Data Mining: Patterns of User Navigation

Kirtika Goel, Bhairvee Singh

Abstract: We propose a data mining model that captures the user navigation behavior patterns. The user navigation sessions are modeled as a hypertext probabilistic grammar whose higher probability strings correspond to the user's preferred trails. An algorithm to anciently mine such trails is given. We make use of the N gram model which assumes that the last N pages browsed affect the probability of the next page to be visited. The model is based on the theory of probabilistic grammars providing it with a sound theoretical foundation for future enhancements. Moreover, we propose the use of entropy as an estimator of the gram-mar's statistical properties. Extensive experiments were conducted and the results show that the algorithm runs in linear time, the grammar's entropy is a good estimator of the number of mined trails and the real data rules confirm the effectiveness of the model.

I. INTRODUCTION

Data Mining and Knowledge Discovery is an active research discipline involving the study of techniques which search for patterns in large collections of data. Meanwhile, the explosive growth of the World Wide Web (known as the web) in recent years has turned it into the largest source of available online data. Therefore, the application of data mining techniques to the web, called web data mining, was the natural subsequent step and it is now the focus of an increasing number of researchers.

In web data mining there are currently three main research directions: (i) mining for information, (ii) mining the web link structure, and (iii) mining for user behavior patterns. Mining for information focuses on the development of techniques to assist users in processing the large amounts of data they face during navigation and to help them and the information they are looking for, see for example. Mining the link structure aims at developing techniques to take ad-vantage of the collective judgment of web page quality in the form of hyperlinks, which can be viewed as a mechanism of implicit endorsement, see. The aim is to identify for a given subject the authoritative and the hub pages. Authoritative pages are those which were conferred authority by the existing links to it, and hubs are pages that contain a collection of links to related authorities. Finally, the other research direction, which is being followed by an increasing number of researchers, is mining for user navigation patterns. This research focuses on techniques which study the user behavior when navigating within a web site.

In this paper we propose a new model for handling the problem of mining log data which directly captures the semantics of the user navigation sessions. We model the user navigation records, inferred from log data, as a hypertext probabilistic grammar whose higher probability generated strings correspond to the user's preferred trails. Therefore, a compact and self contained model of user interaction with the web is provided. There are two contexts in which such model is potentially useful. On the one hand, it can help the service provider to understand the users needs and as a result improve the quality of its service. The quality of service can be improved by providing adaptive pages suited to the individual user, by building dynamic pages in advance to reduce waiting time, or by providing a speculative service which sends, in addition to the requested document, a number of other documents that are expected to be requested in the near future. On the other hand, such a model can be useful to the individual web user by acting as a personal assistant integrated with his/her web browser.
II. HYPERTEXT PROBABILISTIC GRAMMARS

A log file can be seen as a per-user ordered set of web page requests from which it is possible to infer the user navigation sessions.

The user navigation sessions inferred from the log data are modeled as a hypertext probabilistic language generated by a hypertext probabilistic grammar which is a proper subclass of probabilistic regular grammars. A HPG is a probabilistic regular grammar which has a one-to-one mapping between the set of non-terminal symbols and the set of terminal symbols. Each non-terminal symbol corresponds to a web page and a production rule corresponds to a link between pages. Moreover, there are two additional states, S and F, which represent the start and finish states of the navigation sessions.

The probability of a production from the start state is proportional to the number of times the corresponding state was visited, implying that the destination node of a production with higher probability corresponds to a state that was visited more often. We define $f$ as a parameter that attaches the desired weight to a state being the first in a user navigation session. If $f = 0$ only states which were the first in a session have probability greater than zero of being in a production from the start state, on the other hand if $f = 1$ all state visits are given proportionate weight. Note that when $f > 0$ every grammar state has an initial probability greater than zero. The probabilities of the productions from the start state correspond to the vector of initial probabilities, $f$, and the probabilities of the other productions correspond to the transition matrix of a Markov chain.

In the example shown in Figure 1 we have 6 user sessions with a total of 24 page requests, wherein state A1 was visited 4 times, 2 of which are the first state in a user session, therefore, since $f = 0.5$ we have $f(A1) = 0.24^{4} + 0.5^{2} = 0.25$. Figure 2 shows the grammar inferred from the given set of trails for $N = 1$ and $f = 0.5$. (We freely utilize in our figures the duality between grammars and automata.)

<table>
<thead>
<tr>
<th>Sess</th>
<th>User trail</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A1 ! A2 ! A3 ! A</td>
</tr>
<tr>
<td>2</td>
<td>A1 ! A5 ! A3 ! A</td>
</tr>
<tr>
<td>3</td>
<td>A5 ! A2 ! A4 ! A</td>
</tr>
<tr>
<td>4</td>
<td>A5 ! A2 ! A3</td>
</tr>
<tr>
<td>5</td>
<td>A5 ! A2 ! A3 ! A</td>
</tr>
<tr>
<td>6</td>
<td>A4 ! A1 ! A5 ! A</td>
</tr>
</tbody>
</table>

Fig. 1. An example set of user's trails.

