Appeared in Proc. of 5th Annual World Conference on the WWW and Internet (Webnet 2000) , pp.464-469, San Antonio, Texas, Oct 30 - Nov 4, 2000.

Models for User Access Patterns on the Web: Semantic Content versus Access History

Arun Ross, Charles B. Owen, Aditya Vailaya
Department of Computer Science and Engineering,
Michigan State University, East Lansing, MI, USA.
{rossarun,cbowen,vailayaa}@cse.msu.edu

Abstract: This work focuses on clustering a site into groups of documents that are predictive of future user accesses. Two approaches have been developed and tested. The first approach uses semantic information inherent in the documents to facilitate the clustering process. User access history is then used to reorganize the clusters iteratively so as to better indicate access patterns. This method was found to not be an effective solution to the problem. Hence, a second method based on hierarchical clustering of trail information was developed. This method is shown to be far more effective than the first method.

1 Introduction

With the rapid proliferation of websites on the Internet over the past few years, it has become imperative for websites to enhance the quality of service that they provide in order to attract and sustain user traffic. The average user is interested only in a limited subset of the available content at a website. The emphasis therefore should be on developing tools that aid the user select that subset (automatic customization of hyperlink presentation order, for example). Such a strategy warrants predicting a user's actions based on past user-activity at the website.

One way to facilitate prediction would be to develop a model for user access patterns. The assumption is that patterns exist in aggregate user access histories that allow the behavior of one user to be predicted based on the behavior of previous users. The first step towards modeling user access patterns is modeling the site. Site modeling involves organizing and grouping the pages (or documents) present in the site. A variety of criterion can be used, atleast in theory, to group the documents available on the web server. These criterion can be placed under two broad categories: (a) organization based on the content of documents ([Green, 1998, Weiss et al., 1996, Fowler et al., 1996]) and (b) organization based on the access history of documents ([Joshi and Krishnapuram, 1998, Perkowitz and Etzioni, 1997, Mobasher et al., 1996]). In this paper we describe two approaches that we have developed to induce clustering of documents.

2 The World Cup 1998 Server Log Data

The test data for this research consists of 14 weeks of server logs from the 1998 World Cup Soccer site (http://www.france98.org). The data was collected during the period of the 1998 World Cup games. The server logs (the user access history for 14 weeks) were provided by Hewlett-Packard Labs ([Arlitt and Jin, 1999]). The following table lists some basic statistics about this data set.

Number of weeks	14
Total page requests	1, 350, 004, 229
Number of distinct IP addresses	2, 769, 788
HTML requests	38.59%
Image requests	35.03%
Other requests (audio, video, etc.)	26.38%

The site is structured as a bilingual English/French site. We have considered the English content only. The trail data is very large, so we have focused on two representative weeks for our initial tests - weeks 3 and 7, a medium and a large traffic week.

3 Problem Definition

Let $D = \{d_1, d_2, \ldots, d_N\}$ refer to the set of N HTML documents present in the server. The user access history can then be expressed as, $L = \{< m_1, t_1, d_1 >, < m_2, t_2, d_2 >, \ldots, < m_M, t_M, d_M >\}$, where m_j indicates the source of the request (an IP address or a machine identifier), t_j is the server access time, and d_j is the requested document. Processing this information results in a set $S = \{S_1, S_2, \ldots, S_i\}$ of user sessions. Each session $S_j = \{d_{j1}, d_{j2}, \ldots d_{jk}\}$ indicates the order in which documents were requested by a single machine.

The goal in this work is to identify K document clusters such that user sessions span the minimum number of clusters. Thus we have to identify K clusters, $C = \{c_1, c_2, \ldots, c_K\}$, with each $c_i \subseteq D$, such that the number of inter-cluster transitions (ICT), normalized with respect to the total number of transitions, is minimized. A transition is simply two sequential accesses in a user session. The inter-cluster transition criteria is defined in Eq. 1.

$$ICT(K) = \frac{\sum_{j=1}^{|S|} |\{< d_{ji}, d_{j,i+1} > \in T_j : d_{ji} \in c_p, d_{j,i+1} \in c_q, c_p \neq c_q\}|}{\sum_{j=1}^{|S|} |S_j| - 1} \quad (1)$$

Here T_j represents all transitions in session S_j. The choice of K does, of course, have a significant influence on this formulation and has a major affect on the performance of any system that utilizes clustering for predictive purposes. In our current formulation, we have chosen K empirically. Even for a fixed value of K, the problem of finding the optimal value of $ICT(K)$ is practically intractable [Jain and Dubes, 1988]. Thus, we apply heuristics to perform the above clustering.

