A User Behavior Model for Web Page Navigation

Şule Gündüz[*]and M. Tamer Özsu [†]

October 2002

[*]on leaving from Department of Computer Science, Istanbul Technical University, Istanbul, Turkey.
[†]School Of Computer Science, University of Waterloo, Waterloo, ON

Abstract

Making recommendation requires predicting what is of interest to a user at a specific time. Even the same user may have different desires at different times. It is important to extract the aggregate interest of a user from his or her navigational path through the site in a session. This paper concentrates on the discovery and modelling of the user's aggregate interest in a session. This approach relies on the premise that the visiting time of a page is an indicator of the user's interest in that page. The proportion of times spent in a set of pages requested by the user within a single session forms the aggregate interest of that user in that session. We first partition user sessions into clusters such that only sessions which represent similar aggregate interest of users are placed in the same cluster. We employ a model-based clustering approach and partition user sessions according to similar amount of time in similar pages. In particular, we cluster sessions by learning a mixture of Poisson models using Expectation Maximization algorithm. The resulting clusters are then used to recommend pages to a user that are most likely contain the information which is of interest to that user at that time. Although the approach does not use the sequential patterns of transactions, experimental evaluation shows that the approach is quite effective in capturing a Web user's access pattern. The model has an advantage over previous proposals in terms of speed and memory usage.

1 Introduction

Web mining is defined as the use of data mining techniques to automatically discover and extract information from Web documents and services [11]. With the rapid growth of the World Wide Web, the study of modelling and predicting a user's access on a Web site has become more important. It has been used to improve the Web performance through caching [1, 29] and prefetching [16, 26] , to recommend related pages [25], improve search engines [7] and personalize browsing in a Web site [16]. Given a user's (who may, for example, be a customer in an e-commerce site) current actions, the goal is to determine which Web pages (items) will be accessed (bought) in the near future.

There are three steps in this process [32]. Since the data source is Web server log data for Web usage mining, the first step is to clean the data and prepare for mining the usage patterns. The second step is to extract usage patterns, and the third step is to build a predictive model based on the extracted usage patterns. Fundamental methods of data cleaning and preparation are given in [8, 34]. The main techniques traditionally used for modelling usage patterns in a Web site are collaborative filtering (CF) [6], clustering pages or user sessions [13, 23], association rule generation [30], sequential pattern generation [2, 31] and Markov models [3, 10, 26]. The prediction step is the real-time processing of the model, which considers the active user session and makes recommendations based on the discovered patterns.

However, the discovery of usage patterns discussed above is not sufficient to accurately describe the user's navigational behavior in a *server session*[1]. An important feature of the user's navigation path is the time that a user spends on different pages [28]. Even the same person

[1]The term *server session* is defined as the click stream of page views for a single visit of a user to a Web site [8]. In this paper we will use this term interchangeably with "user session" and "user transaction".

may have different desires at different times. If we knew the desire of a user every time he or she visits the Web site, we could use this information for recommending pages. Unfortunately, experience shows that users are rarely willing to give explicit feedback. Thus, the time spent on a page is a good measure of the user's interest in that page, providing an implicit rating for that page. If a user is interested in the content of a page, he or she will likely spend more time there compared to the other pages in his or her session.

The most commonly used techniques to predict the user's next request are sequential patterns, association rules and Markov models. These techniques work well for Web sites that do not have a complex structure, but experiments on complex, highly interconnected sites show that the storage space and runtime requirements of these techniques increase due to the large number of patterns for sequential pattern and association rules, and the large number of states for Markov models.

In this paper, we present a new model that uses only the visiting time and visiting frequencies of pages without considering the access order of page requests in user sessions. The resulting model has lower run-time computation and memory requirements, while providing predictions that are at least as precise as previous proposals. Our objective in this paper is to assess the effectiveness of *non-sequentially ordered* pages in predicting navigation patterns. To capture the relationships between visited pages in one session we use frequent item sets extracted from the Web log data. The key idea behind this work is that user sessions can be clustered according to the similar amount of time that is spent on similar pages within a session. In particular, we model user sessions in log data as being generated in the following manner: (i) When a user arrives to the Web site, his or her current session is assigned to one of the clusters, (ii) the behavior of that user in this session, in terms of visiting time, is then generated from a Poisson model of visiting times of that cluster. Since we do not have the actual cluster assignments, we use a standard learning algorithm, the Expectation-Maximization (EM) [9], to learn the cluster assignments of transactions as well as the parameters of each Poisson distribution. The resulting clusters consist of transactions in which users have similar interests and each cluster has its own parameters representing these interests.

The next page request of an active user is predicted using parameters of the cluster to which the active user is assigned. A performance analysis of the model is conducted using a new approach to calculate a recommendation score for the next request of an active user session. The experimental results show that with proper preprocessing, our model yields a good prediction accuracy. Beside this, the results are robust across sites with different structures.

The rest of the paper is organized as follows. Section 2 briefly reviews the work related to Web usage mining and describes equations for training a mixture model with EM algorithm. Section 3 presents the proposed model. Section 4 provides detailed experimental results. Finally, in Section 5 we conclude and discuss future work.

2 Background

2.1 Web Usage Mining

In general, Web mining is a common term for three knowledge discovery domains that are concerned with mining different parts of the Web: Web Content Mining, Web Structure Mining, and Web Usage Mining [5, 20]. While Web content and structure mining utilize real or primary data on the Web, Web usage mining works on the secondary data such as Web server access logs, proxy server logs, browser logs, user profiles, registration data, user sessions or transactions, cookies, user queries, and bookmark data. Web usage mining refers to the application of data mining techniques to discover usage patterns from these secondary data, in order to understand and better serve the needs of Web-based applications. The usage data collected at different sources will represent the navigation patterns of different segments of the overall Web traffic, ranging from single-user, single-site browsing behavior to multi-user, multi-site access patterns. The information provided by the data sources can all be used to construct/identify several data abstractions, such as users, server sessions, episodes, click stream, and page views [17].

Collaborative filtering techniques work by grouping users such that each group has similar patterns [6]. On the other hand, page recommendations in [21, 22] are based on clusters of pages found from the server log for a site. The system recommends pages from clusters that most closely match the current session. The system described in [23] clusters user sessions using a fuzzy clustering algorithm and allows a page or user to be assigned to more than one cluster.

Recently, some authors have used association rules [30] and sequential patterns [2, 31] in recommender systems [27]. These approaches suffer from the problem of generating a large number of rules which makes on-line recommendation inefficient. It may be possible to prune the rule space, enabling faster on-line prediction [33].

2.2 Frequent Pattern Mining

One of the data mining tasks is the discovery of frequent patterns in the data set. The frequent pattern mining can be formally stated as follows: Let $I = \{i_1, i_2, ..., i_n\}$ be a set of page items, and $S = \{T_1, T_2, ..., T_k\}$ be a transaction set, where T_i ($i \in [1...k]$) be a transaction that contains a subset of items in I. The *support* (or *absolute occurrence frequency*) of a pattern A, which is a subset of items, is the number of transactions in S containing A. A is a frequent pattern if A's support is no less than a predefined minimum support threshold ξ [14].

