreover, most clustering techniques have the disadvantage that they either assume that the number of clusters C is known, or attempt to determine this number by repeating the clustering process for many values of C and then selecting a partition according to a special validity criterion. For all these reasons, we use our fuzzy Competitive Agglomeration (CA) algorithm [10] which can automatically cluster data into the optimal number of components.

As will be explained later, the Web sessions are too complex to convert to simple numerical features. Hence, we propose a new approach to clustering the user sessions based on exploiting inter-session similarities within a relational framework. However, CA deals with object or feature data only. Moreover, the new Web session dissimilarity measure defined in Section 2 is not Euclidean. Therefore, we extend the CA so that it can work on non-Euclidean relational data. The resulting Competitive Agglomeration for Relational Data (CARD) algorithm can deal with complex and subjective distance/similarity measures which are not restricted to be Euclidean. In addition, CARD is reliable, computationally attractive, and practically insensitive to initialization. The Web site for the department of Computer Engineering and Computer Sciences at the University of Missouri was used as a testbed for CARD which successfully analyzed the server access logs and obtained typical session profiles of users.

The rest of the paper is organized as follows, where the sections represent different phases of the knowledge discovery process [8] of categorization of user profiles. In Section 2, we define a new similarity between Web user sessions. In Section 3, we review the CA algorithm and extend it to work on non-Euclidean relational data, resulting in CARD. In Section 4, we define quantitative measures to help us in the interpretation and evaluation of the results of mining the access log data. In Section 5, we present our experimental results. Finally, we conclude with a discussion of our ongoing work in Section 6.

2. Defining the Similarity Between User Sessions

In order to extract the user profile categories from the log data, we start by preprocessing and segmenting the semi-structured log data into individual Web user sessions. Then, we compute the relation matrix consisting of all pairwise Web session similarities. These two preliminary steps are described below.

2.1. *Preprocessing and Segmentation of the access log data into sessions*

The access log for a given Web server consists of a record of all files accessed by users. Each log entry consists of: **(i)** User's IP address, **(ii)** Access time, **(iii)** Request method ("GET", "POST", $\cdots$), etc), **(iv)** URL of the page accessed, **(v)** Data transmission prototcol (typically HTTP/1.0), **(vi)** Return code, **(vii)** Number of bytes transmitted. First, we filter out log entries that are not germane for our task. These include entries that: **(i)** result in any error (indicated by the error code), **(ii)** use a request method other than "GET", or **(iii)** record accesses to image files (.gif, .jpeg, , $\cdots$, etc), which are typically embedded in other pages and are only transmitted to the user's machine as a by product of the access to a certain Web page which has already been logged.

Next, analogous to WEBMINER's association transaction model,[6] the individual log entries are grouped into user sessions. A user session is defined as a sequence of temporally compact accesses by a user. Since Web servers do not typically log usernames (unless *identd* is used), we define a user session as accesses from the same

IP address such that the duration of time elapsed between any two consecutive accesses in the session is within a prespecified threshold. Each URL in the site is assigned a unique number $j \in \{1, \ldots, N_U\}$, where N_U is the total number of valid URLs. Thus, the i^{th} user session is encoded as an N_U-dimensional binary attribute vector $\mathbf{s}^{(i)}$ with the property

$$s_j^{(i)} = \begin{cases} 1 & \text{if the user accessed the } j^{th} \text{ URL during the } i^{th} \text{ session,} \\ 0 & \text{otherwise.} \end{cases}$$

The ensemble of all N_S sessions extracted from the server log file is denoted by $\mathcal{S}$. Note that our scheme will map one user's multiple sessions to multiple user sessions. However, this is not of concern since our attempt is to extract "typical user session profiles". If we assume that the majority of a user's sessions follow a similar profile then clearly no difference is made. On the other hand, this notion of multiple user sessions enables us to better capture the situation when the same user displays a few (different) access patterns on this site. Our approach can be combined with other features (such as cookies) to actually map a typical profile back to a particular user.

2.2. *Adaptation of Session Data to Clustering: Computing The Relation Matrix*

In the absence of any a priori knowledge, an unsupervised classification or clustering method seems to be ideally suited to partition the user sessions. There are two major classes of clustering techniques, those that work with object data or feature vectors, and those that work on relational data (a set of similarities or dissimilarities between the data). Even though the first class of clustering algorithms has been the most popular one and has received a lot of attention, it is not suitable for clustering user sessions. This is because of the high dimensionality of the feature space (there are usually several hundred URLs in a typical Web site), and the likely correlation between the features (URLs that often co-occur in the same session). The Web sessions are too complex to convert to simple numerical features, partly because the organization of the Web site must be taken into account. In fact, the URLs in a site have a hierarchical or tree-like structural composition. Therefore, we define a similarity measure between two sessions that incorporates both the structure of the site, as well as the URLs involved.

We chose the relational approach to clustering since our data (sessions) are not numeric in nature. This approach requires the definition and computation of the dissimilarity/similarity between all session pairs (i.e., the relation matrix) prior to the clustering process. We start by defining the similarity measure between two user-sessions: $\mathbf{s}^{(k)}$ and $\mathbf{s}^{(m)}$. We first consider the simple case where the individual attributes or URLs accessed in the sessions are totally independent and the structure of the site is ignored. Then, we can simply use the cosine of the angle between $\mathbf{s}^{(k)}$ and $\mathbf{s}^{(l)}$ as a measure of similarity

$$S_{1,kl} = \frac{\sum_{i=1}^{Nu} s_i^{(k)} s_i^{(l)}}{\sqrt{\sum_{i=1}^{Nu} s_i^{(k)}}\sqrt{\sum_{i=1}^{Nu} s_i^{(l)}}} \tag{1}$$

