Path Recommender System Based On HMM

ABSTRACT
The continuous growth in use of the World Wide Web is creating difficulties in both the design of web sites to suit a variety of different users. Most traditional methods to solve the problem only focus on the automatic recommendation of web pages. A new approach is tried in this paper to create link navigation models. Hidden Markov Model allows the system to dynamically model the URL access patterns that are observed in navigation logs. The most likely path is calculated by Viterbi algorithm. Recommendation of these paths to users will lead them to information that they need.

KEYWORDS
Hidden Markov Model, Web Usage Mining, Clustering

INTRODUCTION
The continuous growth in use of the World Wide Web is creating difficulties in both the design of web sites to suit a variety of different users. Automated search engines that rely on keyword matching can return the information the user need. However, it cannot offer different service to the individual users. In order to solve these problems, the computer community began to study the recommender system based on web.

Recommender systems apply statistical or knowledge discovery techniques to the problem of making product recommendations during one customer interaction. The ability to track customs’ purchase behavior down to individual mouse clicks can make the vendor recommend the goods by sending these pages describe the goods to customs. Recommmender system can be used to improve theadaptive web site in many ways, such as the improvement of web site’s organization by add the link in pages that seem to be related, individual service of web sites achieved by recommending the pages that the users desired. The traditional recommender systems are based on the technique of web usage mining, such as clustering, association rule, etc.

In this paper, we introduce a new recommender system based on HMM. It predicts and recommends the most likely path that users will follow to find their desired information. We highlight links on the path instead of adding links into web pages in other recommender system.

RELATED WORK
Most recommender system is based on Collaborative filtering. In collaborative filtering, users that tend to give similar ratings to similar objects are presumed to have similar tastes; when a user seeks recommendations of new objects, the site suggests those objects that were highly rated by other users with similar tastes. Others recommender systems exploit the similarity among web pages. When one web page is required by one user, other web pages with high similarity will be recommend to user directly. Markov chains are used in [7] to represent the link navigation model. The tour generator module of this paper can automatically generate the most likely page sequence when the starting URL is input based on the Markov model.
SYSTEM DESIGN

Our survey of other work has led us to make some consideration about a recommender system. The navigation behavior of user reflects the interest of user. Which path user will select not only bases on the organization of web site, but also depends on the background of user, such as the knowledge, the preference. So, we should take the individual or many users’ character into account. For example, Web site of Zhejiang University supports the service of electing the curriculum. The introduction of every departmental major is placed under the catalogue of the department. If the curriculum is the departmental major of two departments, the links to the curriculum page will be added to two departments’ pages. When student of other department wants to elect this curriculum, they must firstly reach the page of each department above. A student of the department of information engineering, named I, and a student of the department of mechanical engineering, named M, both want to find the introduction about the curriculum, the computer control technique, which is the departmental major of two departments: computer science and control science. There exists links to this curriculum in both pages of two departments. To student I, since he is familiar with the computer science, he mostly accesses the curriculum page through page of computer science. However, it’s different to M. He commonly accesses the page through page of control science because of their characters. We want to recommend the path to students who want to know the computer control technique. The path through computer science should be recommended to I. If the other path is highlight, the students maybe don’t follow this path because he isn’t familiar with control department. The different result will be achieved to student M.

Hidden Markov Model can be used to find the most likely state sequence to obtain the observation. So, we applied it to find the most likely path leading to particular information.

Fig.1 is the architecture of the Path Recommender System. There are four components to construct this system: The output of Data Pre-Processing is the record of transaction data denoting the user behavior. HMM is the hardcore of our design. Traditional approaches of web usage mining, such as Clustering, Association Rules are used here to find the similar pages. The finally component calculates the most likely path from one particular page to the page including useful information by Viterbi algorithm and generates the path that will be recommended to users.

In the following section, every component of our system will be described in detail. Considering the importance of hidden Markov model, it is introduced first.