In our study, we use frequent item sets extracted from Web log data only for reducing the dimensionality of the input data. Some of the page views appear in less than 1% of transactions in the entire data set if the Web site has a complex structure. A learning algorithm for predicting the next request of the user will learn not to recommend the pages with a low frequency of request. Thus, reducing the dimensionality of the input data by removing less frequent page requests at the beginning of the learning algorithm makes it efficient. On the other hand, using frequent patterns as a filter for eliminating pages covers simple non-personalized recommendation such that: "users who visit page A also visit page B".

2.3 Mixture Models for Clustering

In this section, we first describe the mixture model for clustering objects and then describe how the parameters of the clusters are derived in the context of the mixture model.

2.3.1 Model-Based Cluster Analysis

The process of grouping a set of physical or abstract objects into classes of similar objects is called *clustering.* A *cluster* is a collection of data objects that are similar to one another within the same cluster and dissimilar to objects in other clusters [12]. Clustering methods range from those that are largely heuristic to more formal procedures based on statistical models.

Model-based clustering methods optimize the fit between the given data and some mathematical model. Such methods are often based on the assumption that the data are generated by a mixture of underlying probability distributions [15]. Given a data set of K observations $\mathbf{D} = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$, every observation $\mathbf{x}_i, (i \in [1, ..., K])$ is generated according to a probability distribution defined by a set of parameters, denoted $\mathbf{\Theta}$. The probability distribution consists of a mixture model of components $c_j \in C = \{c_1, c_2, ..., c_G\}$. The parameters of each component, $\mathbf{\Theta}_g$, is a disjoint subset of $\mathbf{\Theta}$, where $\mathbf{\Theta}_g$ $(g \in [1...G])$ is a vector specifying the probability distribution function (pdf) of the g^{th} component. An observation, $\mathbf{x}_i$, is created by first selecting a mixture component according to the mixture weights (or cluster prior probabilities), $p(c_g|\mathbf{\Theta}) = \tau_g$, where $\sum_{g=1}^{G} \tau_g = 1$, then having this selected mixture component generates an observation according to its own parameters, with distribution $p(\mathbf{x}_i|c_g; \mathbf{\Theta}_g)$. Thus, the likelihood of a data point, $\mathbf{x}_i$, can be characterized with a sum of total probabilities over all mixture components:

$$p(\mathbf{x}_i|\mathbf{\Theta}) = \sum_{g=1}^{G} p(c_g|\mathbf{\Theta}) p(\mathbf{x}_i|c_g, \mathbf{\Theta}_g) = \sum_{g=1}^{G} \tau_g p(\mathbf{x}_i|c_g, \mathbf{\Theta}_g) \tag{1}$$

Statisticians refer to such a model as *mixture model with G components.* Thus, the model-based clustering problem consists of finding the model, i.e. the model structure and parameters for that structure that best fit the data. The parameters are chosen in two ways. The *maximum likelihood* (*ML estimation*) approach maximizes:

$$\ell_{ML}(\mathbf{\Theta}_1, ..., \mathbf{\Theta}_G; \tau_1, ..., \tau_G|D) = \prod_{i=1}^{K} \sum_{g=1}^{G} \tau_g p(\mathbf{x}_i|c_g, \mathbf{\Theta}_g) \tag{2}$$

The second approach maximizes the *posterior probability* (*MAP estimation*) of $\mathbf{\Theta}$ given the data:

$$\ell_{MAP}(\mathbf{\Theta}_1, ..., \mathbf{\Theta}_G; \tau_1, ..., \tau_G|D) = \prod_{i=1}^{K} \sum_{g=1}^{G} \frac{\tau_g p(\mathbf{x}_i|c_g, \mathbf{\Theta}_g) p(\mathbf{\Theta})}{p(D)} \tag{3}$$

The term $p(D)$ can be ignored in Equation (3), since it is not a function of $\mathbf{\Theta}$.

In our study, we will use the ML estimate, since we do not have any prior knowledge about the model parameters. In practice, the *log* of these expressions is often used. Thus, the *log* likelihood of Equation (2) is:

$$L(\mathbf{\Theta}_1, ..., \mathbf{\Theta}_G; \tau_1, ..., \tau_G | D) = \sum_{i=1}^{K} \ln \left(\sum_{g=1}^{G} \tau_g p(\mathbf{x}_i | c_g, \mathbf{\Theta}_g) \right) \tag{4}$$

The set of parameters of the model ($\mathbf{\Theta}$) include mixture weights representing cluster prior probabilities (τ_g), which indicate the probability of selecting different mixture components and the set of the parameters of the probability distribution assumed for the data:

$$\mathbf{\Theta} = \{\mathbf{\Theta}_1, ..., \mathbf{\Theta}_G, \tau_1, ..., \tau_G\}, \quad \sum_{g=1}^{G} \tau_g = 1 \tag{5}$$

2.4 EM Algorithm for Clustering

The model parameters can be trained using the *Expectation Maximization* (EM) algorithm. The EM algorithm is a very general iterative algorithm for parameter estimation by maximum likelihood when some of the random variables involved are not observed (i.e., considered missing or incomplete). In the expectation step (E-step), the values of the unobserved variables are essentially “filled in”, where the filling-in is achieved by calculating the probability of the missing variables, given the observed variables and the current values of parameters. In the maximization step (M-step), the parameters are adjusted based on the filled-in variables [15].

Let $D = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$ be a set of K observed variables, and $H = \{\mathbf{z}_1, ..., \mathbf{z}_K\}$ represent a set of K values of hidden variables Z, such that each $\mathbf{z}_i$ is in the form of $\mathbf{z}_i = \{z_{1i}, ..., z_{Gi}\}$ and corresponds to a data point $\mathbf{x}_i$. It can be assumed that Z is discrete and represents the class (or cluster) labels for the data with the following possible values:

$$z_{ji} = \begin{cases} 1 & \text{if} \quad \mathbf{x}_i \text{ belongs to cluster } j; \\ 0 & \qquad \text{otherwise.} \end{cases}$$

If Z could be observed, then the ML estimation problem would be based on the maximization of the quantity:

$$\mathrm{L}_c(\mathbf{\Theta}; D, H) \triangleq \ln p(D, H | \mathbf{\Theta}) \tag{6}$$

In the presence of missing data, we calculate conditional expectation of the complete data likelihood given the observed data and the current parameter estimate as follows:

$$Q(\mathbf{\Theta}, \mathbf{\Theta}') = E\left[L_c(D, H | \mathbf{\Theta}) | D, \mathbf{\Theta}'\right] \tag{7}$$

where the term $L_c(D, H|\mathbf{\Theta})$ is:

$$L_c(D, H | \mathbf{\Theta}) = \sum_{i=1}^{K} \ln p(\mathbf{x}_i, \mathbf{z}_i | \mathbf{\Theta}) \tag{8}$$

Equation (7) involves $\mathbf{\Theta}$, which is the parameter of the complete likelihood and $\mathbf{\Theta}'$, which is the parameter of the conditional distribution of complete data.