2.1 The $N$-gram Model

We use the N gram concept [6] to determine the assumed user memory when navigating within the site, where $N = f 1$, is called the history depth. Therefore, when the user is visiting a page it is assumed that only the $N$ previously visited pages influence the link he will choose to follow next. The intuition is that the user has a limited memory of the previously browsed pages and that the next choice depends only on the last $N$ pages browsed. In an N gram model each of the states of the HPG corresponds to a sequence of $N$ pages visited.

The drawback of the N gram model is the increase in the number of states as the history depth
increases. For a web site with \( n \) pages and for a given history depth \( N \) the expected number of grammar states is \( n:b^{(N-1)} \), where \( b \) is the average number of out links in a page. As such, there is a clear trade-off between the modeled user memory (measured by the history depth) and the grammar complexity (measured in the number of states).

Finally, the hypertext grammar model is incremental in the sense that when more log data becomes available it can be incorporated in the model without the need of rebuilding the grammar from scratch. Hence, the HPG is a compact way of keeping the browsing history since the grammar size depends only on the number of pages visited.

Algorithm 1
(Build Grammar \((T, N)\))

1. begin
2. for \( i = 1 \) to \( jTj \)
3. for \( k = 1 \) to \( N \) do
4. \( \text{State}[k] = T^i[k] \) end for
5. \( \text{StateSet: update(State)} \)
6. for \( j = N + 1 \) to \( jTj \)
7. \( \text{LeftState} = \text{State} \)
8. \( \text{State: shiftLeft() } \)
9. \( \text{State}[N] = T^i[j] \)
10. \( \text{StateSet: update(State)} \)
11. \( \text{Productions: update(LeftStateState)} \)
12. end for
13. end for
14. end.

2.2 The Entropy of a HPG
We propose to use the entropy of the HPG as an estimator of the statistical properties of the language generated by the grammar. The entropy is a measure of the uncertainty in the outcome of a random variable and in this context the sample space is the set of all strings generated by the grammar. If we had no information at all about the user interaction with the web the most rational thing would be to
assume that all pages had the same probability of being visited and all its links have the same probability of being chosen. The entropy is maximum in this case and there is no point in looking for patterns or trying to predict a random behavior. On the other hand, if the entropy is close to zero the user behavior has few uncertainties, and a small set of short rules should contain enough information to characterize his/her behavior. In conclusion, the intuition behind the estimator is that if the entropy of a probabilistic grammar is close to zero there should be a small set of strings with high probability and if the entropy is high then there should be a large number of strings with similar and low probability.

Assuming a transition with probability one from state F to S a HPG corresponds to an irreducible and a periodic Markov chain with a stationary distribution vector \( \mathbf{f} \) and transition matrix \( A \). Thus, we can estimate the entropy with the following expression:

\[
H = H(\mathbf{f}) + \sum_{i,j} (\mathbf{f}_i A_{ij} \log A_{ij})
\]

where \( H(\mathbf{f}) \) is included to take into account the randomness of the choice of the initial page, see [10] for detail on the entropy of a Markov chain. Note that we use the vector of initial probabilities \( \mathbf{f} \) as an estimator of the stationary vector \( \mathbf{f} \), since it is proportional to the number of times each state was visited. The value of \( H \) can be normalized to be in the range between 0 and 1 by considering its ratio with the corresponding random grammar. The random grammar consists in a grammar with the same structure but in which all states have their out-links probabilities according to a uniform distribution.

Such a measure which is an estimator of the statistical distribution of the grammar string probabilities can be useful in helping the user in the specification of the support and confidence thresholds.

### III. EXPERIMENTAL EVALUATION

To assess the performance and the effectiveness of the proposed model experiments were conducted with both random and real data. Tests with random data provide the means of evaluating many different topologies and configurations of a HPG and tests with real data allow us to verify whether or not the model is potentially useful in practice.

The method used to create the random data consisted of four consecutive steps: (i) given the required number of states and the average branching factor randomly create a directed graph, (ii) for each grammar state assign outgoing links weights according with the chosen probability distribution, (iii) verify if the resulting grammar has the required properties, that is, if every state has a path to F and if not add a link to F, (iv) normalize the grammar's weights, that is, calculate the production probabilities.

#### 3.1 Performance

The first objective of the experiments was to evaluate the algorithm performance and its scalability. The experiments were conducted for several model configurations where the grammar's size, \( n \), varied between 100 and 4000 states, the confidence threshold varied between 0:1 and 0:5. Note that the grammar size depends only on the number of pages in the hypertext system and not on the number of user sessions in the log data. In order to have a measure of support which allows us to compare the results for grammars with different sizes we have decided to define support in a way that takes into account the number of states \( n \), that is \( f = x_n \) where \( x \) is an integer value.

#### 3.2 Entropy

In order to verify the utility of the grammar's entropy as an estimator of the statistical properties of the grammar's language we calculated for each grammar size and confidence threshold the correlation between: (i) the entropy and the number of rules, (ii) the entropy and the number of iterations, and (iii)
the entropy and the average rule length. Moreover, to assess the effect that both the confidence and support thresholds have on the grammar's entropy we decided to also measure the entropy of a grammar inferred from the set of mined rules, which we call the posterior grammar. The posterior grammar is no more than a HPG whose initial input trails are a set of rules mined from its parent grammar. We define PER as the entropy rate of the posterior grammar, and ER is the entropy of the parent grammar but using the vector of initial probabilities of the posterior grammar as the estimator for the stationary vector in the parent grammar.