HIDDEN MARKOV MODEL

Hidden Markov Model is a finite set of states, each of which is associated with a probability distribution. Transitions
among the states are governed by a set of probabilities called *transition probabilities*. In a particular state an outcome or *observation* can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are “hidden” to the outside; hence the name Hidden Markov Model. HMM with discrete probability distributions is used in our case. In order to define an HMM completely, following elements are needed.

- The number of states of the model, $N$.
- A set of observation sequence $O = o_1, o_2, \ldots, o_l$.
- The number of observation symbols in the alphabet, $M$.
- A set of states $Q$. $q_i \in Q$ denotes the current state.
- A set of state transition probabilities $\Lambda = \{a_{ij}\}$ where $a_{ij} = p\{q_{i+1} = j \mid q_i = i\}, \qquad 1 \leq i, j \leq N$
- Transition probabilities should satisfy the normal stochastic constraints, $a_{ij} \geq 0, \qquad 1 \leq i, j \leq N$ and $\sum_{j=1}^{N} a_{ij} = 1, \quad 1 \leq i \leq N$
- A probability distribution in each of the states, $B = \{b_j(k)\}$.
- $b_j(k) = p\{o_l = v_k \mid q_i = j\}, \quad 1 \leq j \leq N, \quad 1 \leq k \leq M$ where $v_k$ denotes the $k^{th}$ observation symbol in the alphabet, and $O_l$ the current parameter vector.
- Following stochastic constraints must be satisfied.
  \[ b_j(k) \geq 0, \quad 1 \leq j \leq N, \quad 1 \leq k \leq M \quad \text{and} \quad \sum_{k=1}^{M} b_j(k) = 1, \quad 1 \leq j \leq N \]
- The initial state distribution, $\pi = \{\pi_i\}$, where, $\pi_i = p\{q_1 = i\}, \quad 1 \leq i \leq N$

We can use the compact notation $\lambda = (\Lambda, B, \pi)$ to denote an HMM with discrete probability distributions. Thus, the probability of a given sequence $O = o_1, o_2, \ldots, o_l$ being observed is the sum of the probability of all the paths that it have passed through.

\[ P(O \mid \lambda) = \sum_{q_1, \ldots, q_l} \prod_{i=1}^{l} P(q_{i+1} \rightarrow q_i) P(o_i \mid q_i) \]

In summary, we can find the most likely state sequence for a given sequence of observations, $O = o_1, o_2, \ldots, o_l$ and a model $\lambda = (\Lambda, B, \pi)$ by

\[ V(O \mid \lambda) = \arg \max_{q_1, \ldots, q_l} \prod_{i=1}^{l} P(q_{i+1} \rightarrow q_i) P(o_i \mid q_i) \quad \text{(formula 1)} \]

The general HMM is described above. The following is the definition of our HMM. Firstly, our web site is equivalent to a directed graph $G$ in which nodes represent pages and edges represent hyperlinks. Web pages are classified as navigation pages or content pages. User’s navigation behavior can be described as follows. When user wants to search
given information, he will hit the links until he reaches content page. If he needs information in this page, he will read it in detail, otherwise, he will begin his new navigation behavior. Then, navigation path of the user can be described as a HMM. Following are definitions:

- The node denoting the web page is a state $q \in Q$.
- The information included in the content page is the observation symbols $v_k$. The number of observation symbols is equal to the number of content pages.
- The information is required by users in a period of time is the observation sequence $O = o_1, o_2, \ldots, o_t$.
- The connective nodes $q_i, q_j$ have transition probabilities $a_{ij}$.
- To each node $q$, denoting either navigation page or content page, probability that users require $o_i = v_k$ through $q$ can be described as $B = \{b_q(k)\}$.

Based on the model above, we can find the most likely state sequence, namely the navigation path sequence to the given information that users need.