It can be seen that $S_{1,kl}$ simply measures the number of identical URL's accessed during the two sessions relative to the number of URL's accessed in both sessions. $S_{1,kl}$ has the desirable properties: $S_{1,kk} = 1, S_{1,kl} = S_{1,lk}$, and $S_{1,kl} > 0/\forall k \neq l$. The problem with this similarity measure is that it completely ignores the hierarchical organization of the Web site, which will adversely affect the ability to capture correct profiles. For example, the session pair {/courses/cecs345} and {/courses/cecs343}, as well as the session pair {/courses/cecs345} and {/research/grants}. will receive a 0 similarity score according to S_1. However, it is evident that the first two sessions are more similar than the second pair, because both users in the first sessions seem to be interested in courses. In addition to capturing similar interest, the similarity measure should take into account the level of specificity of the requests. For instance, one would expect the sessions {/courses/cecs345/projects/proj1} to be more similar to {/courses/cecs345/projects} than to {/courses/cecs343} because there is more overlap between the URLs in the first two sessions along the directory hierarchy tree. This leads us to define a similarity measure on the structural URL level that will be used in the computation of the similarity at the session level. This inter-URL similarity measure should take into account the fact that the placement of the URLs along the Web site's directory hierarchy tends to be based on their conceptual similarity. This means that it is reasonable to assume that URLs which are placed under the same subdirectory are somewhat related. For this reason, we first provide a syntactic model for the entire Web site as a tree with the nodes representing different URLs. The tree is similar to that of a directory where an edge connects one node to another if the URL corresponding to the latter is hierarchically located under that of the former, for example {/courses} and {/courses/cecs345}. The root of the tree (the node with no incoming edges) corresponds to the highest level URL in the Web site (/), usually corresponding to the main or home page. Taking into account this syntactic representaion of two URLs, their similarity will be assessed by comparing the location of their corresponding nodes on the tree. This is done by comparing the paths from the root of the tree to the two nodes. Hence, we define the "syntactic" similarity between the i^{th} and j^{th} URLs as

$$S_u(i,j) = \min\left(1, \frac{|p_i \cap p_j|}{\max\left(1, \max\left(|p_i|, |p_j|\right) - 1\right)}\right) \tag{2}$$

where p_i denotes the path traversed from the root node to the node corresponding to the i^{th} URL, and $|p_i|$ indicates the length of this path or the number of edges included in the path. Note that this similarity which lies in $[0,1]$ basically measures the amount of overlap between the paths of the two URLs. Now the similarity on the session level which incorporates the syntactic URL similarities is defined by correlating all the URL attributes and their similarities in two sessions as follows

$$S_{2,kl} = \frac{\sum_{i=1}^{N_U} \sum_{j=1}^{N_U} s_i^{(k)} s_j^{(l)} S_u(i,j)}{\sum_{i=1}^{N_U} s_i^{(k)} \sum_{j=1}^{N_U} s_j^{(l)}} \tag{3}$$

Unlike S_1, this similarity uses soft URL level similarieties that vary between 0 and 1 depending on their conceptual similarity instead of 0 or 1, if they are different or identical respectively. For the special case when all the URLs accessed during session $s^{(k)}$ have zero similarity with the URLs accessed during session $s^{(l)}$, i.e., $S_u(i,j) = 0$ if $i \neq j$, $S_{2,kl}$ reduces to $S_{2,kl} = \sum_{i=1}^{N_U} s_i^{(k)} s_i^{(l)} / \sum_{i=1}^{N_U} s_i^{(k)} \sum_{j=1}^{N_U} s_j^{(l)}$ and when the two sessions are idential, this value further simplifies to $S_{2,kk} = 1/\sum_{i=1}^{N_U} s_i^{(k)}$ which can be considerably small depending on the number of URLs accessed. This means that this similarity measure will be rather unintuitive, because ideally the similarity should be maximal for two identical sessions. Besides identical sessions, this similarity will generally be underestimated for session pairs who share some identical URLs while the rest of the unshared URLs have low syntactic similarity. In general for such sessions where the syntactic URL similarities are low, $S_{1,kl}$ provides a higher and more accurate session similarity. On the other hand, when the syntactic URL similarities are high, $S_{2,kl}$ is higher and more accurate. Therefore, we can define a new similarity between two Web sessions that takes advantage of the desirable properties of S_1 and S_2 by forming a maximally optimistic aggregation as follows:

$$S_{kl} = \max(S_{1,kl}, S_{2,kl}) \tag{4}$$

Finally, for the purpose of relational clustering, this similarity is mapped to the dissimilarity measure $d_s^2(k,l) = (1 - S_{kl})^2$. Note that squaring the complement of the similarity has the effect of amplifying the difference between similar and different sessions. This dissimilarity measure satisfies the desirable properties of a "distance" measure: $d_s^2(k,k) = 0$, $d_s^2(k,l) >= 0 \ \forall k,l$, and $d_s^2(k,l) = d_s^2(l,k) \ \forall k,l$. However, it is not a true "metric" distance measure for the following reasons. First, as a result of the special properties of the URL level similarity defined in (2), it is possible for two distinct sessions to have zero dissimilarity. This occurs whenever $\sum_{i=1}^{N_U} \sum_{j=1}^{N_U} s_i^{(k)} s_j^{(l)} S_u(i,j) = \sum_{i=1}^{N_U} s_i^{(k)} \sum_{j=1}^{N_U} s_j^{(l)}$, or equivalently $\sum_{j=1}^{N_U} s_i^{(k)} s_j^{(l)} S_u(i,j) = s_i^{(k)} \sum_{j=1}^{N_U} s_j^{(l)}$ for all $i = 1, \ldots, N_U$. This is particularly true if the URL level similarities are 1 for all the URLs accessed in the two sessions. A typical example consists of the singleton sessions {/courses/cecs345/syllabus.html} and {/courses/cecs345}. This property is actually desirable for our application, because we consider these two sessions to fit the same profile. In other words our URL level similarity considers a particular URL and the URL corresponding to its parent directory as being maximally related or having a URL level similarity of one. Similarly, any two URLS that share the same parent, i. e., that are directly located under the same subdirectory are deemed to be maximally related. This is a reasonable assumption because such URLs generally share the same subject.

