A User Interest Model for Web Page Navigation

Șule Gündüz[⋆] and M. Tamer Özsu

School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
{sgunduz,tozsu}@db.uwaterloo.ca

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 [4]. 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. There are three steps in this process [2]. 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. The prediction

⋆ On leave from Department of Computer Science, Istanbul Technical University, Istanbul, Turkey.

step is the real-time processing of the model, which considers the active user session and makes recommendations based on the discovered patterns.

An important feature of the user's navigation path in a *server session*[1] is the time that a user spends on different pages [11]. Even the same person 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.

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. We employ a model based clustering algorithm and partition user sessions according to the similar amount of time that is spent on similar pages within a session. Since we do not have the actual cluster assignments, we use a standard learning algorithm, the Expectation-Maximization (EM) [3], 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 techniques and describes equations for training a mixture model with EM algorithm. Section 3 presents the proposed model. Section 4 provides detailed experimental results. In Section 5, we examine related work. Finally, in Section 6 we conclude and discuss future work.

2 Background

2.1 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

[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 [2]. In this paper we will use this term interchangeably with "user session" and "user 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 [5].

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 together 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".

2.2 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.

Model-Based Cluster Analysis. 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, defined by a set of parameters, denoted $\mathbf{\Theta}$ [6]. An observation $\mathbf{x}_i$ in a data set of K observations, $\mathbf{D} = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$, is generated by a mixture of G components as follows:

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

where $\mathbf{\Theta}_g$ ($g \in [1...G]$) is a vector specifying the probability distribution function (pdf) of the g^{th} component, c_g, and $\sum_{g=1}^{G} p(c_g|\mathbf{\Theta}) = \sum_{g=1}^{G} \tau_g = 1$.

Statisticians refer to such a model as *mixture model with G components*. The *maximum likelihood* (*ML estimation*) approach maximizes the *log* likelihood of the training data in order to learn the model parameters:

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

3 Web Page Recommendation 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 [8]. The second log is from ClarkNet Web server which is a full Internet access provider for the Metro Baltimore-Washington DC area [7]. 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 [10]. 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

[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.

to guide users to the content pages, the requests are counted for each page in the transaction data sets. This process 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. Some of the page views appear together 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. We apply FP-tree algorithm [5] 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: $\langle t_i,\ \{\langle \textit{page requests}\rangle\},\ (time_{p_1}, time_{p_2}, ..., time_{p_n})\rangle$, where t_i is a unique transaction number and $\langle \textit{page requests}\rangle$ 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 $\langle \textit{page requests}\rangle$, 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

3.2 Clustering User Transactions in Web Log Data

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. 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) $WP = \{p_1, p_2, ..., p_n\}$. The n-dimensional vector represents aggregate interest of the user as mentioned in the previous subsection. 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 [6]. Since a user transaction is an n-dimensional vector of normalized visiting times, we can easily adapt this assumption to our model. 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 $m > 0$:

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

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. According to the independence assumption, a user transaction $\mathbf{x}_i$ is generated in a mixture model of Poisson distributions as follows:

$$p(\mathbf{x}_i|\mathbf{\Theta}) = \sum_{g=1}^{G} \tau_g \left(\prod_{j=1}^{n} \frac{e^{-\theta_{gj}} \ (\theta_{gj})^{x_{ij}}}{x_{ij}!} \right) \tag{3}$$

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

The model parameters to be learned are then:

$$\mathbf{\Theta} = \{\mathbf{\Theta}_1, ..., \mathbf{\Theta}_G, \tau_1, ..., \tau_G\}, \ \mathbf{\Theta}_g = (\theta_{g1}, ..., \theta_{gn}) \, , \ \sum_{g=1}^{G} \tau_g = 1 \tag{4}$$

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 E-step involves an update of 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. The M-step consists of the update of cluster priors and Poisson parameters. At the end of the EM algorithm each cluster has its own set of parameters such that:

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

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. 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:

$$rs_{gi} = \theta_{gi} \tag{5}$$

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 $g \in [1, ..., G]$.