The Q-function in Equation (7) can be expanded as follows:

$$\begin{aligned}
E\left[L_c(D,H|\mathbf{\Theta})|D,\mathbf{\Theta}'\right] &= E\left[\sum_{i=1}^{K}\ln p(\mathbf{x}_i,\mathbf{z}_i|\mathbf{\Theta})|D,\mathbf{\Theta}'\right] \\
&= \sum_{l=1}^{G}\sum_{i=1}^{K}\ln p(\mathbf{x}_i,\mathbf{z}_i|\mathbf{\Theta})\prod_{j=1}^{K}p(z_{lj}|\mathbf{x}_j,\mathbf{\Theta}') \\
&= \sum_{i=1}^{K}\sum_{l=1}^{G}\left(\ln p(\mathbf{x}_i,\mathbf{z}_i|\mathbf{\Theta})p(z_{li}|\mathbf{x}_i,\mathbf{\Theta}')\right)\prod_{j\neq i}\sum_{l=1}^{G}p(z_{lj}|\mathbf{x}_j,\mathbf{\Theta}') \\
&= \sum_{i=1}^{K}\sum_{l=1}^{G}\ln p(\mathbf{x}_i,\mathbf{z}_i|\mathbf{\Theta})p(z_{li}|\mathbf{x}_i,\mathbf{\Theta}') \\
&= \sum_{i=1}^{K}\sum_{\mathbf{z}_i}\ln p(\mathbf{x}_i,\mathbf{z}_i|\mathbf{\Theta})p(\mathbf{z}_i|\mathbf{x}_i,\mathbf{\Theta}') \\
&= \sum_{i=1}^{K}\sum_{\mathbf{z}_i}p(\mathbf{z}_i|\mathbf{x}_i,\mathbf{\Theta}')\ln\left[p(\mathbf{x}_i|\mathbf{z}_i,\mathbf{\Theta})p(\mathbf{z}_i|\mathbf{\Theta})\right] \\
&= \sum_{i=1}^{K}\sum_{\mathbf{z}_i}p(\mathbf{z}_i|\mathbf{x}_i,\mathbf{\Theta}')\left[\ln p(\mathbf{x}_i|\mathbf{z}_i,\mathbf{\Theta})+\ln p(\mathbf{z}_i|\mathbf{\Theta})\right] \qquad (9)
\end{aligned}$$

At each EM iteration the Q-function is maximized with respect to the parameters $\mathbf{\Theta}$ using the current parameters $\mathbf{\Theta}'$. At the end of each iteration, a set of new optimal parameters $\mathbf{\Theta}$ becomes the current parameters $\mathbf{\Theta}'$ for the next iteration. Given these steps, the EM algorithm can be implemented as follows:

1. Choose an initial estimate for parameter set $\mathbf{\Theta}'(0)$, and set $n = 0$.
2. **(E)xpectation Step:** For n, compute $Q(\mathbf{\Theta}, \mathbf{\Theta}'(n))$ using Equation (7).
3. **(M)aximization Step:** Replace the current estimate $\mathbf{\Theta}'(n)$ with the new estimate $\mathbf{\Theta}'(n+1)$ where,
$$\mathbf{\Theta}'(n+1) = argmax_{\mathbf{\Theta}}Q(\mathbf{\Theta}, \mathbf{\Theta}'(n))$$
4. Set $n = n + 1$ and iterate steps 2 and 3 until convergence.

By iteratively applying the E-step and M-step, the parameters $\mathbf{\Theta}$ will converge to at least a local maximum of the *log* likelihood function. There are several reasonable choices for a convergence criterion.

3 Web Page Recommendation Model

This section presents the proposed model. As discussed in the introduction, Web usage mining consists of three steps. For the first step, we use cleaning and filtering methods in order to identify unique users and user sessions. In the second step, we cluster the transactions in the training set according to the similar amount of time in similar pages using model-based clustering. The model parameters are learned with EM algorithm under the assumption that the data come from a mixture of Poisson distributions. Using these model parameters cluster profiles are built for every cluster. For the last step, the transactions in the test set are assigned to one of the clusters that has the highest probability given the visiting time of current transaction's active page. The recommendation engine then predicts the current transaction's next page by ranking the recommendation scores calculated for each page in the most similar cluster.

3.1 Data Preparation and Cleaning

In this research, we use three sets of server logs. The first one is from the NASA Kennedy Space Center server over the months of July and August 1995 [19]. The second log is from ClarkNet Web server which is a full Internet access provider for the Metro Baltimore-Washington DC area [18]. This server log was collected over the months of August and September, 1995. The last server log is from the Web server at the University of Saskatchewan from June to December, 1995 [24]. These are well-known data sets that have been used in other studies. For each log data set we apply the same pre-processing steps[2]. First, all entries from the server log files are stored in a relational database. Next, the log entries are converted into a set of user sessions as follows: the irrelevant log entries are eliminated such that only URL page requests of the form "GET ...html" are maintained. The *visiting page time*, which we define as the time difference between consecutive page requests, is calculated for each page. For the last page of the user session, we set the page time to be the mean of visiting page times for the page taken across all sessions in which the page is not the last page request. A new session is created when a new IP-address is encountered or if the visiting page time exceeds 30 minutes for the same IP-address. Thus, a session consists of ordered sequence of page visits. We eliminate sessions whose *session length*[3] is less than or equal to 2 in order to eliminate the effect of random accesses to the Web site. It is important to note that these are only heuristics to identify users and user sessions, and other heuristics may be employed in future studies.

The visiting times are normalized across the visiting times of the pages in the same session, such that the normalized time has a value between 1 and 10. If a page is not in the user session, then the value of corresponding normalized time is set to 0. This normalization captures the relative importance of a page to a user in a transaction. The *aggregate interest* of a user in a transaction is then defined by a vector which consists of the normalized visiting times of that transaction. In order to determine navigation pages that provide links to guide users to the content pages, the requests are counted for each page in the transaction data sets. This process

[2]Except further cleaning techniques for the "NASA" data set of which the details are given in the next section.

[3]The length of a session or transaction is determined by the number of pages requested by one user within a server session.

shows that the page requests are very scattered, i.e. even the most popular pages such as home pages are requested in about 10% of the transactions. Since our objective is to recommend pages that contain a portion of the information content that the Web site provides, the page views that appear in more than 10% of the transactions are eliminated from the transaction data sets. We apply FP-tree algorithm [14] for discovering pages that are frequently requested together. Pages that appear together in more than 1% of all transactions are used for recommendation in order to capture the relationship between page requests. This filtering step produces a set of URL's $P = \{p_1, ..., p_n\}$. The pages that are not in the set of P are removed from the user transactions. Finally, a filtering method is applied in order to eliminate transactions whose length is less than 4 or longer than twice the average length of the transactions. Since the data sets have different characteristics, the cleaning step results in different numbers of transactions and page numbers. Even before filtering the data, 80% of sessions in "ClarkNet" Web log and "University of Saskatchewan" Web log have lengths less than four page requests. After cleaning the data sets, the number of transactions are decreased significantly in these logs. Table 1 shows the number of remaining URL's and the number of transactions for each data set. The output of this step is a set of user transactions, where each user transaction is in the form of: $\langle t_i,\ \{\langle page\ requests\rangle\},\ (time_{p_1}, time_{p_2}, ..., time_{p_n})\rangle$, where t_i is a unique transaction number and $\langle page\ requests\rangle$ is a subset of P. The aggregate interest in transaction t_i is represented by the vector, $(time_{p_1}, time_{p_2}, ..., time_{p_n})$, where $time_{pi}$ is the normalized visiting time of page p_i if p_i is in the $\langle page\ requests\rangle$, or 0 otherwise.