The Web session dissimilarity measure also violates the triangular inequality for metric distances in some cases. For instance, the dissimilarity between the sessions {/courses/cecs345/syllabus} and {/courses/cecs345} is zero. So is the dissimilarity between {/courses/cecs345} and {/courses/cecs401}. However, the dissimilarity between {/courses/cecs401} and {/courses/cecs345/syllabus} is not zero (it is 1/4). This illustrates another desirable property for profiling sessions which is that the dissimilarity becomes more stringent as the accessed URLs get farther from the root because the amount of specificity in user accesses increases correspondingly. Hence, the proposed dissimilarity measure fits our subjective criteria of Web session similarity.

3. Clustering the User Sessions Using CARD

3.1. *The Competitive Agglomeration (CA) algorithm*

Let $\mathcal{X} = \{\mathbf{x}_j \mid j = 1, \ldots, N\}$ be a set of N vectors and let $\mathbf{B} = (\beta_1, \ldots, \beta_c)$ represent a C-tuple of prototypes each of which characterizes one of the C clusters. The CA algorithm [10] minimizes

$$J(\mathbf{U}, \mathbf{B}; \mathcal{X}) = \sum_{i=1}^{C} \sum_{j=1}^{N} (u_{ij})^2 d^2(\mathbf{x}_j, \beta_i) \; - \; \alpha \sum_{i=1}^{C} \left[\sum_{j=1}^{N} u_{ij} \right]^2 \tag{5}$$

subject to

$$\sum_{i=1}^{C} u_{ij} = 1, \qquad \text{for } j \in \{1, \cdots, N\}. \tag{6}$$

In (5), $d^2(\mathbf{x}_j, \beta_i)$ represents the distance from feature vector $\mathbf{x}_j$ to the prototype β_i, u_{ij} represents the degree of membership of feature point $\mathbf{x}_j$ in cluster β_i, and $\mathbf{U} = [u_{ij}]$ is a $C \times N$ constrained fuzzy C-partition matrix.[9] It should be noted that the number of clusters C in (5) is dynamically updated in the CA algorithm. The first component in (5) allows us to control the shapes and sizes of the clusters and to obtain compact clusters. The global minimum of this component is achieved when the number of clusters C is equal to the number of samples N. The second component in (5) is the sum of squares of the cardinalities of the clusters which allows us to control the number of clusters. The global minimum of this term (including the negative sign) is achieved when all points are lumped into one cluster, and all other clusters are empty. When both components are combined and α is chosen properly, the final partition will minimize the sum of intra-cluster distances, while partitioning the data set into the smallest possible number of clusters. The clusters which are depleted as the algorithm proceeds will be discarded, as explained later.

To minimize (5) with respect to $\mathbf{U}$, we apply Lagrange multipliers and obtain

$$\begin{aligned} J(\mathbf{U}, \mathbf{B}; \mathcal{X}) \;=\; & \sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^2 d^2(\mathbf{x}_j, \beta_i) \;-\; \alpha \sum_{i=1}^{C}\left[\sum_{j=1}^{N} u_{ij}\right]^2 \;- \\ & \sum_{j=1}^{N}\lambda_j\Big(\sum_{i=1}^{C} u_{ij} - 1\Big). \end{aligned} \tag{7}$$

We then fix $\mathbf{B}$ and solve

$$\frac{\partial J}{\partial u_{st}} = 2u_{st}d^2(\mathbf{x}_t, \beta_s) \;-\; 2\alpha\sum_{j=1}^{N} u_{sj} - \lambda_t = 0, \tag{8}$$

to obtain an updating equation for the memberships u_{st}. Equation (8) is a set of $N \times C$ linear equations with $N \times C + N$ variables (u_{st}, and λ_t). Using them in conjunction with the N equations resulting from the constraint in (6), one can solve for the $N \times C + N$ variables. However, the solution can be simplified considerably by assuming that the membership values do not change significantly from one iteration to the next, and by computing the term $\sum_{j=1}^{N} u_{sj}$ in (8) using the membership values from the previous iteration. With this assumption, (8) reduces to

$$u_{st} = \frac{2\alpha \times N_s + \lambda_t}{2d^2(\mathbf{x}_t, \beta_s)}, \tag{9}$$

where

$$N_s = \sum_{j=1}^{N} u_{sj}. \tag{10}$$

is the cardinality of cluster s. Using (9) and the constraint in (6), we obtain

$$u_{st} = u_{st}^{\mathrm{FCM}} + u_{st}^{\mathrm{Bias}}, \tag{11}$$

where

$$u_{st}^{\mathrm{FCM}} = \frac{\frac{1}{d^2(\mathbf{x}_t, \beta_s)}}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_t, \beta_k)}}, \tag{12}$$

and

$$u_{st}^{\mathrm{Bias}} = \frac{\alpha}{d^2(\mathbf{x}_t, \beta_s)}\big(N_s - \overline{N}_t\big). \tag{13}$$

In (13), $\overline{N}_t$ is defined as

$$\overline{N}_t = \frac{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_t, \beta_k)} N_k}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_t, \beta_k)}},$$

which is simply a weighted average of the cluster cardinalities, where the weight of each cluster reflects its proximity to the feature point $\mathbf{x}_t$ in question.