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:

$$rs_{gi} = f_{gi} \tag{6}$$

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

$$rs_{gi} = f_{gi} \times \theta_{gi} \tag{7}$$

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:

$$rs_{gi} = f_{gi} \times \frac{1}{(entropy)_{gi}} \times \theta_{gi} \tag{8}$$

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:

$$rs_{gi} = \log f_{gi} \times \frac{1}{(entropy)_{gi}} \times \theta_{gi} \tag{9}$$

3.3 Recommendation Engine

The real-time component of the model calculates cluster posterior probability $P(c_g|w)$ for every cluster $c_g \in C = \{c_1, ..., c_G\}$ where w 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:

$$precision(t) = \frac{m}{|PS(w, \xi)|} \tag{10}$$

In our experiments, we try different values for the recommendation score ranging from 0.1 to 0.9. If the recommendation score is high then fewer recommendation are produced. If it is small then irrelevant pages are recommended with a low recommendation score. Our experiments show that setting ξ to 0.5 and w to 2 produces few bur highly relevant recommendations. 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 2 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 3 presents 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.

Table 2. Results (in %) Of the Model

Data Set	No.Of	Method 1		Method 2		Method 3		Method 4		Method 5	
	Clusters	H-R	Pre.	H-R	Pre.	H-R	Pre.	H-R	Pre.	H-R	Pre.
NASA	23	41	29	42	28	43	33	43	34	43	30
ClarkNet	4	35	34	40	30	38	36	38	36	38	33
U.of.Sask.	8	35	33	38	32	38	38	38	39	35	33

Table 3. Results (in %) Of the Model for different Number of Clusters

Data Set	No.Of	Method 1		Method 2		Method 3		Method 4		Method 5	
	Clusters	H-R	Pre.	H-R	Pre.	H-R	Pre.	H-R	Pre.	H-R	Pre.
NASA	10	37	27	41	27	41	29	41	32	39	27
NASA	30	33	25	36	25	35	28	36	29	35	25
NASA	20	41	29	42	28	42	33	43	34	43	30
ClarkNet	20	34	31	36	29	36	32	36	32	35	31
ClarkNet	30	34	31	38	30	38	33	38	33	36	32
ClarkNet	8	36	32	42	29	40	35	40	35	39	33
U.of.Sask.	4	32	31	34	30	34	33	34	34	33	31
U.of.Sask.	30	33	30	37	33	37	33	38	33	35	31
U.of.Sask.	10	33	33	39	32	36	35	36	38	35	34

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. The unique page views in that Web site are retrieved using a *web crawler* implemented for this work. Since "ClarkNet" Web server does not exist anymore and the URL requests in the "University of Saskatchewan" log data are not up-to-date, we can not apply the same technique to these data sets.

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. This may be due to the fact 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. 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 as in Table 2 using a different recommendation method. We use the recommendation model as proposed in [9], 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.

5 Related Work

The major classes of recommendation services are based on collaborative filtering techniques and the discovery of navigational patterns of users. The main techniques for pattern discovery are sequential patterns, association rules, Markov models, and clustering.

Collaborative filtering techniques predict the utility of items of an active user by matching, in real-time, the active user's preferences against similar records (nearest neighbors) obtained by the system over time from other users [1]. One shortcomings of these approaches is that it becomes hard to maintain the prediction accuracy in a reasonable range while handling the large number of items (dimensions) in order to decrease the on-line prediction cost.

Some authors have used association rules, sequential patterns and Markov models in recommender systems. 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. It may be possible to prune the rule space, enabling faster on-line prediction.

Page recommendations in [9] 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. Two crucial differences between our approach and the previous one are that we consider the user interest as a statistical model and we partition user sessions using a model-based approach. The latter method enables us to assign user transactions to more than one cluster with a probability, allowing calculation of recommendation scores using more than one cluster profile. Consequently, as the experiments demonstrate, our model's precision and robustness is superior. Furthermore, our model has the flexibility to represent the user interest with a mixture of binomial distributions (or with different distributions) if one wishes to ignore the visiting time in determining the navigational pattern. 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 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.

6 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. Finally, a deeper study is needed for determining the initial parameters of the EM algorithm. With proper initial parameters for the clustering algorithm the results may be better.

References

1. J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.
2. 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.
3. 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.
4. O. Etzioni. The world wide web: Quagmire or gold mine. *Communications of the ACM*, 39(11):65–68, 1996.
5. 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.
6. D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. The MIT Press, 2001.
7. ClarkNet WWW Server Log. http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html.
8. NASA Kennedy Space Center Log. http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html.
9. B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Improving the effectiveness of collaborative filtering on anonymous web usage data. *Proceedings of the IJCAI 2001 Workshop on Intelligent Techniques for Web Personalization (ITWP01)*, Aug. 2001. Seattle.
10. The University of Saskatchewan Log. http://ita.ee.lbl.gov/html/contrib/Sask-HTTP.html.
11. 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.