A Web Recommendation System Based on Maximum Entropy

Xin Jin, Bamshad Mobasher,Yanzan Zhou
Center for Web Intelligence
School of Computer Science, Telecommunication, and Information Systems
DePaul University, Chicago, Illinois, USA
{xjin,mobasher,yzhou}@cs.depaul.edu

Abstract

In this paper, we propose a Web recommendation system based on a maximum entropy model. Under the maximum entropy principle, multiple sources of knowledge about users' navigational behavior in a Web site can be seamlessly combined to discover usage patterns and to automatically generate the most effective recommendations for new users with similar profiles. In this paper we integrate the knowledge from page-level clickstream statistics about users' past navigations with the aggregate usage patterns discovered through Web usage mining. Our experiment results show that our method can achieve better prediction accuracy when compared to standard recommendation approaches, while providing a better interpretation of Web users' diverse navigational behaviors.

1. Introduction

Web recommendation systems have been shown to greatly help Web users in navigating the Web, locating relevant and useful information, and receiving dynamic recommendations from Web sites on possible products or services that match their interests. Web usage mining is one of the main approaches used to build Web recommendation systems. *Web usage mining* involves the applications of data mining and machine learning techniques to discover usage patterns (or build user models) through the analysis of Web users' historical navigational activities. A new user's profile can be matched with the discovered usage patterns to find "like-minded" users (sometimes referred to as *nearest neighbors*). The profiles of these nearest neighbors can then be used to recommend items of interest to the current user.

The discovery of usage patterns is the central problem in Web usage mining. By applying data mining or machine learning techniques to the users' historical navigational data, we are able to discover various usage patterns. Generally, most usage patterns capture shallow associations among pages or among users based on their navigation activities. We call such patterns *page-level patterns*. These patterns can capture common characteristics about users' behavior. However, they do not provide explicit insight into the users' underlying interests and preferences, thus limiting the effectiveness of recommendations as well as the ability of the system to interpret and explain the recommendations.

Recent work has focused on methods to model users preferences at "higher" abstraction levels. In [8], an approach is proposed to dynamically learn an online shopper's "taste", using a set of pre-defined semantic features, and generate recommendations. In [13], Web items (documents) are first clustered based on users' navigational data, and then user behavior models are built at the item-cluster level. In [11], a Web usage mining framework based on Probabilistic Latent Semantic Analysis (PLSA) is proposed. A set of hidden variables are used to represent users' common tasks when navigating the Web site. The PLSA model is applied to quantitatively characterize users' tasks, as well as the relationships between Web pages and these tasks, resulting in a set of discovered *task-level patterns*. In general, these approaches can provide better understanding of Web users' interests and preferences, which may lead to the improvement of recommendation accuracy or interpretability.

In this paper, we propose a Web recommendation system based on a maximum entropy model. Maximum Entropy model is a powerful statistical technique with solid theoretical foundations. It has been widely used in Natural Language Processing, Information Retrieval, Text Mining and other areas [16, 12, 1, 13]. Under maximum entropy principle, different sources of

knowledge can be combined by defining a set of feature functions representing various constraints on individual knowledge sources. In this paper, our maximum entropy recommendation system combines different levels of knowledge about Web users' navigational bahavior. Specifically, we combine page-level patterns from clickstream data with "task-level" patterns discovered in the Web usage mining framework based on Probabilistic Latent Semantic Analysis described above. The experimental results show that our recommendation system can achieve better accuracy and interpretability when compared to recommendations based on the standard first-order Markov models that are often used for making page predictions.

This paper is organized as follows. In Section 2, we present our recommendation method based on the maximum entropy model. In Section 3, we present our method of defining features associated with usage patterns from two levels of abstraction (i.e., page-level and task-level). We report our experimental results on two real data sets in Section 4. Finally, we conclude this paper and discuss some possible improvements to the model as future work.

2. Recommendations Based on Maximum Entropy

In this section, we present our recommendation system based on a maximum entropy model. It consists of two components. The offline component accepts constraints from the training data and estimates the model parameters. The online part reads an active user session and runs the recommendation algorithm to generate recommendations (a set of pages) for the user.