As seen from (11), u_{st}^{FCM} in (12), is the membership term in the FCM algorithm[9] which takes into account only the relative distances of the feature point to all clusters. The second component in (11), u_{st}^{Bias}, is a signed bias term which depends on the difference between the cardinality of the cluster of interest, and the weighted average of cardinalities from the point of view of feature point $\mathbf{x}_t$. For clusters with cardinality higher than average, the bias term is positive, thus appreciating the membership value. On the other hand, for low cardinality clusters, the bias term is negative, thus depreciating the membership value. This leads to a gradual erosion of the cardinality of spurious clusters. When the cardinality of a cluster drops below a threshold, we discard the cluster, and update the number of clusters. Since the initial partition has an overspecified number of clusters, each cluster is approximated by many small clusters in the beginning. As the algorithm proceeds, the second term in (5) causes each cluster to expand and include as many points as possible. At the same time, the constraint in (6) causes adjacent clusters to compete. As a result, only a few clusters will survive, while others will shrink and eventually become extinct.

It should be noted that when a feature point $\mathbf{x}_j$ is close to only one cluster (say cluster i), and far from other clusters, we have

$$N_i \approx \overline{N}_j, \qquad \text{or} \qquad u_{ij}^{\text{Bias}} \approx 0.$$

In this case the membership value, u_{ij}, is independent of the cluster cardinalities, and reduces to u_{ij}^{FCM}. In other words, if a point is close to only one cluster, it will have high membership value in this cluster and no competition is involved. On the other hand, if a point is close to many clusters, these clusters will compete for this point based on cardinality. This encourages the formation of larger clusters, i.e., it promotes agglomeration.

The choice of α in (5) is important in the CA algorithm since it reflects the importance of the second term relative to the first term. To make the algorithm independent of the distance measure, α should be proportional to the ratio of the two terms. That is

$$\alpha \propto \frac{\sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^2 d^2(\mathbf{x}_j,\beta_i)}{\sum_{i=1}^{C}\left[\sum_{j=1}^{N} u_{ij}\right]^2}.$$

In all examples described in this paper, we choose α to be

$$\alpha(k) = \eta(k)\frac{\sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^2 d^2(\mathbf{x}_j,\beta_i)}{\sum_{i=1}^{C}\left[\sum_{j=1}^{N} u_{ij}\right]^2}. \tag{14}$$

In (14), α and η are functions of the iteration number k. A good choice for η is the exponential decay defined by

$$\eta(k) = \eta_0 e^{-k/\tau}, \tag{15}$$

where η_0 is the initial value and τ is the time constant. It can easily be verified from (15) that when $\eta_0 > 1$, more emphasis is placed on the second term of the criterion to bolster the competition process in the early iterations, whereas the first term is emphasized in the later iterations, when the number of clusters has supposedly reached a stable value to provide more accurate estimates of the cluster prototypes.

It should be noted that depending on the value of α, the membership values u_{ij} may not be confined to [0, 1]. In fact, u_{ij} can become negative if N_i is very small and point $\mathbf{x}_j$ is close to other dense clusters (i.e. $\overline{N}_j$ is high). This generally implies that cluster β_i is spurious. In this case, it is safe to set u_{ij} to zero to indicate that feature vector $\mathbf{x}_j$ is atypical of cluster β_i. It is also possible for u_{ij} to become larger than 1 if N_i is very large and feature point $\mathbf{x}_j$ is close to other spurious clusters (i.e. $\overline{N}_j$ is low). In this case it is safe to clip u_{ij} to 1 to indicate that feature vector $\mathbf{x}_j$ is typical of cluster β_i.

Minimization of (5) with respect to the prototypes to derive the update equation for $\mathbf{B} = (\beta_1, \ldots, \beta_c)$, varies according to the choice of the prototypes and the distance measure. Each choice leads to a different algorithm. Since the second term in (5) does not depend on the prototypes and the distance measure, the parameter update equations in the CA algorithm are the same as those in traditional objective-function-based fuzzy clustering algorithms that minimize the sum of squared distances.[11] For instance, to detect spherically shaped clusters, we use the Euclidean distance,

$$d^2(\mathbf{x}_j, \beta_i) = \|\mathbf{x}_j - \mathbf{c}_i\|^2 , \tag{16}$$

where $\mathbf{c}_i$ is the center of cluster β_i. By differentiating the clustering criterion resulting from substituting the distances (16) into (5), it is easy to show that the equation for the centers which minimize this objective function for the i^{th} cluster, are given by

$$\mathbf{c}_i = \frac{\sum_{j=1}^{N} (u_{ij})^2 \mathbf{x}_j}{\sum_{j=1}^{N} (u_{ij})^2}. \tag{17}$$

When the clusters are ellipsoidally shaped, it is appropriate to estimate the center and covariance matrix for each cluster and use the following distance measure proposed by Sebestyen and Roemder [12] (and later independently derived by Gustafson and Kessel [13])

$$d^2(\mathbf{x}_j, \beta_i) = d^2_{Cij} = \left|\mathbf{C}_i\right|^{1/n} (\mathbf{x}_j - \mathbf{c}_i)^T \mathbf{C}_i^{-1} (\mathbf{x}_j - \mathbf{c}_i), \tag{18}$$

where $\mathbf{c}_i$ is the center of cluster β_i, and $\mathbf{C}_i$ is its covariance matrix. We refer to this measure as the GK distance measure in the simulation experiments. Since the second term in (5) does not depend explicitly on the prototype parameters, $\beta_i = (\mathbf{c}_i, \mathbf{C}_i)$, the necessary conditions for the minimization of (5) with respect to $\mathbf{B}$ are the same as those in the Gustafson-Kessel [13] (G-K) algorithm (with m=2), i. e., the centers are updated as in (17), and the covariance matrices are updated using

$$\mathbf{C}_i = \frac{\sum_{j=1}^{N} (u_{ij})^2 \left(\mathbf{x}_j - \mathbf{c}_i\right)\left(\mathbf{x}_j - \mathbf{c}_i\right)^T}{\sum_{j=1}^{N} (u_{ij})^2}. \tag{19}$$