	NASA	ClarkNet	University of Saskatchewan
Number of URL's	92	67	171
Number Of Transactions	15369	6846	7452

Table 1: Characteristics of Cleaned Log Data Sets

Example 1. A sample transaction set, for a Web site with ten pages, is shown in Table 2. For simplification, we represent page requests with unique page numbers, such that each page number corresponds to a page in P. □

3.2 Clustering User Transactions in Web Log Data

In this section, we first describe the specific mixture model that we use for clustering the user transactions in Web log data. Next, the update parameters for training the mixture model of Poisson distributions with the Expectation Maximization algorithm are given. We use a model-based technique to group the user transactions according to the interests of users in each transaction. We assume the data to be generated in the following fashion:

1. When a user arrives at a Web site, his or her transaction is assigned to one of G clusters with some probability.

2. Given that a user's transaction is in a cluster, his or her next request in that transaction is generated according to a probability distribution specific to that cluster.

Transaction Number	Page Number	Aggregate Interest
1	[0, 2, 5, 4, 9]	[1, 0, 8, 0, 3, 10, 0, 0, 0, 1]
2	[3, 8, 5, 9, 6]	[0, 0, 0, 2, 0, 1, 10, 0, 8, 10]
3	[6, 5, 4, 3, 9, 8]	[0, 0, 0, 2, 1, 1, 9, 0, 8, 10]
4	[0, 2, 5, 4, 9, 8, 1]	[10, 10, 3, 0, 1, 6, 0, 0, 1, 4]
5	[0, 9, 7, 1, 4, 2, 5]	[1, 1, 7, 0, 3, 10, 0, 1, 0, 1]
6	[8, 9, 5, 2, 1, 0]	[10, 9, 2, 0, 0, 6, 0, 0, 1, 4]
7	[0, 4, 5, 2]	[1, 0, 8, 0, 4, 10, 0, 0, 0, 0]

Table 2: A transaction set as running example

As mentioned in Section 2, we assume that the data are produced by a mixture model. Every transaction is generated according to the probability distribution defined by a subset of model parameters, denoted $\mathbf{\Theta}_g$. Let $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$ be a set of K user transactions and $\mathbf{C}$ be a discrete valued variable taking values $c_1, ..., c_G$, which corresponds to an unknown cluster assignment of a user transaction. Then the mixture model for a user transaction is:

$$\begin{aligned} p(\mathbf{X} = \mathbf{x}_i|\mathbf{\Theta}) &= \sum_{g=1}^{G} p(\mathbf{C} = c_g|\mathbf{\Theta})p(\mathbf{X} = \mathbf{x}_i|\mathbf{C} = c_g, \mathbf{\Theta}_g) \\ &= \sum_{g=1}^{G} \tau_g p(\mathbf{X} = \mathbf{x}_i|c_g, \mathbf{\Theta}_g) \qquad (10) \end{aligned}$$

where τ_g is the probability of selecting cluster c_g. A user transaction, $\mathbf{x}_i$, is considered to be an $n-$dimensional vector of visiting page times, $(x_{i1}, x_{i2}, ..., x_{in})$, where x_{ij} is the normalized time that the user spent in page p_j; each p_j is a page view in the set of pages (in a given site) $WP = \{p_1, p_2, ..., p_n\}$. These times are normalized across the pages in a single transaction and the n-dimensional vector represents aggregate interest of the user as mentioned in the previous section.

In our case, the mixture model can be regarded as a distribution in which the class labels are missing. Although we reduce the dimensions of the input data using frequent pattern mining, there is still a problem of how to estimate the probabilities. One of the key ideas to handle this problem is to impose a structure on the underlying distribution, for example by assuming the independence of dimensions:

$$p(\mathbf{x}_i) = \prod_{j=1}^{n} p_j(x_{ij}) \qquad (11)$$

Since a user transaction is an n-dimensional vector of normalized visiting times, we can easily adapt this assumption to our model. Even the order of visiting pages may be different in two user transactions, each transaction can be represented by the equal vectors if the normalized page times corresponding to the same page in each transaction are equal.

Example 2. To illustrate the independence assumption for our model, consider the transaction 1, 4 and 7 in Table 2. The order of page requests in transaction 1 and 7 are different.

However, the aggregate interests of these transactions are very similar, because the normalized page times of each page are similar. Although the first 5 pages in transactions 1 and 4 are requested in the same order, the aggregate interests of these transactions are not similar. According to our clustering criteria transactions 1 and 7 would be in the same cluster, whereas transaction 4 would be in a different cluster. Thus, the value of the m^{th} dimension of a transaction, where $m \in n$, is independent to the values in the preceding dimensions. □

The independence assumption enables us to use n separate probability distributions to model each dimension of a user transaction. To model this data, we assume that the data at each dimension have been generated by a mixture of Poisson distributions. A random variable X has a *Poisson distribution* with parameter m if for some $m > 0$ [4]:

$$p(X = k) = e^{-m}\frac{m^k}{k!} \qquad k = 0, 1, ...$$

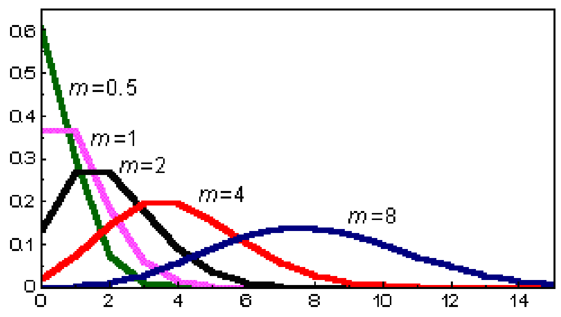

Figure 1: Shape of the Poisson distribution with different parameters

Figure 1 presents the shape of the Poisson distribution with different parameters, m. As m increases, the shape of the Poisson distribution begins to resemble a bell shaped distribution. The Poisson model can be used to model the rate at which individual events occur, for example the rate at which a user transaction has the value 1 for a particular page. To confirm our assumption, that the data in each dimension have been generated by a Poisson distribution, the histogram of the occurrence of each of the ten possible values at each dimension has been plotted. The histograms verify our assumption. Figure 2 presents one of these histograms. As can be seen, the histogram has the shape of the Poisson distribution with a low parameter m.

According to the independence assumption, a user transaction $\mathbf{x}_i$ is generated in a cluster g by a Poisson model as follows:

$$p(\mathbf{x}_i|c_g, \mathbf{\Theta}_g) = \prod_{j=1}^{n} \frac{(\theta_{gj})^{x_{ij}}\ e^{-\theta_{gj}}}{x_{ij}!} \tag{12}$$

where θ_{gj} is the parameter of the Poisson distribution for a dimension j in cluster g.