4 Method 1: Clustering Using Semantic and Trail Information

In addition to the user trail information, document content is also indicative of relationships among site pages. Consequently, we chose to initiate a study of the relationship of semantic clustering and trail clustering. Semantic clustering can be used to "bootstrap" access pattern clustering by providing an initial grouping of pages that can then be iteratively reclustered with increasing influence by the trail information. Also, semantic clustering provides a mechanism for insertion of new documents into document sets prior to the availability of trail information (document routing - [Hull et al., 1996]).

The documents at the server site are first clustered (K clusters) according to their semantic content (words in the documents). This semantic clustering consists of the following two steps: (1) *Word Clustering*: This involves extracting unique words from all documents and using them as features. Since the number of words in a document and hence, the entire set of documents at the server can become huge, we reduce the number of features. A number of methods have been described in the literature for reducing the dimensionality of the feature space (such as singular value decomposition (SVD) [Deerwester et al., 1990] and feature clustering [Wulfekuhler and Punch, 1997]). We use the feature clustering method, since it is fast and has been shown to be an efficient dimensionality reduction method. (2) *Document Clustering*: The N documents are then partitioned into K clusters using the new, reduced, feature set.

4.1 Word Clustering

Let $W = \{w_1, w_2, ..., w_M\}$ be the set of unique words (after stemming the words, removing stop words and combining words that always occur together) extracted from the N documents. The pattern matrix (with the words acting as features) for the N documents can be represented as $F_D = [f(d_1)\, f(d_2) \ldots f(d_N)]^t$, where $f(d_i) = [b_1^i\, b_2^i \ldots b_M^i]^t$ and $b_j^i = 1$, if the word w_j occurs in d_i and $b_j^i = 0$, otherwise. F_D is an $N \times M$ matrix. Since the value of M can be very large, we first cluster the words into a smaller subset prior to clustering the documents. In order to do so, we cluster the inverted pattern matrix (F_D^t, where each row now represents a word (and the N columns represent the documents) into M' clusters ($M' << M$). In our current implementation, M' is chosen empirically (500 in early experiments). We employ the K-Means clustering algorithm, with a normalized cosine-measure as the dissimilarity measure between two feature vectors. The feature clustering yields a new set of words, $W' = \{w'_1, w'_2, ..., w'_{M'}\}$, where $|W'| = M'$. The new pattern matrix (for the N documents) is defined as $F'_D = [f'(d_1)\, f'(d_2) \ldots f'(d_N)]^t$, where $F'_D(i,j) = 1$ if any of the words assigned to cluster W'_j occur in D_i.

4.2 Document Clustering

Having clustered the words, we next cluster the N documents into M' clusters based on the reduced dimensionality pattern matrix, F'_D. We again apply the K-Means clustering algorithm, with the normalized cosine-measure as the dissimilarity measure between two feature vectors, as the clustering algorithm.

4.3 Using User Access Information

With time, server logs with user requests become available to the server. These are then used to refine the semantic clustering performed above.

Define a new pattern matrix, $H_D = [\omega_F^t F'_D \;\; \omega_G^t G_D]_{N \times (M'+K)}$, where

$$G_D(i,j) = \frac{\sum_{k=1}^{|S|} \left[\sum_{d_m \in c_j} l_k(d_i, d_m) + I(d_i \in c_j) \sum_{m=1}^{N} l_k(d_m, d_i) \right]}{\sum_{k=1}^{|S|} \left[\sum_{m=1}^{N} l_k(d_i, d_m) + \sum_{m=1}^{N} l_k(d_m, d_i) \right]}, \; 1 \le i \le N, \; 1 \le j \le K,$$

and

$$\omega_F(i) = \sqrt{1 - \alpha^2(t)}, \; 1 \le i \le N,$$
$$\omega_G(i) = \alpha(t), \; 1 \le i \le N,$$

$l_k(d_l, d_m)$ is the number of transitions from d_l to d_m in user session k. $\alpha(t)$ is a parameter which monotonically increases with time, t. As more and more server log data become available, $\alpha(t)$ is increased to give a higher weight to user access information and a smaller weight to the semantic content in the HTML documents. Our current implementation does not change the value of $\alpha(t)$ with time, since the server and document set are fixed (we are post-processing historical data). We have empirically chosen $\alpha(t) = 0.7$. $G_D(i,j)$ is the normalized count of the number of transitions from document d_i to documents in cluster c_j and all transitions into d_i if $d_i \in c_j$. The new pattern matrix is then subjected to the K-means clustering algorithm in order to derive a fresh set of clusters. This process is repeated in an iterative fashion.