Generally, we begin Web usage analysis with the data preparation process. Raw Web log data is cleaned and sessionized to generate user sessions. Each user session is a logical representation of a user's single visit to the Web site (usually within certain time interval). After data preparation (see [5] for details), we have a set of user sessions $U = \{u_1, u_2, \cdots, u_m\}$, a set of pages $P = \{p_1, p_2, \cdots, p_n\}$. The Web session data can be conceptually viewed as a $m \times n$ session-page matrix $UP = [w(u_i, p_j)]_{m \times n}$, where $w(u_i, p_j)$ represents the weight of page p_j in user session u_i. The weights can be binary, indicating the existence or non-existence of the page in the session, or they may be a function of the occurences or duration of the page in that session. Each user session can also be represented as a sequence of pages, as $u_i = \langle p_i^1, p_i^2, \cdots, p_i^t \rangle$, where $p_i^j \in P$.

To make predictions or generate recommendations for active users, we aim to compute the conditional probability $Pr(p_d|H(u_i))$ of a page p_d being visited next given a user's recent navigational history $H(u_i)$, here $p_d \in P$, and $H(u_i)$ represents a set of recently visited pages by u_i. Since the most recent activities have greater influence on reflecting a user's present interests, we only use the last several pages to represent a user's history.

2.1. The Maximum Entropy Model

In our maximum entropy model, the training data is used to set constraints and train the model. A constraint represents certain characteristic of the training data which should be satisfied by the model. To implement constraints, we define one or more feature functions ("features" for short). For example, we define a feature as

$$f_{p_a,p_b}(H(u_i), p_d) = \begin{cases} 1 & \text{if user } u_i \text{ visited pages } p_a \\ & \text{and } p_b \text{ consecutively} \\ 0 & \text{otherwise} \end{cases}$$

Such a feature captures certain characteristic (i.e., visiting pages p_a and p_b consecutively) of user sessions in the training data. For example, from a user session $u_i = \langle p_2, p_3, p_5 \rangle$, we can derive a pair $\langle H(u_i), p_d \rangle$, where $H(u_i) = \{p_2, p_3\}$ and $p_d = p_5$. Suppose that we have defined a feature f_{p_3,p_5} as above. This feature will be evaluated to be 1 for session u_i. While another feature f_{p_2,p_5} will be evaluated to be 0 for this session, since p_2 and p_5 were not visited consecutively. We will discuss our method of identifying features in the next section. Based on each feature f_s, we represent a constraint as:

$$\begin{aligned} &\textstyle\sum_{u_i} \sum_{p_d \in P} Pr(p_d|H(u_i)) \bullet f_s(H(u_i), p_d) \\ &= \textstyle\sum_{u_i} f_s(H(u_i), D(H(u_i))) \end{aligned}$$

where $D(H(u_i))$ denotes the page following u_i's history in the training data. Every constraint restricts that the expected value of each feature w.r.t. the model distribution (LHS of the above equation) should always equal its observation value in the training data (RHS).

After we have defined a set of features, $F = \{f_1, f_2, \cdots, f_t\}$, and accordingly, generated constraints for each feature, it's guaranteed that, under all the constraints, a unique distribution exists with maximum entropy [10]. This distribution has the form:

$$Pr(p_d|H(u_i)) = \frac{exp(\sum_s \lambda_s f_s(H(u_i), p_d))}{Z(H(u_i))} \qquad (1)$$

where $Z(H(u_i)) = \sum_{p_d \in P} Pr(p_d|H(u_i))$ is the normalization constant ensuring that the distribution sums

to 1, and λs are the parameters needed to be estimated (see below). Thus, each source of knowledge can be represented as features (and constraints) with associated weights. By using Equation 1, all knowledge sources (represented by various features) are taken into account to make predictions about users' next action given their past navigational activity.

There have been several algorithms [6, 15, 9] which can be applied to estimate the λs in equation 1. These algorithms share a common idea: for each feature, we iteratively update paramteters λs in order to move the expected value and the observed value (w.r.t. this feature) closer, until they are euqal to each other. Of these algorithms, SCGIS has been shown to be more efficient especially when the number of outcomes is not very large. In our experiments, the outcomes will be the items (pages or products) to predict, with the total number in the order of several hundred. So we adopt this algorithm in our experiments. We also apply the smoothing technique suggested by Chan and Rosenfeld [4], which has shown to be able to improve the model by using some prior estimates for model parameters.