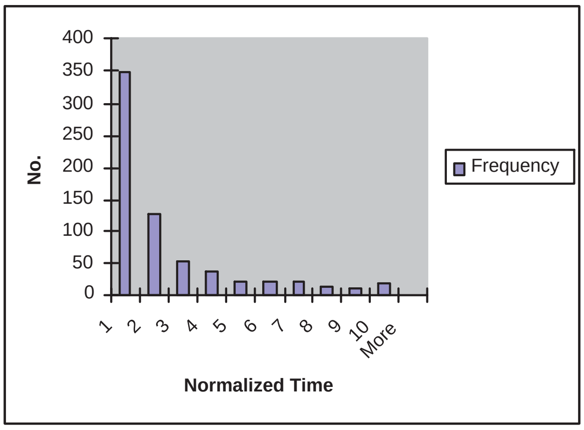

Figure 2: Histogram of a page from NASA Web server

By combining Equation (10) and Equation (12) we obtain:

$$p(\mathbf{x}_i|\mathbf{\Theta}) = \sum_{g=1}^{G} \tau_g \left(\prod_{j=1}^{n} \frac{(\theta_{gj})^{x_{ij}} \; e^{-\theta_{gj}}}{x_{ij}!} \right) \tag{13}$$

where θ_{gj} $(g \in [1...G], j \in [1...n])$ is the Poisson parameter of cluster c_g at dimension j.

Example 3. For the transaction set in Table 2, there are 10 Poisson parameters for each cluster, where the number of unique pages is 10 in that data set. □

The model parameters to be learned are then:

$$\mathbf{\Theta} = \{\mathbf{\Theta}_1, ..., \mathbf{\Theta}_G, \tau_1, ..., \tau_G\}, \; \mathbf{\Theta}_g = (\theta_{g1}, ..., \theta_{gn}) \, , \; \sum_{g=1}^{G} \tau_g = 1 \tag{14}$$

3.2.1 Learning the Model Parameters

We can train the model parameters of the mixture model, developed in the previous subsection, using EM algorithm where the conditional independence assumption is enforced during Maximization step. The learning algorithm is carried out for each component of the model. There are several reasons for using EM algorithm:

- We want to represent the behavior of the user in one transaction using Poisson distribution.
- It is linear to the number of transactions.
- It is robust to noisy data.
- It accepts as input the desired number of clusters.
- It provides a cluster membership probability per transaction.

page number	cluster 1	cluster 2	cluster 3
9	0.1917	0.2030	0.2049
8	0.2126	0.2380	0.2095
7	1.0020	1.0067	0.9942
6	0.0954	0.0923	0.1111
5	0.1395	0.1548	0.1623
4	0.4321	0.4011	0.4178
3	0.4864	0.4824	0.4972
2	0.1711	0.1685	0.1728
1	0.1464	0.1377	0.1481
0	0.2204	0.2207	0.1996

Table 3: Poisson parameters for three clusters

- It can handle high dimensionality.
- It converges fast given a good initialization.

In order to implement the EM algorithm we should pick the number of clusters (G), an initial starting point ($\mathbf{\Theta}'(0)$), and a convergence criteria. To determine the number of clusters, we run the algorithm with several numbers of clusters. We initialize the parameters of our components, $\mathbf{\Theta}_g$, ($g \in [1...G]$) by estimating the Poisson parameters for a single component model and then randomly perturbing the parameter values by a small amount to obtain G sets of parameters. Finally, we determine the convergence criteria such that the algorithm converges when the log likelihoods of two consecutive iterations on the training data differ less than 0.001%. There is a trade-off between the estimation accuracy of parameters and the number of iterations. With a smaller value the number of iteration required for convergence will enlarge so that the algorithm converge in a longer period of time. If it is greater than the selected value then the estimation for the parameters would be less precise.

Example 4. Let us determine the initial parameters of the EM algorithm for the data set in Table 2. Assume that the number of clusters is 3 and the cluster prior probabilities are set to $\tau_g = 1/3$ where $g \in [1, 2, 3]$. We determine 10 initial values for Poisson parameters for each cluster, giving a total of 30 Poisson parameters for the model, as mentioned earlier. Table 3 presents these parameters. Then, for the first iteration of the EM algorithm the cluster prior of the first cluster would be $\tau_1 = 1/3$, and the Poisson parameter of the second dimension in that cluster is $\theta_{11} = 0.2207$. $\square$

To compute the Q-function of Equation (9) in the E-step, we should compute the conditional probability of missing class labels given the current parameter set $\mathbf{\Theta}'$. We define this probability as *cluster-posterior* probability, $P_{ig}(\mathbf{\Theta}')$, that the transaction $\mathbf{x}_i$ arose from the g^{th} cluster. We can write the cluster-posterior probability using Bayes'rule as:

$$\begin{aligned} P_{ig}(\boldsymbol{\Theta}') &= p(\mathbf{C} = c_g|\mathbf{x}_i) \\ &= \frac{p(\mathbf{C} = c_g)p(\mathbf{x}_i|c_g, \boldsymbol{\Theta}'_g)}{p(\mathbf{x}_i)} \\ &= \frac{\tau_g p(\mathbf{x}_i|c_g, \boldsymbol{\Theta}'_g)}{\sum_{j=1}^{G} \tau_j p(\mathbf{x}_i|c_j, \boldsymbol{\Theta}'_j)} \end{aligned} \tag{15}$$

The Q-function can be written as:

$$Q(\boldsymbol{\Theta}, \boldsymbol{\Theta}') = \sum_{i=1}^{K} \sum_{g=1}^{G} P_{ig}(\boldsymbol{\Theta}') \left[\ln p(\mathbf{x}_i|c_g, \boldsymbol{\Theta}_g) + \ln \tau_g\right] \tag{16}$$

In M-step, keeping the cluster-posterior probabilities fixed, we reassign a new set of parameters $\boldsymbol{\Theta}'(n+1)$ so as to maximize the expected log likelihood of the training data. The Q-function is maximized subject to the constraint that the cluster priors sum to 1. In order to perform constrained maximization, a Lagrange multiplier is used. The estimating equations for cluster priors are as follows:

$$\begin{aligned} \frac{\partial}{\partial \tau_g} \left[Q(\boldsymbol{\Theta}, \boldsymbol{\Theta}') - \lambda \sum_{j=1}^{G} \tau_j \right] &= 0 \\ \sum_{i=1}^{K} P_{ig}(\boldsymbol{\Theta}') \left[\frac{1}{\tau_g}\right] - \lambda &= 0 \end{aligned} \tag{17}$$

from which it follows:

$$\lambda \tau_g = \sum_{i=1}^{K} P_{ig}(\boldsymbol{\Theta}') \tag{18}$$

If we sum Equation (18) over g we obtain:

$$\lambda = \sum_{i=1}^{K} \sum_{g=1}^{G} P_{ig}(\boldsymbol{\Theta}') = K \tag{19}$$