3.2. *Synthetic Simulation results*

Before extending the CA algorithm to the case of relational data, we illustrate its effectiveness in clustering 2-dimensional object data because the results are easier to inspect in this case. In all examples shown in this section, a cluster β_i was discarded if its cardinality (N_i) is less than 5. The initial value of η (η_0) was set to 5 and the time constant τ was set to 10. The initial partition is obtained by running the FCM algorithm [9] (with the Euclidean distance measure and m=2) for 5 iterations. The FCM algorithm is then applied with the appropriate distance measure for 5 iterations. This allows the clusters to be approximated by many prototypes at the beginning. In our experience, the CA algorithm is independent of the initial number of clusters C_{max} as long as C_{max} is chosen to be much larger than the expected number of clusters. In all examples, we tried many different values for C_{max} ranging from 15 to 30, and in all cases, the algorithm converged to the same final partition. In this paper we display the intermediate partitions of the algorithm when C_{max} is chosen to be 20.

In Fig.1, we illustrate the performance of the CA algorithm when the Euclidean distance is used. Fig.1(a) is a synthetic data set that consists of four clusters of various sizes and densities. Fig.1(b) shows the prototype parameters of the initial partition. The prototypes (centers) locations are shown as "+" symbols superimposed on the data set. As can be seen in this figure, large clusters are split into many clusters whereas small clusters are split into fewer clusters. Fig.1(c) shows the intermediate results after 3 iterations where the number of clusters is reduced to 13. Fig.1(d) shows the final results of the CA algorithm (after 15 iterations).

In Fig.2, we illustrate the performance of the CA algorithm when the Gustafson-Kessel (G-K) distance [13] is used to find ellipsoidal clusters. Fig.2(a) shows a data set containing six Gaussian clusters of various sizes, shapes, and orientations. Fig.2(b) shows the initial prototypes. As before, The "+" signs indicate the cluster centers, and the ellipses shown in the figure enclose points having a Mahalanobis distance less than 9. Fig.2(c) shows the prototypes obtained after 4 iterations of the CA algorithm where the number of clusters is reduced to 10. Fig.2(d) shows the final results after convergence of the CA algorithm (after 12 iterations).

3.3. *Extension of the Competitive Agglomeration (CA) algorithm to Non-Euclidean Relational Data*

The CA was originally formulated to deal with object or feature data only. For our application, the CA must be extended so that it can work on a matrix of relations or pairwise dissimilarities between the data entities. The approach that we take in

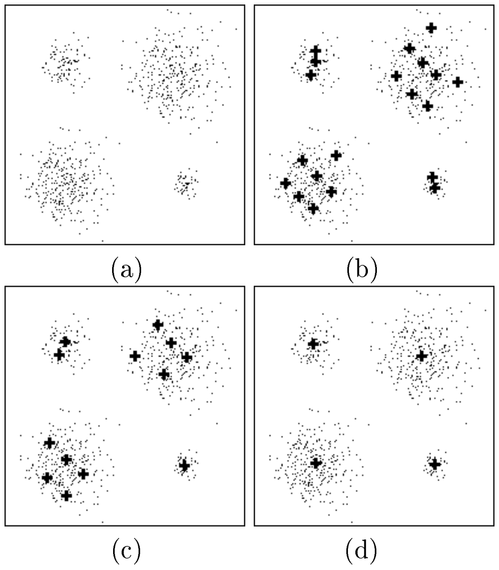

Fig. 1. Intermediate results of the CA algorithm on a data set with spherical clusters (a) original image, (b) initial prototypes, (c) results after 3 iterations, and (d) final results.

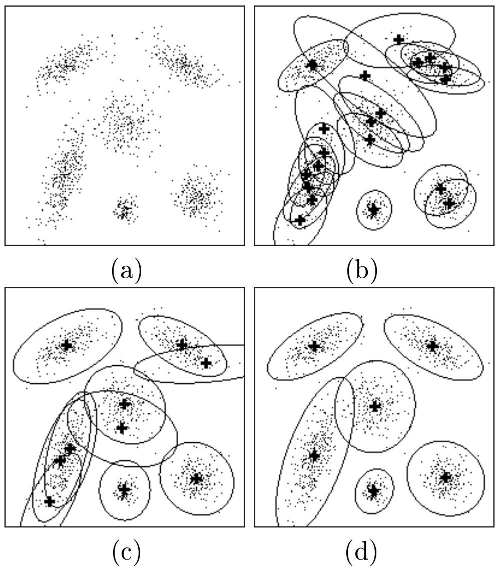

Fig. 2. Intermediate results of the CA algorithm on a data set with ellipsoidal clusters (a) original image, (b) initial prototypes, (c) results after 4 iterations, and (d) final results.