2.2. Generating Recommendations

After we have estimated the parameters associated with each feature, we can use Equation 1 to compute the probability of any unvisited page p_i being visited next given certain user's history, and then pick the pages with highest probabilities as recommendations. Given an active user u_a, we compute the conditional probability $Pr(p_i|u_a)$. Then we sort all pages based on the probabilities and pick the top N pages to get a recommendation set. The algorithm is as follows:

Input: an active user session u_a, parameters λs estimated from the training data.

Output: a list of N pages sorted by probability of being visited next given u_a.

1. Consider the last L pages of the active user session as a sliding window (these pages are considered as this user's history), and identify the dominant task using Equation 3 in Section 3. (L is the size of the sliding window, also the length of the user history.)

2. For each page p_i that does not appear in the active session, assume it is the next page to be visited, and evaluate all the features based on their definitions (in Section 3).

3. Using Equation 1 to compute $Pr(p_i|u_a)$.

4. Sort all the pages in descending order of $Pr(p_i|u_a)$ and pick the top N pages to get a recommendation set.

3. Identifying Features

In our model, we use two levels of knowledge about Web users' navigational behavior, namely page-level and task-level usage patterns. For each type of pattern we define features that capture the constraints we would like to specify on the pattern. (Here, we use binary features, which have been widely used in different fields [16, 12, 1, 13].)

3.1. Features Based on Page-level Usage Patterns

Page-level patterns capture Web users' common navigation paths with statistical significance. As we stated before, there have been many different data mining and machine learning techniques which can be applied to the users' navigational data to generate page-level usage patterns. For simplicity, here we adopt the first-order Markov model to generate page transitions. For each page transition $p_a \rightarrow p_b$ where $Pr(p_b|p_a) \neq 0$, we define a feature function as:

$$f_{p_a,p_b}(H(u_i), p_d) = \begin{cases} 1 & \text{if page } p_a \text{ is the last page} \\ & \text{in } H(u_i), \text{ and } p_b = p_d, \\ 0 & \text{otherwise} \end{cases}$$

3.2. Features Based on Task-Level Usage Patterns

3.2.1. Discovery of Task-level Patterns

Page-level usage patterns can be directly used for generating recommendations for new users. However, they provide no insight into users' underlying interests, thus limiting the accuracy of recommendations. To address this problem, some have proposed directly modeling Web users' hidden interests and preferences. Here we employ the approach proposed in [11] to identify task-level usage patterns.

Let us assume there exist a set of hidden (unobserved) variables $Z = \{z_1, z_2, ..., z_l\}$ corresponding to users' common interests (tasks). Each usage observation, which corresponds to an access by a user to a Web resource in a particular session which is represented as an entry of the $m \times n$ co-occurrence matrix UP, can be modeled as

$$Pr(u_i, p_j) = \sum_{k=1}^{l} Pr(z_k) \bullet Pr(u_i|z_k) \bullet Pr(p_j|z_k), \quad (2)$$

summing over all possible choices of z_k from which the observation could have been generated.

Now, in order to explain a set of usage observations (U, P), we need to estimate the parameters $Pr(z_k)$, $Pr(u_i|z_k)$, $Pr(p_j|z_k)$, while maximizing the log-likelihood of the observations.

We use the Expectation-Maximization (EM) algorithm [7] to perform maximum likelihood parameter estimation of $Pr(z_k)$, $Pr(u_i|z_k)$, $Pr(p_j|z_k)$. These probabilities quantitatively measure the relationships among users, Web pages, and common interests (tasks). For example, $Pr(p_j|z_k)$ represents the probability of page p_j being visited given a certain task z_k is pursued.

Recall that each user session can also be represented as a page sequence, $u_i = \langle p_i^1, p_i^2, \cdots, p_i^t \rangle$, $p_i^j \in P$. Using the relationships between pages and tasks, we employ the following algorithm to transform each user session into a sequence of tasks.

Input: a user session u_i, page-task associations $Pr(p_j|z_k)$.

Output: a user session represented as a task sequence.