It is expected that the incremental inclusion of user session information will improve the predictive performance. As discussed in the next section, this does turn out to be true, though not to the extent expected.

4.4 Experimental Results

The first step in the algorithm is the clustering of words into reduced feature vectors. Here are some statistics relevant to the word clustering process:

Number of words after stemming and removing stop words	19,227
Number of words after removing words occurring in single documents	13,105
Number of words after combining words always occurring together	11,002

Fig. 1(a) illustrates the result of the word clustering step in the algorithm. The vast majority of clusters have word counts of around 20 words as expected, given the partitioning of 11,002 words into 500 clusters.

While some clusters were compact (the intra-cluster error was less) other clusters had large intra-cluster errors. Members that are at distances greater than a chosen threshold could be treated as outliers and reclustered separately. As future work, we plan on performing these experiments.

The next step of the algorithm performs document clustering using the reduced-size word feature vectors. This is semantic clustering only. The number of unique HTML English documents considered was 5,841 and the value of K was set to 500. The results of this process are summarized below:

WEEKS →	**Week 3**	**Week 7**
Number of transitions	852,125	14,892,627
Penalty	761,069	14,227,220
$ICT(500)_{SM}$	0.89	0.95

The term *penalty* refers to the number of out-of-cluster transitions. These results indicate that semantic clusters do not appear to be effective predictors of user trails. In week 3 for example, only 11% of all transitions are to pages that are semantically related. Most of the clusters, as seen in the histogram in Fig. 1(b), are reasonably sized.

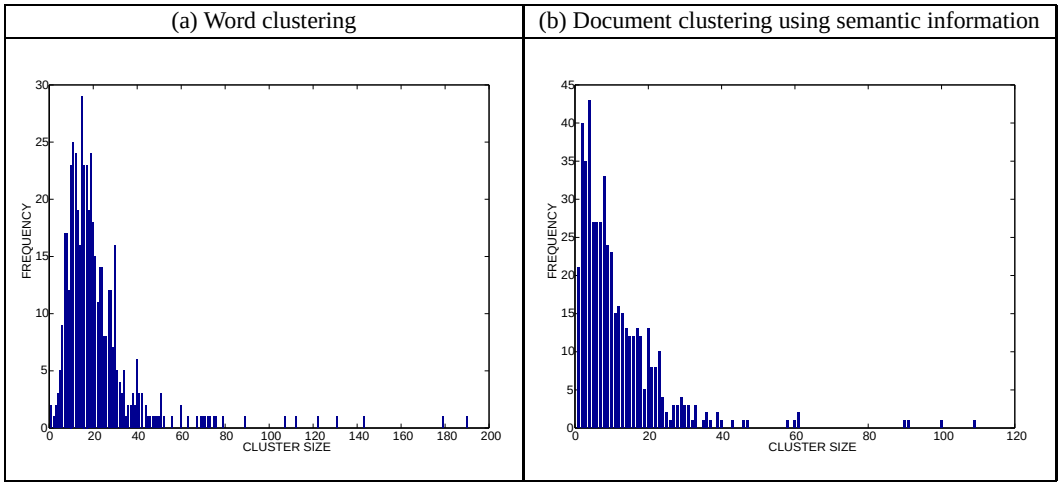

Figure 1: Histogram of cluster sizes

The final step in the process involves the application of the user session information. This is an iterative process, in that the clustering is based on the participation of documents in initial clusters. If the algorithm performs properly, we expect to see decreasing ICT values for each iteration. The algorithm was applied to the test data. The results are tabulated below:

WEEKS →	**Week 3**	**Week 7**
ITERATION 1	Penalty : 613, 490, $ICT(500)_{SL}$ = 0.720	Penalty : 14, 119, 968, $ICT(500)_{SL}$ = 0.948
ITERATION 2	Penalty : 598, 703, $ICT(500)_{SL}$ = 0.700	Penalty : 13, 873, 159, $ICT(500)_{SL}$ = 0.931
ITERATION 3	Penalty : 590, 950, $ICT(500)_{SL}$ = 0.693	Penalty : 13, 774, 633, $ICT(500)_{SL}$ = 0.925
ITERATION 4	Penalty : 588, 814, $ICT(500)_{SL}$ = 0.691	Penalty : 13, 861, 116, $ICT(500)_{SL}$ = 0.931

As seen above, repeated iterations do not considerably reduce the penalty and the predictive power of the method does not seem to be that useful.