Last equation follows from the fact that $\sum_{g=1}^{G} P_{ig}(\boldsymbol{\Theta}') = 1$. By combining Equation (18) and Equation (19), we obtain the equation for updating the cluster probabilities:

$$\widehat{\tau}_g = \frac{1}{K} \sum_{i=1}^{K} P_{ig}(\boldsymbol{\Theta}') \tag{20}$$

Similarly we can maximize the Q-function with respect to the parameters of Poisson model, $\mathbf{\Theta}_g$, under the independence assumption:

$$\begin{aligned}
\frac{\partial}{\partial \theta_{gm}} \left[Q(\mathbf{\Theta}, \mathbf{\Theta}') \right] &= 0 \\
\frac{\partial}{\partial \theta_{gm}} \left[\sum_{i=1}^{K} P_{ig}(\mathbf{\Theta}'_g) \left(\ln \prod_{j=1}^{n} \frac{(\theta_{gj})^{x_{ij}}\ e^{-\theta_{gj}}}{x_{ij}!} + \ln \tau_g \right) \right] &= 0 \\
\sum_{i=1}^{K} P_{ig}(\mathbf{\Theta}') \left[\frac{x_{im}}{\theta_{gm}} - 1 \right] &= 0 \qquad (21)
\end{aligned}$$

which yields the following update equation for Poisson parameters:

$$\widehat{\theta}_{gm} = \frac{\sum_{i=1}^{K} (P_{ig}(\mathbf{\Theta}') x_{im})}{\sum_{i=1}^{K} P_{ig}(\mathbf{\Theta}')} \qquad (22)$$

At the end of the EM algorithm each cluster has its own set of parameters such that:

$$pc_g = \{\tau_g, (\theta_{g1}, ..., \theta_{gn})\}$$

Example 5. For the data set in Table 2 we compute in the E-step the cluster posterior probabilities using Equation (15). In the M-step we update the model parameters using Equation (20) and Equation (22). Thus, the parameters in Table 3 and the cluster priors are updated in each M-step. The E and M-steps are applied until the convergence criteria is obtained. The output of this algorithm is the set of cluster parameters. For example, $\{0.3; (0.02, 1.2, ...)\}$ tells us that a cluster has a prior probability of 0.3 and the Poisson parameter of the first page is 0.02, of the second page is 1.2 and so on. The clusters in Table 4 are obtained by assigning each transaction to the cluster that has the highest cluster-posterior probability. □

Cluster No.	Transaction Number	Page Number	Normalized Time
1	1	[0, 2, 5, 4, 9]	[1, 0, 8, 0, 3, 10, 0, 0, 0, 1]
	5	[0, 9, 7, 1, 4, 2, 5]	[1, 1, 7, 0, 3, 10, 0, 1, 0, 1]
	7	[0, 4, 2, 5]	[1, 0, 8, 0, 4, 10, 0, 0, 0, 0]
2	2	[3, 8, 5, 9, 6]	[0, 0, 0, 2, 0, 1, 10, 0, 8, 10]
	3	[6, 5, 4, 3, 9, 8]	[0, 0, 0, 2, 1, 1, 9, 0, 8, 10]
3	4	[2, 4, 9, 5, 8, 0, 1]	[10, 10, 3, 0, 1, 6, 0, 0, 1, 4]
	6	[8, 9, 5, 2, 1, 0]	[10, 9, 2, 0, 0, 6, 0, 0, 1, 4]

Table 4: Clusters built in Example 2

3.2.2 Cluster Profiles

In order to obtain a set of pages for recommending and rank these pages in this set *recommendation scores* are calculated for every page in each cluster using the Poisson parameters of that cluster. Thus, each cluster has a set of recommendation scores additional to its parameter set created in the previous subsection. We modify the cluster parameters such that each cluster has a recommendation score set, $RS_g = \{rs_{g1}, ..., rs_{gn}\}$, where $rs_{gi}, i \in [1, ..., n]$ is the recommendation score for page p_i in cluster c_g. The updated cluster parameters are then in the form:

$$pc_g = \{\tau_g; (\theta_{g1}, ..., \theta_{gn}); (rs_{g1}, ..., rs_{gn})\}$$

Those are the only parameters that the system needs in order to produce a set of pages for recommendation. We define the number of parameters stored in the memory as *model size*. It is clear that the smaller the model size the faster the online prediction.

We use five different methods for calculating recommendation scores for every page. The recommendation scores are then normalized such that the maximum score has a value of 1. These methods are as follows:

Method 1. For the first method, we only use the Poisson parameters of the active cluster as recommendation scores, namely:

$$rs_{gi} = \theta_{gi} \tag{23}$$

For the remaining calculations we assign each transaction in the training set to a cluster that has the highest posterior probability. Next we count the number of requests for every page in each cluster. We define this number as popularity, (f_{gi}), where $i \in [1, ..., n]$ and $g \in [1, ..., G]$. For example, if R_{p_i} is the total number of page requests for page p_i in the a cluster c_g, then the popularity of that page in that cluster is:

$$f_{gi} = R_{p_i}$$

Method 2. In the second method we use only the popularity information for recommending pages. The intuition behind this is to recommend pages that are most likely visited in a cluster. The recommendation score for the page p_i in active cluster c_g is then:

$$rs_{gi} = f_{gi} \tag{24}$$

Method 3. For the third method, we calculate recommendation scores by multiplying the popularity by the Poisson parameter:

$$rs_{gi} = f_{gi} \times \theta_{gi} \tag{25}$$

According to our clustering criteria, the normalized visiting page times for a given page in a cluster should not vary greatly among transactions. Thus, we take advantage of a technique used in decision theory called the *entropy*. We calculate the entropy for each page using the relative frequency of each of the ten possible values of normalized times. A low entropy value means that the visiting time of that page mostly has one of the normalized values. High entropy value, on the other hand, indicates wide divergence in page visiting times among transactions.

Method 4. We use for the fourth recommendation method the entropy values. Our recommendation score is then:

$$rs_{gi} = f_{gi} \times \frac{1}{(entropy)_{gi}} \times \theta_{gi} \tag{26}$$

Method 5. For the last calculation, the *log* of the popularity is taken in order to decrease the effect of the popularity in recommendation score:

$$rs_{gi} = \log f_{gi} \times \frac{1}{(entropy)_{gi}} \times \theta_{gi} \tag{27}$$

3.3 Recommendation Engine

The real-time component of the model calculates cluster posterior probability $P(c_g|s_i)$ for every cluster $c_g \in C = \{c_1, ..., c_G\}$ where s_i is the portion of a transaction in test set that is used to find the most similar cluster. The active transaction is assigned to the cluster that has the highest probability. We define this cluster as the *active cluster*. A *recommendation set*, which is the set of predicted pages by the model, is then produced ranking the recommendation scores of active cluster in descending order. The recommendation set consists of pages which have a recommendation score greater than a threshold ξ (or top N items with the highest recommendation scores where N is a fixed number) in the active cluster and that the user has not visited yet. The choice of specific alternative depends on the evaluation metric discussed in the next section.