1. Get the first L pages from u_i and put them into a sliding window W (with size of L, equals the length of a user's history $H(u_i)$). Using Bayesian updating [17], we compute the probability of each task given all the pages in the window W as

$$Pr(z|W) = \frac{Pr(W|z)Pr(z)}{Pr(W)} \propto Pr(z) \prod_{p \in W} Pr(p|z) \tag{3}$$

2. Pick those task(s) with probability exceeding a pre-defined threshold as the current task(s) and output them.

3. Move the sliding window to the right (remove the first page from the window, and add the next page in u_i into the window), recompute the probability of each task given the current sliding window, and pick the dominant task(s). Repeat this process until the sliding window reaches the end of this session.

After running this algorithm, each user session can be represented as a sequence of tasks, $u_i = \langle z_x, \cdots, z_y \rangle$, where z is either a single task or a small set of tasks. This representation gives us a direct understanding of users' interests and the temporal changes in these interests. Now we can apply data mining techniques to generate task-level patterns from these task sequences. For example, we can run simple first-order Markov model to compute probabilities of task transitions, $Pr(z_b|z_a)$.

3.2.2. Using the Discovered Task-Level Patterns as Features

Compared to page-level patterns, task-level patterns can capture Web users' underlying interests and their temporal changes, providing better understanding of users' diverse behaviors. Given a task transition $z_a \rightarrow z_b$, we define a feature as $f_{z_a,z_b}(H(u_i), p_d)$:

$$f_{z_a,z_b}(H(u_i), p_d) = \begin{cases} 1 & \text{if the dominant task of } H(u_i) \text{ is } z_a, \text{ and after moving sliding window right to include } p_d, \ z_b \text{ will be the dominant task,} \\ 0 & \text{otherwise} \end{cases}$$

4. Experimental Results

We use two real Web site data sets to empirically measure the accuracy of our recommendation system. Our accuracy metric is called *Hit Ratio* and is used in the context of top-N recommendation framework: For each user session in the test set, we take the first K pages as a representation of an active session to generate a top-N recommendations. We then compare the recommendations with page $K+1$ in the test session, with a match being considered a *hit*. We define the Hit Ratio as the total number of hits divided by the total number of user sessions in the test set. Note that the Hit Ratio increases as the value of N (number of recommendations) increases. Thus, in our experiments, we pay special attention to smaller number recommendations (between 1 and 10) that result in good hit ratios.

In our experiments, we use Web server log data from two Web sites. The first data set is based on the server log data from the host Computer Science department. The site is highly dynamic, involving numerous online applications, including admissions application, online advising, online registration, and faculty-specific Intranet applications. After data preprocessing, we identify 21,299 user sessions (U) and 692 pageviews (P), with each user session consisting of at least 6 pageviews. We choose 30 as the task number and set the length of user history to 4. This data set is referred to as the "CTI data." This second data set is based on *Network Chicago* which combines the programs and activities of the Chicago Public Television and Radio (*www.networkchicago.com*). Total of 4,987 user sessions and 295 Web pageviews were included for analysis. Each user session consists of at least 6 pageviews.

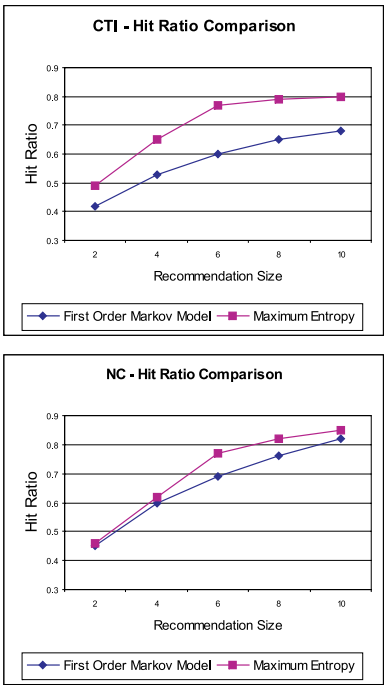

Figure 1. Comparison of Recommendation Accuracy: Maximum Entropy Model v. Markov Model.)

In contrast to the CTI data, this site is comprised primarily of static pages grouped together based on their association with specific content areas. In this case, we expect users' navigational behavior reflect their interests in one or more programs represented by the content areas. In the experiment, task number is set to 15 and the length of user history is set to 4. We refer to this data set as the "NC data."