5 Method 2: Hierarchical Trail Clustering

Method 1 proved to be complex computationally due to the iterative application of the K-means algorithm and the method produced rather disappointing results. Thus, we used what we had learned about the structure of the data to develop another method that does not use the semantic information contained in documents for site clustering. Instead, it uses only the trail information. This method uses the hierarchical clustering technique on a proximity matrix generated from the user trail information. In order to perform hierarchical clustering, a similarity (or dissimilarity) measure between documents must be defined. A crucial part of the hierarchical method involves updating the similarity measures after combining two documents or two clusters. For this application we chose the single-link technique which, unlike other techniques available, helps combine nodes that occur in a trail. The clustering routine generates a dendogram, which indicates the clusters and their components at various thresholds ([Jain and Dubes, 1988]).

5.1 Similarity Measure

In order to compute a similarity measure, the transition matrix was used. Each entry in the transition matrix indicates the number of transitions between pairs of documents, i.e., entry $t_{i,j}$, in the $N \times N$ transition matrix $T = \{t_{i,j}\}$, denotes the total number of transitions from document i to document j as observed from the access log. Clearly, the matrix T need not be symmetrical. However, it is an indicator of the pair-wise similarity between documents. The transition matrix is used to generate the similarity matrix $S = \{s_{i,j}\}$, which is a $N \times N$ symmetrical matrix with entry $s_{i,j}$ indicating the similarity value between documents i and j. We present below the technique that was used to transform T into S.

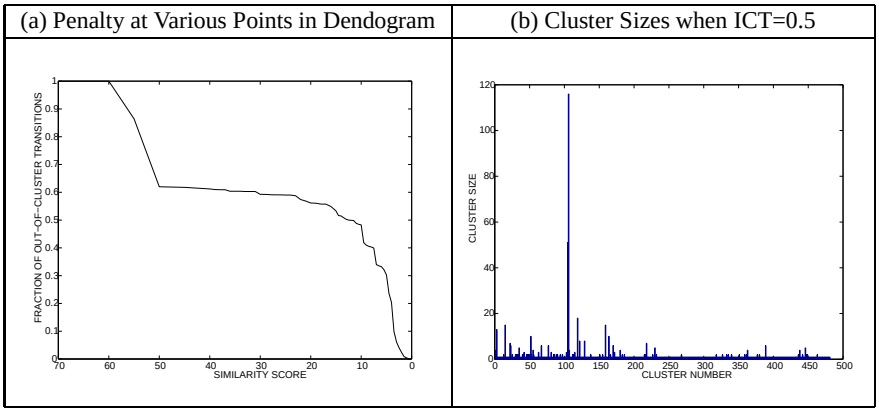

Figure 2: Performance of the Hierarchical Clustering Technique

5.1.1 *Preprocessing I*

In order to offset the noise introduced by long trails (effect of proxy servers and/or web spiders), trails of size greater than 40 were eliminated. After preprocessing, only 866 of the 5841 documents had at least a single non-zero entry in the week 3 transition matrix. Thus the first set of experiments operated on a 866×866 transition matrix representing $759,640$ transitions between the documents.

To compute the similarity matrix S, the matrices T and T' were used - T' being the transpose of T. While each entry $t_{i,j}$ in T denotes the number of transitions *from* document i to document j , each entry $t'_{i,j}$ in T' denotes the number of transitions *to* document i from document j. Clearly, these values are identical prior to normalization.

1. Step 1: As a first step, all rows of T and T' are normalized as follows:
   $t_{1i,j} = \frac{t_{i,j}}{\sum_{k=1}^{N} t_{i,k}}, t_{2i,j} = \frac{t'_{i,j}}{\sum_{k=1}^{N} t'_{i,k}}$
2. Step 2: The similarity matrix is computed as follows:
   $s_{i,j} = \max\{\min\{t_{1i,j}, t_{2j,i}\}, \min\{t_{1j,i}, t_{2i,j}\}\}$

5.1.2 *Preprocessing II*

One would intuitively expect the method described in the previous section to result in reasonably sized clusters. However, results indicate (see Fig. 2(b)) that the single link technique operating on the similarity matrix constructed by this method has the tendency to form a single large cluster. In order to find out the reason for this anomaly the transition matrix was examined in greater detail. It was observed that only 287 of the 866 documents were involved in at least 100 transitions. Thus we used the following heuristic to prune the size of the transition matrix and alter it's entries: only those documents that were involved in at least a 100 different transitions (either into them or out of them) were considered. The transition matrix eventually considered 287 documents and had a total of $749,270$ transitions.