4 Experimental Results

In this research we use three different transaction sets prepared for experiments as mentioned in Section 3. We measure the performance of our technique using the proposed methods for calculating recommendation scores. Approximately 30% of these cleaned transactions are randomly selected as the test set, and the remaining part as the training set. The experiments are repeated with different number of clusters and with different initial parameters for EM algorithm.

We define the following metrics to evaluate our method:

Hit-Ratio Given the visiting time of a page in the current transaction, the model recommends three pages that have the highest recommendation score in the active cluster. A hit is declared if any one of the three recommended pages is the next request of the user. The hit-ratio is the number of hits divided by the total number of recommendations made by the system.

Precision For each transaction t in the test set we select the first w requests in t. These w requests are used to calculate the active cluster and produce the recommendation set. The recommendation set contains all the pages that have a recommendation score greater than the threshold ξ and that are not in the first w requests. We denote this set as $PS(w, \xi)$ and the number of pages in this set that match with the remaining part of active transaction as m. Then the precision for a transaction is defined as:

$$precision(t) = \frac{m}{|PS(w, \xi)|} \tag{28}$$

We perform the experiments with different number of clusters. We choose these numbers according to the number of transactions in the training sets and the number of pages in the Web site. We identify that the values for the number of clusters in Table 5 are best among the other values we consider. For these numbers we have a higher *log* likelihood for the training sets as well as a better prediction accuracy for the test sets. The increase of the *log* likelihood means that the model fit better to the data. Table 6, 7 and 8 present the prediction accuracy of the model for different number of clusters. As can be seen in the tables, the model is insensitive to the number of clusters in a reasonable range around the best numbers of clusters. The remarkable changes in the number of clusters results in a decrease of the performance of the model.

As can be seen in the tables the accuracy of predictions greatly increases in "NASA" data set, when we use "Hit-Ratio" metric. The cause for that may be that we apply further cleaning methods to this data set. Some of the page views in the log data are still available in the "NASA" Web site. On the other hand, "ClarkNet" Web server does not exist anymore and the URL requests in the "University of Saskatchewan" log data are not up-to-date. The unique page views are obtained in the last two data sets by listing the unique URL's, whereas the pages of the Web site of NASA Kennedy Space Center are retrieved using a *web crawler* implemented for this work. Determining different URL's that correspond to the same page in the Web site produces a significant improvement in the prediction accuracy for "NASA" data set.

	NASA no. of Clusters = 23 $w = 2$, $\xi = 0.5$		ClarkNet no. of Clusters = 4 $w = 2$, $\xi = 0.5$		University of Saskatchewan no. Clusters = 8 $w = 2$, $\xi = 0.5$	
	Hit-Ratio	Precision	Hit-Ratio	Precision	Hit-Ratio	Precision
Method 1	41%	29%	35%	34%	35%	33%
Method 2	42%	28%	40%	30%	38%	32%
Method 3	43%	33%	38%	36%	38%	38%
Method 4	43%	34%	38%	36%	38%	39%
Method 5	43%	30%	38%	33%	35%	33%

Table 5: Results Of the Recommendation Model

As mentioned in the previous section, we use 5 different methods for calculating recommendation scores. The application of methods that calculate the recommendation scores using popularity term results in marked improvement of the prediction accuracy. This is not surprising, because the popularity represents the common interest among transactions in each cluster. The results show that using entropy during calculation of recommendation score does not improve the accuracy as much as we expected. Initially, we assumed this may be due to the fact that EM algorithm learns the best model in terms of model parameters. However, even the results of experiments with different number of clusters reflect the same characteristic. Further

	NASA no. Clusters = 20 $w = 2$, $\xi = 0.5$		ClarkNet no. Clusters = 8 $w = 2$, $\xi = 0.5$		University of Saskatchewan no. of Clusters = 10 $w = 2$, $\xi = 0.5$	
	Hit-Ratio	Precision	Hit-Ratio	Precision	Hit-Ratio	Precision
Method 1	41%	29%	36%	32%	33%	33%
Method 2	42%	28%	42%	29%	39%	32%
Method 3	42%	33%	40%	35%	36%	35%
Method 4	43%	34%	40%	35%	36%	38%
Method 5	43%	30%	39%	33%	35%	34%

Table 6: Results Of the Model for different Number of Clusters

	NASA no. Clusters = 10 $w = 2$, $\xi = 0.5$		ClarkNet no. Clusters = 20 $w = 2$, $\xi = 0.5$		University of Saskatchewan no. of Clusters = 4 $w = 2$, $\xi = 0.5$	
	Hit-Ratio	Precision	Hit-Ratio	Precision	Hit-Ratio	Precision
Method 1	37%	27%	34%	31%	32%	31%
Method 2	41%	27%	36%	29%	34%	30%
Method 3	41%	29%	36%	32%	34%	33%
Method 4	41%	32%	36%	32%	34%	34%
Method 5	39%	27%	35%	31%	33%	31%

Table 7: Results Of the Model for different Number of Clusters

examination of the cluster profiles indicate that the popularity of some pages in most of the clusters are zero due to the sparse and scattered nature of the data. Thus, we can not observe the effect of the entropy, since we multiply the inverse of the entropy with popularity for calculating the recommendation scores. One way to address this issue may be to use MAP estimation when model parameters are learned. This will smooth the popularity to have non-zero values for pages that have not been requested in a cluster. However, in general we can use method 4 for calculating recommendation scores discarding the metric we use for evaluation.

For evaluating the effect of the Poisson model, we repeated the experiments with the same training sets and the same number of clusters using a different clustering algorithm. We selected k-means algorithm, because it is comparable to our model in terms of speed and memory usage. For evaluation we measured only the precision of the test sets. The “ClarkNet” data set has a precision of 15%, whereas the “NASA” data set has 4% and the “University of Saskatchewan” has 5%. These results prove that modelling the user transaction with a mixture of Poisson distributions produces satisfactory prediction rates with an acceptable computational complexity in real-time and memory usage.

We provide some intuitive arguments for why our model has an advantage in terms of speed and memory usage. The online prediction time correlates strongly with the model size. The smaller the model size the faster the online recommendation. Since we only store the cluster parameters for the prediction of the next page request, our model size is very small. The model

	NASA no. Clusters = 30 $w = 2$, $\xi = 0.5$		ClarkNet no. Clusters = 30 $w = 2$, $\xi = 0.5$		University of Saskatchewan no. of Clusters = 30 $w = 2$, $\xi = 0.5$	
	Hit-Ratio	Precision	Hit-Ratio	Precision	Hit-Ratio	Precision
Method 1	33%	25%	34%	31%	33%	30%
Method 2	36%	25%	36%	29%	37%	33%
Method 3	35%	28%	36%	32%	37%	33%
Method 4	36%	29%	36%	33%	38%	33%
Method 5	35%	25%	39%	32%	35%	31%

Table 8: Results Of the Model for different Number of Clusters

size only increases with the number of clusters or the number of pages in the Web site when the Web site has a complex structure. However, it is clear that in that case the application of methods such as sequential pattern mining, association rules or Markov models generate more complex models due to the increasing size of rules or states. Thus, all of this models require some pruning steps in order that they be effective. However, our model provides a high prediction accuracy with a simple model structure.