We identify features as discussed in Section 3 and build our recommendation system according to the algorithm of Section 2. For comparison, we build another recommendation system based on the standard first-order Markov model to predict and recommend which page to visit next. The Markov-based system models each page as a state in a state diagram with each state transition labeled by the conditional probabilities estimated from the actual navigational data from the server log data. It generates recommendations based on the state transition probability of the last page in the active user session.

Figure 1 depicts the comparison of the *Hit Ratio* measures for the two recommender systems in each of the two data sets. The experiments show that the maximum entropy recommendation system has a clear overall advantage in terms of accuracy over the first-order Markov recommendation system on the CTI data. As for the NC data, our model also performs better in general, but the overall advantage is not as apparent as in the CTI data. One explanation of the difference in the results is that the CTI Web site provides many distinct functionalities for users. Users can collect admission information, submit online applications, perform degree audits, etc. While the content-based functionalities from NC site are not so distinct as those of CTI site. Therefore, the benefit of using tasks there is not so big.

The maximum entropy model captures not only users' immediate navigational interests reflected in the next visited page, but also the functional tasks in which the users are engaged at the given point in time. Therefore, besides the improvements in recommendation accuracy, using task-level patterns enables us to identify users' hidden interests and their temporal changes, which lead to better interpretability of recommendations.

5. Conclusions

In this paper, we propose a Web recommendation system based on a maximum entropy model. We show that different levels of knowledge about historical users' navigational behavior can be seamlessly combined together to generate recommendations for new users. The knowledge include the page-level statistics about users' historical navigation and the usage patterns about Web users' underlying interests and their temporal changes. Our experiment shows that our recommendation system can achieve better recommendation accuracy, while providing a better interpretation of Web users' diverse navigational behavior, when compared to the standard Markov model for page recommendations. In this paper, our intention has been to show the benefits of combining different levels of knowledge about Web users' navigational activity using the maximum entropy principle. Therefore, here we only use a simple first order Markov model to generate page-level usage patterns, while the PLSA model has been used to identify and characterize task-level patterns. We believe this model can be improved in these ways: using other state-of-art models such as Latent Dirichlet Allocation [2] to model users' underlying interests and tasks, and enhancing the basic maximum entropy model by applying effective feature selection methods [1, 10, 3], or adopting mixture of maximum entropy models [14].

References

[1] A. Berger, S. D. Pietra, and V. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1), 1996.

[2] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[3] S. Chen and R. Rosenfeld. Efficient sampling and feature selection in whole sentence maximum entropy language models. In *Proceedings of ICASSP-1999*, 1999.

[4] S. Chen and R. Rosenfeld. A guassian prior for smoothing maximum entropy models. Technical report, CMU, 1999.

[5] 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.

[6] J. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.

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

[8] R. Ghani and A. Fano. Building recommender systems using a knowledge base of product semantics. In *Proceedings of the Workshop on Recommendation and Personalization in E-Commerce, at the 2nd Int'l Conf. on Adaptive Hypermedia and Adaptive Web Based Systems*, Malaga, Spain, May 2002.

[9] J. Goodman. Sequential conditional generalized iterative scaling. In *Proceedings of NAACL-2002*, 2002.

[10] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, MA, 1998.

[11] X. Jin, Y. Zhou, and B. Mobasher. Web usage mining based on probabilistic latent semantic analysis. In *Proceedings of the Tenth ACM SIGKDD Conference(2004)*, 2004.

[12] K. Nigram, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *Proceedings of IJCAI-1999*, 1999.

[13] D. Pavlov and D. Pennock. A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In *Proceedings of Neural Information Processing Systems(2002)*, 2002.

[14] D. Pavlov, A. Popescul, D. Pennock, and L. Ungar. Mixtures of conditional maximum entropy models. Nec technical report, NEC Research Institute, 2002.

[15] S. D. Pietra, V. D. Pietra, and J. Lafferty. Inducing features of random fields. Technical report, CMU, 1995.

[16] R. Rosenfeld. Adaptive statistical language modeling: A maximum entropy approach. Phd dissertation, CMU, 1994.

[17] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, 2002.