order to accomplish this goal is similar to the one taken by Hathaway et al.[14] when they formulated the relational dual of the Fuzzy C Means (FCM) algorithm. One way to obtain a partition of the data objects implicitly inducing a given relation matrix, is to derive the distances from the implicit objects to a set of C implicit prototypes that summarize the data objects of each cluster in the partition. For the case when the relation matrix corresponds to Euclidean pairwise distances between data in a Euclidean vector space, it can be proved[14] that the squared Euclidean distance, $d_{ik}^2 = \|\mathbf{x}_j - \mathbf{c}_i\|^2$, from feature vector $\mathbf{x}_j$ to the center of the i^{th} cluster, $\mathbf{c}_i$, can be written in terms of the relation matrix $\mathbf{R}$ as follows:

$$d_{ik}^2 = (\mathbf{R}\mathbf{v}_i)_k - \mathbf{v}_i\mathbf{R}\mathbf{v}_i/2, \tag{20}$$

where $\mathbf{v}_i$ is the membership vector defined by

$$\mathbf{v}_i = \frac{(u_{i1}^m, \ldots, u_{iN}^m)^t}{\sum_{j=1}^N u_{ij}^m}. \tag{21}$$

Equation (20) allows the computation of the distance between the data points and cluster prototypes in each iteration when only the relational data, $\mathbf{R}$, are given, starting with a set of initial fuzzy memberships, u_{ij}, that describe the degree of belongingness of the j^{th} data object in the i^{th} cluster. Once that the implicit distance values, d_{ik}^2, have been computed using (20), the fuzzy memberships can be updated to optimize the clustering criterion, resulting in a new fuzzy partition of the data. When these two steps are repeated, an iterative optimization process is created to optimize the objective function of the clustering algorithm. Therefore, a relational dual of CA exists for the special case where the object data and relational data satisfy

$$\mathbf{R} = [R_{ij}] = \|\mathbf{x}_j - \mathbf{c}_i\|^2 \tag{22}$$

This means that even when only relational data is available in the form of an $N \times N$ relation matrix, the relational dual of CA is expected to perform in an equivalent way to CA provided that the relation matrix, $\mathbf{R}$, is Euclidean, i.e., there exists a set of N points in $\mathcal{R}^{n-1}$, called a realization of $\mathbf{R}$, satisfying (22).

When a realization does not exist for the relation matrix, $\mathbf{R}$, the relational dual of the CA may fail mainly because some of the distances computed using (20) may be negative. To overcome this problem, we use the β-spread transform [15] to convert a non-Euclidean matrix $\mathbf{R}$ into an Euclidean Matrix $\mathbf{R}_\beta$ as follows

$$\mathbf{R}_\beta = \mathbf{R} + \beta\,(\mathbf{M} - \mathbf{I}) \tag{23}$$

where β is a suitably chosen scalar, $\mathbf{I} \in \mathcal{R}^{n\times n}$ is the identity matrix and $\mathbf{M} \in \mathcal{R}^{n\times n}$ satisfies $M_{ij} = 1$ for $1 \le i, j \le n$. It was suggested in [15] that the distances d_{ik}^2 be checked in every iteration for negativity, which indicates a non-Euclidean relation matrix. In that case, the β-spread transform should be applied with a suitable value of β to make the d_{ik}^2 positive again. An underestimate for the lower bound on β was derived [15] and related to the necessary shift that is needed to make the distances positive. This result can be summarized as

$$\Delta\beta = \max_{i,k}\{-2d_{ik}^2/\left\|\mathbf{v}_i - \mathbf{e}_k\right\|^2\}, \tag{24}$$

where $\mathbf{e}_k$ denotes the k^{th} column of the identity matrix. An alternative approach [16] to this problem imposes Kuhn-Tucker conditions to ensure positivity of the memberships.

The resulting CARD algorithm is summarized below:

```
The Competitive Agglomeration For Relational Data (CARD) Algorithm
Fix the maximum number of clusters $C = C_{max}$;
Initialize $k = 0$; $\beta = 0$; $\mathbf{U}^{(0)}$; $N_i$, $1 \le i \le C$ using (10);
Repeat
    Compute membership vectors $\mathbf{v}_i$    for $1 \le i \le C$ using (21);
    Compute $d_{ik}^2 = (\mathbf{R}_\beta \mathbf{v}_i)_k - \mathbf{v}_i^t \mathbf{R}_\beta \mathbf{v}_i/2$ for $1 \le i \le C$ and $1 \le k \le N_S$;
    If ($d_{ik}^2 < 0$ for any $i$ and $k$) then {
        Compute $\Delta\beta$ by using (24);
        Update $d_{ik}^2 \leftarrow d_{ik}^2 + (\Delta\beta/2) * \|\mathbf{v}_j - \mathbf{e}_k\|^2$ for $1 \le i \le C$ and $1 \le k \le N_S$;
        Update $\beta = \beta + \Delta\beta$;
    }
    Update $\alpha(k)$ using (14); Update $\mathbf{U}^{(k)}$ using (11); Compute $N_i$ using (10);
    If ($N_i < \epsilon_1$) Discard $i^{th}$ cluster and update $C$;
    $k = k + 1$ ;
Until (memberships stabilize).
```

4. Interpretation and Evaluation of the Results

The results of applying CARD on the user session relational data are interpreted using the following quantitative measures. First, the user sessions are assigned to the closest clusters based on the distances computed in (20). This creates C clusters $\mathcal{X}_i = \left\{ \mathbf{s}^{(k)} \in \mathcal{S} \mid d_{ik} < d_{jk} \; \forall j \neq i \right\}$, for $1 \le i \le C$.