This method gave the best clustering results. Fig. 3(a) shows the fraction of out-of-cluster transitions as the number of clusters is varied by cutting the dendogram at various points. Fig. 3(b) shows the number of documents in each cluster when the fraction of out-of-cluster transitions is approximately 0.3. An ICT value of 0.3 indicates that 70% of all transistions were predicted by the clustering. In the figure it can be seen that 70% prediction is accomplished with less than 120 clusters, with a maximum cluster size of 32. Average cluster size is much smaller. There is a tendency for the clustering to emit a larger number of small clusters. We are examining ways to recluster this emitted set in order to form more uniform groupings of the site content.

6 Conclusion and Future Work

The result summarized above needs to be verified on the entire 14 weeks of the data set. We are examining this in several ways. Currently we are using the entire data set for both training and testing. We plan to partition that data set into training and testing to verify that this method is not resulting in over-fitting to the data.

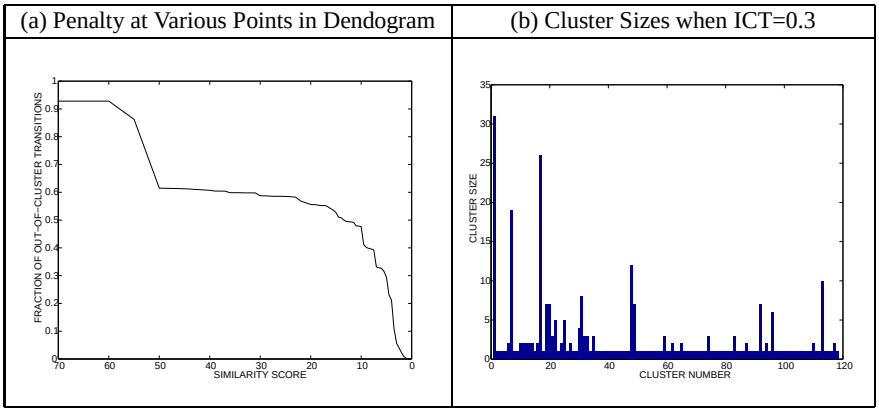

Figure 3: Performance of the Hierarchical Clustering Technique after considering only those documents involved in atleast a 100 transitions

The clustering method presented in this paper necessarily produces non-overlapping clusters. While this is considered to be effective in many cases, there may be cases where common nodes exist in trails. This is particularly true if central index nodes are expected to be elements of the user trails. We plan to incorporate a technique that would facilitate generation of overlapping clusters.

The apparent ineffectiveness of semantic methods does leave the document routing problem as an open issue in this work. If new documents cannot be selected by semantic characteristics, it is not certain how they can be routed to initial clusters. As presented, the techniques in this report will ignore new documents until sufficient user history has been developed.

References

[Arlitt and Jin, 1999] Arlitt, M. and Jin, T. (1999). Workload characterization of the 1998 world cup web site. Report HPL-1999-35, IRCS.

[Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

[Fowler et al., 1996] Fowler, R. H., Kumar, A., and Williams, J. (1996). Visualizing and browsing www semantic content. In *Proceedings of the First Annual IEEE/ACM Conference on Emerging Technologies and Applications in Communication*.

[Green, 1998] Green, S. J. (1998). Automated link generation: can we do better than term repetition? In *Seventh International World Wide Web Conference*, Brisbane, Australia.

[Hull et al., 1996] Hull, D., Pedersen, J., and Schuetze, H. (1996). Document routing as statistical classification. In *Proceedings of AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA.

[Jain and Dubes, 1988] Jain, A. K. and Dubes, R. C. (1988). *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, New Jersey.

[Joshi and Krishnapuram, 1998] Joshi, A. and Krishnapuram, R. (1998). Robust fuzzy clustering methods to support web mining. In *Proc. ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*.

[Mobasher et al., 1996] Mobasher, B., Jain, N., Han, E.-H. S., and Srivastava, J. (1996). Web mining: Pattern discovery from world wide web transactions. Report TR-96050, Dept. of Computer Science, University of Minnesota.

[Perkowitz and Etzioni, 1997] Perkowitz, M. and Etzioni, O. (1997). Adaptive web sites: Automatically learning from user access patterns. In *Sixth International WWW Conference*, Santa Clara, CA, USA.

[Weiss et al., 1996] Weiss, R. et al. (1996). Hypursuit: A hierarchical network search engine that exploits content-link hypertext clustering. In *Hypertext'96: The Seventh ACM Conference on Hypertext*, Washington D.C., USA.

[Wulfekuhler and Punch, 1997] Wulfekuhler, M. R. and Punch, W. (1997). Finding salient features for personal web page categories. In *6th International World Wide Web Conference*.