**IMPLEMENTATION**

**Data Pre-Processing**

Generally, there are many files accessed as a result of a request by a client to view a web page. Thus, the process of removing irrelevant entries from logs is referred to as *data cleaning*. After data cleaning, we only have one record of one page. The record consist of four parts: user’s IP address, user host name, access begin time, the URL of web pages. The second step is *transaction identification*. This process divides a large session into multiple small meaningful parts. A reasonable assumption seems to be that a given user treats each page in one of two ways, either for navigation purposes to find links to desired data, or for actual information. Due to the difference of the web page’s attribute, there will be several transaction modules. In this paper we use the maximal forward reference transaction identification module illustrated in [2].

Finally, we format the result of *transaction identification*. Navigation behavior can be described as follows: In a transaction, one user wants to find the particular information from current page. When he arrives at one content page, he may desire information of this page or not. If he needs this information, he will navigate this page in detail, otherwise he will not navigate the page and begin new transaction to find information again. So whether or not he need one content page is based on the time he spend on this page. So, we define a threshold to determine the page is useful or not. The result of data pre-processing is the formatted transaction records. Table 1 illustrated transaction record of following HMM.
Table 1 The field and Description of transaction record

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transaction_ID</td>
<td>Transaction Identification</td>
</tr>
<tr>
<td>Info_Content</td>
<td>The information included in content page that user desired in current record (If the user didn’t navigate this page, this field will be null)</td>
</tr>
<tr>
<td>AccessPath</td>
<td>The paths that user went through for the information of Info_Content</td>
</tr>
</tbody>
</table>

The Probability of Hidden Markov model

Before the Viterbi is applied, we should calculate the transition probabilities \( a_{ij} \) and probability distribution in each of the states \( b_j(k) \). The maximum likelihood principle is applied here to estimate the \( a_{ij} \) and \( b_j(k) \).

a). \( a_{ij} = p(q_{t+1} = j | q_t = i) \) can be estimated as follows:

\[
a_{ij} = \frac{C(q_t, q_{t+1})}{C(q_t)}
\]

Where \( C(q_t, q_{t+1}) \) is the number of times \( q_{t+1} \) follows \( q_t \) in the transaction and \( C(q_t) \) is the number of \( q_t \) in the transaction. In other words, it represents the probability of transition from one page \( q_t \) to the other page \( q_{t+1} \).

b). \( b_j(k) = p(o_k = v_q | q_t = j) \) can be estimated by

\[
b_j(k) = \frac{C(q_t, o_k)}{C(q_t)}
\]

Where \( C(q_t, o_k) \) denotes the number of times that \( q_t \) and \( o_k \) occur in the record of transaction concurrently and \( C(q_t) \) denotes the number of \( q_t \) in transaction. In other words, it represents the probability that user needs the information \( o_k \) when he is navigating page \( q_t \). \( a_{ij} \) and \( b_j(k) \) can be calculated based on the transaction records.

Traditional Algorithm of Web Usage Mining

In this paper, we only describe clustering algorithm used in following experiments. Page clustering collects web pages having high similarities.

Firstly, Similarity of web pages \( p_1 \) and \( p_2 \) is defined:

**Definition 2:** \( P(p_1 | p_2) \) represents the number of the user request \( p_2 \) before he visits \( p_1 \) in a transaction.
Similarity of web pages \( p_1, p_2 \) can be defined as 
\[
SIM(p_1, p_2) = \frac{P(p_1 \mid p_2) + P(p_2 \mid p_1)}{2}
\]
(formula 2)

Based on formula 2, the similarity matrix can be created for all pages having been visited by user. K-means clustering algorithm is used to cluster these pages, and finally web pages can be classified into several clustering sets: 
\[
C = \{c_1, c_2, \ldots, c_n\}. \quad (c_j \text{ are the sets of pages}).
\]

**Tour Generator**

Given a start web page, the tour generator outputs a sequence of states, namely web pages, which is the most likely sequence that can obtain the information included in content page. The Viterbi algorithms is used here to find the most likely sequence.