After the crisp assignment of user sessions to the automatically determined number (C) of clusters, the sessions in cluster $\mathcal{X}_i$ are summarized in a typical session "profile" vector $\mathbf{P}_i = \left(P_{i1}, \ldots, P_{iN_U}\right)^t$. The components of $\mathbf{P}_i$ are URL weights which represent the "probability of access" of each URL during the sessions of $\mathcal{X}_i$ as follows

$$P_{ij} = p\left(\mathbf{s}_j^{(k)} = 1 | \mathbf{s}_j^{(k)} \in \mathcal{X}_i\right) = \frac{|\mathcal{X}_{i_j}|}{|\mathcal{X}_i|}, \tag{25}$$

where $\mathcal{X}_{i_j} = \left\{ \mathbf{s}^{(k)} \in \mathcal{X}_i \mid s_j^{(k)} > 0 \right\}$. The URL weights P_{ij} measure the significance of a given URL to the i^{th} profile. Besides summarizing profiles, the components of the profile vector can be used to recognize an invalid profile which has no strong or frequent access pattern. For such a profile, all the URL weights will be low.

Several classical cluster validity measures can be used to assess the goodness of the partition. The intra-cluster or within-cluster distance represents an average of the distances between all pairs of sessions within the the i^{th} cluster, and is given by $\overline{D}_{Wi} = \sum_{\mathbf{s}^{(k)} \in \mathcal{X}_i} \sum_{\mathbf{s}^{(l)} \in \mathcal{X}_i, l \neq k} d_{kl}^2 / |\mathcal{X}_i| \, (|\mathcal{X}_i| - 1)$. This is inversely related to the compactness or goodness of a cluster. A good guideline to use when evaluating clusters based on the intra-cluster distances is to compare these values to the total average pairwise distance of all sessions. The latter corresponds to the intra-cluster distance if all the user sessions were assigned to one cluster (i.e., no category information is used). Also it is important to recall that all distances are in $[0, 1]$. The inter-cluster or between-cluster distance represents an average of the distances between sessions from the i^{th} cluster and sessions from the j^{th} cluster, and is given by $\overline{D}_{Bij} = \sum_{\mathbf{s}^{(k)} \in \mathcal{X}_i} \sum_{\mathbf{s}^{(l)} \in \mathcal{X}_j, l \neq k} d_{kl}^2 / |\mathcal{X}_i| \, |\mathcal{X}_j|$. For a good partition, the inter-cluster distances should be high because they measure the separation between clusters.

5. Mining Web User Profiles using CARD

The Web mining procedure described in the previous sections was used to extract typical user session profiles from the log data of the Web site for the department of Computer Engineering and Computer Sciences at the University of Missouri at Columbia. The log data from accesses to the server during a period of 12 days was used. After filtering out irrelevent entries, the data was segmented into 1703 sessions. The maximum elapsed time between two consecutive accesses in the same session was set to 45 minutes. The number of distinct URLs accessed in valid entries was 369. While applying CARD to the relational data, a cluster was discarded if its cardinality (N_i) was less than 5. The initial value of η (η_0) was set to 0.0002, the time constant τ was set to 10, and C_{max} was chosen to be 50. The initial distance values d_{ik}^2 were obtained by randomly choosing C_{max} rows from the relation matrix and computing the fuzzy memberships.

After clustering the relational data with CARD, the final number of clusters was 20. Table 1 illustrates four profiles computed using (25), where only the significant URLs ($P_{ij} > 0.15$) are displayed, and the individual components are displayed in the format $\{P_{ij}$ - j^{th} URL$\}$. The sessions were assigned to the closest cluster and the session clusters or profiles were examined qualitatively and are summarized in Table 2 which also lists the cardinality and the intra-cluster distance.

The results show that CARD succeeded in delineating many different profiles in the user sessions. Except for the 14^{th} cluster, all clusters correspond to real profiles reflecting distinct user interests. The profiles followed the access patterns on typical users – the general "outside visitor" is captured in profiles 1 and 16, prospective students in profile 7, students in CECS438/CECS352 (taught by the same faculty member) in profile 6 etc. The goodness of these clusters is recognizable through their low intra-cluster distances (considerably lower than the total pairwise session

Table 1. Web user profile examples.

i	$\mathbf{P}_i$
5	{.75 - /cecs_computer.class} {.94 - /courses.html} {.98 - /courses_index.html} {.95 - /courses10.html} {.34 - /courses30.html} {.20 - /courses_Webpg.html} {.28 - /courses20.html} {.85 - /}
6	{.58 - /~joshi/courses/cecs352} {.27 - /~joshi/courses/cecs352/slides-index.html} {.20 - /~joshi/courses/cecs352/text.html} {.18 - /~joshi/courses/cecs352/handout.html} {.20 - /~joshi/courses/cecs352/outline.html} {.15 - /~joshi/courses/cecs438} {.18 - /~joshi/courses/cecs352/environment.html} {.16 - /~joshi/courses/cecs352/proj}
15	{1.0 - /~lan/cecs353} {.43 - /~lan/cecs353/assign1.html} {.25 - /~lan/cecs353/syl.html} {.25 - /~lan/cecs353/outline.html} {.75 - /~lan/cecs353/assign2.html} {.18 - /~lan}
16	{.15 - /faculty.html} {.17 - /people.html} {.15 - /people_index.html} {1.0 - /}

Table 2. Web session clusters.