5 Conclusion and Future Work

We have considered the problem of modelling the interest of a Web user during his or her single visit to the Web site. In this article, the mixture of Poisson model is proposed for modelling the interest of a user in one transaction. The experiments show that the model can be used on Web sites with different structures. Although one of the logs in the experiments were from a commercial Web site the results from this data set were satisfying. To confirm our finding, we compare our model to k-means clustering algorithm. Results show that our model improves the efficiency significantly. As stated before, although we do not use the information about the request order of pages in transactions, the proposed model is able to very efficiently capture the sequential behavior of users.

We are now extending the model in several ways. The implemented filtering method removes pages that are not in frequent item sets of length bigger than one. In the future version of the model, we will propose auxiliary methods for recommendation in case the current page request is not in the frequent item sets. Other improvements would be to update the model parameters as the training set is incremented. Since the log data of the Web site increases due to new users' requests, over time it is highly desirable to perform the update of the model parameters incrementally. Further extension of the model can be achieved by using a different method for normalization of page times in order to apply the model in real-time. The page time will be divided into 10 separate values using time windows. For example, if a page is visited less than the first time window, it's normalized time could be set to 1. The time windows will be determined by applying a statistical analysis on the visited times of pages. Finally, a deeper study is needed for determining the initial parameters of the EM algorithm and for learning the model parameters using MAP estimation. With proper initial parameters for the clustering algorithm and a better predictor for model parameters the results may be better.

References

[1] C. C. Aggarwal, J. L. Wolf, and P. S. Yu. Caching on the world wide web. *IEEE Transactions on Knowledge and Data Enginnering*, 11(1):95–107, Feb. 1999.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. *Proceedings of the International Conference on Data Engineering (ICDE), Taipei, Taiwan*, March 1995.

[3] C. R. Anderson, P. Domingos, and D. S. Weld. Relational markov models and their application to adaptive web navigation. *Proceedings of the Eighth ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining*, July 23–26 2002. Edmonton, AB, Canada.

[4] R. Bartoszyński and M. Niewiadomska-Bugaj. *Probability and Statistical Inference.* John Walley & Sons, Inc., 1996.

[5] J. Borges and M. Levene. Data mining of user navigation patterns. *Proceedings of the WEBKDD'99 Workshop on Web Usage Analysis and User Profiling*, pages 31–36, August, 15 1999. San Diego, CA, USA.

[6] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of theFourteenth Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[7] S. Brin and L. Pagepp. The anatomy of large-scale hypertextual web search engine. *Proceedings of Int. Worls Wide Web Conference (WWW'98)*, pages 107–117, 1998.

[8] R. Cooley, B. Mobasher, and J. Srivastava. Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems*, 1(1), 1999.

[9] A. P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society*, 39(1):1–38, 1977.

[10] M. Deshpande and G. Karypis. Selective markov models for predicting web-page accesses. *Proceedings of the First SIAM International Conference on Data Mining (SDM'2001)*, 2001.

[11] O. Etzioni. The world wide web: Quagmire or gold mine. *Communications of the ACM*, 39(11):65–68, 1996.

[12] J. Han and M. Kamber. *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2001.

[13] J. Han, M. Kamber, and A. K. H. Tung. Spatial clustering methods in data mining: A survey. *H. Miller and J. Han (eds.),* Geographic Data Mining and Knowledge Discovery, *Taylor and Francis*, 2001.

[14] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *Proceedings 2000 ACM-SIGMOD International Conf. on Management of Data (SIGMOD'00)*, May 2000. Dallas, TX.

[15] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining.* The MIT Press, 2001.

[16] J.Pitkow and P.Pirolli. Mining longest repeating subsequences to predict world wide web surfing. *Proceeding USENIX Symposium on Internet Technologies and Systems (USITS'99)*, Oct. 1999.

[17] R. Kosala and H. Blockeel. Web mining research: A survey. *ACM SIGKDD Explorations*, 2(1):1–15, 2000.

[18] ClarkNet WWW Server Log. http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html.

[19] NASA Kennedy Space Center Log. http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html.

[20] S.K. Madria, S.S. Bhowmick, W.K. Ng, and E.-P. Lim. Research issues in web data mining. *Proceedings of Data Warehousing and Knowledge discovery, First Intenational Conference,DaWaK '99*, pages 303–312, 1999.

[21] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery of aggregate usage profiles for web personalization. *Proceedings of the Web Mining for E-Commerce Workshop (WebKDD'2000)*, held in conjunction with the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000), Aug. 2000. Boston.

[22] B. Mobasher, H. Dai, T. Luo, Y. Sung, and J. Zhu. Integrating web usage and content mining for more effective personalization. *Proceedings of the International Conference on E-Commerce and Web Technologies (ECWeb2000)*, Sept. 2000. Greenwich, UK.

[23] O. Nasraoui, R. Krishnapuram, and A. Joshi. Mining web access logs using a fuzzy relational clustering algorithm based on a robust estimator,toronto, canada. *Eight International World Wide Web Conference*, 1999.

[24] The University of Saskatchewan Log. http://ita.ee.lbl.gov/html/contrib/Sask-HTTP.html.

[25] P Pirolli, J. Pitkow, and 118-125 R. Rao. Silk from a sow's ear: Extracting usable structures from the web. *Proceedings of CHI'96 (Vancouver BC), ACM Press*, pages 118–125, April 1996.

[26] R. R. Sarukkai. Link prediction and path analysis using markov chains. *Proceedings of the Ninth International World Wide Web Conference, Amsterdam*, 2000.

[27] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system – a case study. *Proceedings of the WebKdd 2000 workshop at the ACM SIGKDD 2000*, 2000.

[28] C. Shahabi, A. Zarkesh, J. Adibi, and V. Shah. Knowledge discovery from users web-page navigation. *Proceeding of the IEEE RIDE97 Workshop, pages 20-29, Birmingham, England*, April 1997.

[29] J. Shim, P. Scheuermann, and R. Vingralek. Proxy cache algorithms: Design, implementation and performance. *IEEE Transactions on Knowledge and Data Engineering*, 11(4):549–562, Aug. 1999.

[30] R. Srikant and R. Agrawal. Mining generalized association rules. *Proceedings of the 21st International Conference on Very Large Databases, Zurich, Switzerland*, Sep. 1995.

[31] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. *Proceedings of the Fifth International Conference on Extending Database Technology (EDBT), Avignon, France*, March 1996.

[32] J. Srivastava, R. Cooley, M. Deshpande, and P. N. Tan. Web usage mining: Discovery and application of usage patterns from web data. *ACM SIGKDD Explorations*, 1(2):12–23, 2000.

[33] H. Yang, S. Parthasarathy, and S. Reddy. On the use of constrained associations for web log mining. *Proceedings of Fourth WEBKDD Workshop*, 2002. Edmonton, AB, Canada.

[34] O. R. Zaïane. Web usage mining for a better web-based learning environment. *Proceedings of Conference on Advanced Technology for Education*, pages 60–64, June 27–28 2001. Banff, Alberta.