The final problem is how to recommend these likely sequences to user. In our case, the color of links is highlighted to indicate it lead to the similar pages.

**EXPERIMENTS**

In order to evaluate our system, we construct a simple web site as example.

\[
\begin{align*}
&\text{Fig 2. The architecture of one simple web site} \\
&\text{In our case, the nodes \{a, b, c, d, e, f, g, h, i, j\} denote the navigation pages, the others \{k, l, m, n, o, p\} denote the content pages. We select the web server logs in one hour. The clustering algorithm is applied based on the user access records.} \\
&\text{\(a_j\) and \(b_j(k)\) are also calculated based on the transaction records. Finally, there are three clusters: \{p, n, m\}, \{o, m\}, \{k, l\}. Note that because our clustering algorithms is based on how often pages co-occur in user visits, the cluster may overlap each other. The illustration in detail of clustering can be found in [6]. The \(a_j\) and \(b_j(k)\) are predicted in Fig 3 and Fig 4. The capital letter represent the information in the content page that is represented by the corresponding letter. For example, the “N” display the information included in the content page “n”.
\end{align*}
\]
We illustrate our advantages by comparing with Markov model without the definition of $b_j(k)$. We calculate the most likely path from p to N in this case. We calculate the most likely path from p leading to information N here. Both the path1 {p, c, e, i, n} and path2 {p, c, f, i, n} are the likely sequence to the observation N. HMM are used first to obtain the probability:

**Path1:**

$$a(c \mid p) \times b_j(N) \times a(e \mid c) \times b_j(N) \times a(i \mid e) \times b_j(N) \times a(n \mid i) \times b_j(N) = 0.0006$$

**Path2:**

$$a(c \mid p) \times b_j(N) \times a(f \mid c) \times b_j(N) \times a(i \mid f) \times b_j(N) \times a(n \mid i) \times b_j(N) = 0.0001$$

Evidently, the path1 is prior to path2. However, if we use the Markov model, the result will be

**Path1:** $a(c \mid p) \times a(e \mid c) \times a(i \mid e) \times a(n \mid i) = 0.015$

**Path2:** $a(c \mid p) \times a(f \mid c) \times a(i \mid f) \times a(n \mid i) = 0.018$

Evidently, the different result is achieved. The Path2 has the higher probability than Path1. This result lies in the reason that we take no account of the probability $b_j(k)$. We can explain this result in following. There maybe many users pass through the whole or the partial path {p, c, f, i, n}, but they needn’t the information N included in the page n. Thus,
it’s possible that Path2 isn’t the likely path that user usually pass through to navigate the information N. So, the Markov model takes no account of the probability that user need the information N when he currently accesses page p, c, f, i or n. Web site of curriculum is the perfect example. Maybe there are many students of department of information engineering access the web page of department of control science, but they didn’t require the information of the curriculum of the computer control technique. Thus, it’s no use to recommend the path through the department of control science because the students don’t believe curriculum of the computer control technique is placed under the page of the department of control science. If the other path through the department of computer science is recommended, many students of the department of information engineering will follow the path to get the information of the curriculum of the computer control technique. On the other hand, if our user is the student of Mechanical engineering, the different result will be achieved by HMM. Since we do this experiment in the Intranet of Zhejiang University and IP address can provide user’s information, we needn’t develop other methods to obtain it.

CONCLUSION

In this paper, we have presented a new recommender system based on HMM. It generates the most likely path that can lead the user to reach desired information. A comparison with Markov model illustrates the validity of our design. Additionally, traditional recommender system adds the links between pages, which made the web site low efficiency. Our system only recommends the path to user, and it needn’t add other links to web page. In addition, current system is designed for all users, not for individual user. In ongoing work, we are improving the system for individual user to achieve personalization of web site.

REFERENCE

1. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl Analysis of Recommendation Algorithms for E-commerce