3.3 Ngram Model
The first order Markov chain provides an approximate model of user navigation patterns which can be improved by the N gram model noting that as we increase N the entropy decreases. One question that arises when using the N gram model is whether there is a method to determine the order of the Markov chain that best models a given set of user navigation sessions. As was stated in Section 2.1 there is a tradeoff between the model complexity (measured by its number of states) and its accuracy. Moreover, if the order of the chain is too high the model is uninteresting due to the fact that user navigation sessions within a web site are typically short on average and also because the probability of a very long trail being repeated in the future is not very high.

IV. RELATED WORK
Some authors have recently proposed the user of a first order Markov model for predicting user requests on the WWW. In a first order Markov model is proposed to implement a perfecting service to reduce server load. The model is build from past usage information and the transition probabilities between pages are proportional to the number of times both pages were accessed in a predefined time window. The experiment results show the method to be effective in reducing both the server load and the service time. A method is proposed in wherein a dependency graph is inferred and dynamically updated as the server receives requests. There is a node for every requested page and an arc between two nodes exists if the target node was requested within x accesses after the source node, the arc's weight is proportional to the number of such requests. The simulations performed with log data show that a reduction in the retrieval latency can be achieved. In the authors present a study of the quality of a kth order Markov approximation as a model of predicting user surfing patterns. Using information theoretic measures they conclude that the best compromise between the model accuracy and its complexity occurs with the first order Markov model. Moreover, they show that the model probabilities are more stable over time for lower orders models.

V. HEURISTICS FOR MINING HIGH QUALITY PATTERNS
We are currently working on the specification of new algorithms to improve the quality of the results relative to the deterministic DFS used in the experiments reported herein. In fact, as was shown in section 3.1, the exhaustive computation of all grammar strings with probability above the cut-point has the drawback of potentially returning a very large number of rules for small values of the cut-point. Note that if the user wants to find longer rules the threshold needs to be set low. This fact led us to the study of heuristics which allow us to compute a subset of the rule set while being able to control both its size and quality.

5.1 The Iterative Deepening Fine Grained Heuristic
Our first approach is based on the iterative deepening concept wherein the rule set is incrementally augmented until it is close enough to the desired set. We call this method the iterative deepening fine
grained approach. A measure of the distance between the currently explored and the final rule set is provided by the error, and by setting its desired value the user has control over the number of rules to be returned. Informally, the error is defined as the amount of probability left to explore, and it can be estimated by the sum of the probabilities of the trails that are not yet final. Moreover, a criterion is provided to decide which trails to augment at each stage in order to maintain a high quality rule set.

5.2 The Inverse Fisheye Heuristic
The second approach aims at providing the user with a method that gives a relatively small set of long rules. We propose the use of a dynamic threshold which imposes a very strict criterion for small rules and becomes more permissible when the trails get longer. We call this method inverse fisheye and with it, the early stages of the exploration are performed with the threshold value set high implying that only the best trails will pass the evaluation. In the subsequent stages the threshold has its value progressively reduced according to one of the available criterions allowing the trails to be further explored. Moreover, the user has to specify the maximum exploration depth since there is no guarantee that the algorithm terminates, especially when the cut-point decreases at a higher rate than the trail probability.

VI. CONCLUDING REMARKS
We have proposed a model of hypertext to capture user preferences when navigating through the web. We claim that our model has the advantage of being compact, self contained, coherent, and based on the well established work in probabilistic grammars providing it with a sound foundation for future enhancements including the study of its statistical properties. In fact the size of the model depends only on the size of the web site being analyzed and not on the amount of data collected.

The set of user navigation sessions is modeled as a hypertext probabilistic grammar, and the set of strings which are generated with higher probability correspond to the navigation trails preferred by the user. An algorithm to efficiently mine these strings is given. Extensive experiments with both real and random data were conducted and the results show that, in practice, the algorithm runs in linear time in the size of the grammar. Moreover, the entropy of the posterior grammar is shown to be a good estimator of the number of rules output from our algorithm, and the experiments with real data confirm the effectiveness of our algorithm. Our model has potential use both in helping the web site designer to understand the preferences of the site’s visitors and in helping individual users to better understand their own navigation patterns and increase their knowledge of the web's content.

REFERENCE
3. Data Mining of User Navigation Patterns (2000) by Jose Borges , Mark Levene
4. Frequent Pattern Mining in Web Log Data Renáta Iváncsy, István Vajk Department of Automation and Applied Informatics, and HAS-BUTE Control Research Group Budapest University of Technology and Economics Goldmann Gy. tér 3, H-1111 Budapest, Hungary
5. A FRAMEWORK FOR DATA MINING USER NAVIGATION PATTERNS by V. Vijalakshmi et al.