i	$\|\mathcal{X}_i\|$	description	$\overline{D}_{Wi}$
1	114	main page, faculty list, individual faculty pages, research, people and class list	0.56
2	75	Dr. Jurzcyk's pages	0.15
3	32	access statistics pages	0.066
4	66	site manager's pages	0.049
5	207	general course inquiries	0.18
6	165	Dr Joshi's courses cecs352 and cecs438	0.18
7	64	Inquiries about undergraduate degrees and courses	0.28
8	67	accesses to administrator's pages	0.21
9	174	Accesses to the cecs227 class pages	0.11
10	72	Dr Shi's course cecs345 (references and projects)	0.22
11	88	Dr Joshi's research related pages	0.18
12	71	Dr Shi's course cecs345 (Lectures)	0.14
13	147	main page, class list, faculty list and people	0.21
14	114	mixture of unrelated accesses that don't make a strong profile	0.95
15	16	Dr Lan's course cecs353's pages	0.031
16	45	main page, faculty list and people	0.24
17	16	400-level courses' page	0.088
18	105	Dr Saab's course cecs333's pages	0.29
19	18	Dr Skubic's pages (courses, research, vitae ...)	0.14
20	48	Dr Saab's course cecs303's pages	0.13

distance of 0.88), and their high inter-cluster distances (the majority between .9 and 1). Note that all the remaining sessions that do not belong to any profile are lumped in the 14^{th} profile which is easily recognized using the quantitative evaluation measures. In fact, this particular cluster had no significant URLs ($P_{ij} < 0.15$ for all j) and its intra-cluster distance was very high (0.95), which is even higher than the total average pairwise distance of all sessions.

6. Conclusion

In this paper, we have presented a new approach for automatic discovery of user session profiles in Web log data. We defined the notion of a "user session" as being a temporally compact sequence of Web accesses by a user. A new similarity measure to analyze session profiles is presented which captures both the individual URLs in a Web session as well as the structure of the site. The Competitive Agglomeration clustering algorithm which can automatically determine the number of clusters was extended so that it can work on relational data. The resulting Competitive Agglomeration for Relational Data (CARD) algorithm can deal with complex and subjective dissimilarity/similarity measures which are not restricted to be Euclidean or metric. This algorithm was used to successfully cluster the sessions extracted from real server access logs into typical user session profiles. The resulting clusters are evaluated subjectively, as well as based on standard statistical criteria such as the intra-cluster and inter-cluster distances, or based on the significance of the components of a newly defined session "profile" vector which summarizes the typical sessions in each cluster.

Note that in some applications, the frequency of accesses to a Web page within the same session may be important. In that case, the definition of $s_j^{(k)}$ should be modified to $s_j^{(k)}$ = Number of times the j^{th} URL is accessed in the k^{th} session. In ongoing experiments, we are studying the effect of varying the parameters used in our approach on the resulting profiles. Particularly, different values of α are expected to result in different profile hierarchies corresponding to different resolution levels which may be varied as needed. We are also looking into robust profiling methods which have the advantage of being more resistant to noise, and a hierarchical profiling approach where clustering is applied recursively on the profiles found in previous runs. This approach has the potential of providing user session profiles that match any desired level of resolution or detail. We are extending this approach to incrementally update the profiles as more access data is logged by the server, instead of reapplying the entire Web mining process on the whole data. This will considerably reduce the time requirements of our approach.

Acknowledgements

This work was partially supported by two grants from the National Science Foundation (IIS 9800899 to Dr. Krishnapuram and IIS 9801711 to Dr. Joshi).

References

[1] Firefly, *http://www.firefly.com*

[2] A. Joshi, S. Weerawarana, and E. Houstis, *On disconnected browsing of distributed information*, Proc. Seventh IEEE Intl. Workshop on Research Issues in Data Engineering (RIDE), (1997) 101–108.

[3] L. Terveen, W. Hill, and B. Amento, *PHOAKS - A system for sharing recommendations*, Comm. ACM **40-3** (1997) 59–62.

[4] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell, *WebWatcher: A learning apprentice for the World Wide Web*, AAAI Spring Symposium on Information Gathering from Heterogenous, Distributed Environments, (March 1995).

[5] C. Shahabi, A. M. Zarkesh, J. Abidi and V. Shah, *Knowledge discovery from user's Web-page navigation*, Proc. Seventh IEEE Intl. Workshop on Research Issues in Data Engineering (RIDE), (1997) 20–29.

[6] B. Mobasher, N. Jain, E-H. Han, and J. Srivastava *Web mining: Pattern discovery from world wide Web transactions*, Technical Report 96-050, University of Minnesota, (Sep. 1996).

[7] R. Agrawal and R. Srikant, *Fast algorithms for mining association rules*, Proc. of the 20th VLDB Conference, Santiago, Chile (1994) 487–499.

[8] U. Fayad, G. Piatetsky-Shapiro, and P. Smyth *From data mining to knowledge discovery: An overview*, Advances in Knowledge Discovery and Data Mining, eds. U. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI/MIT Press (1996) 1–34.

[9] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York (1981).

[10] H. Frigui and R. Krishnapuram, *Clustering by competitive agglomeration*, Pattern Recognition **30-7** (1997) 1109–1119.

[11] R. Krishnapuram and J. Keller, *Fuzzy and Possibilistic Clustering Methods for Computer Vision*, Neural and Fuzzy Systems, eds. S. Mitra, M. Gupta and W. Kraske, SPIE Institute Series (1994) 133–159.

[12] G. S. Sebestyn and Roemder, *Decision-Making Process in Pattern Recognition*, Macmillan Company, New York (1962)

[13] E. E. Gustafson and W. C. Kessel, *Fuzzy clustering with a fuzzy covariance matrix*, Proc. IEEE CDC, San Diego, California (1979) 761–766.

[14] R. J. Hathaway, J. W. Davenport and J. C. Bezdek, *Relational duals of the c-means algorithms*, Pattern Recognition **22** (1989) 205–212.

[15] R. J. Hathaway and J. C. Bezdek, *NERF c-Means: Non-Euclidean relational fuzzy clustering*, Pattern Recognition **27-3** (1994) 429–437.

[16] L. Kaufman and P. J. Rousseeuw, *Finding groups in Data: Introduction to Cluster Analysis*, John Wiley, Brussels (1990).