Selective Markov Models for Predicting Web Page Accesses

MUKUND DESHPANDE and GEORGE KARYPIS
University of Minnesota

The problem of predicting a user's behavior on a Web site has gained importance due to the rapid growth of the World Wide Web and the need to personalize and influence a user's browsing experience. Markov models and their variations have been found to be well suited for addressing this problem. Of the different variations of Markov models, it is generally found that higher-order Markov models display high predictive accuracies on Web sessions that they can predict. However, higher-order models are also extremely complex due to their large number of states, which increases their space and run-time requirements. In this article, we present different techniques for intelligently selecting parts of different order Markov models so that the resulting model has a reduced state complexity, while maintaining a high predictive accuracy.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*data mining*; I.2.6 [**Artificial Intelligence**]: Learning—*concept learning*

General Terms: Algorithms

Additional Key Words and Phrases: World wide web, web mining, Markov models, predicting user behavior

1. INTRODUCTION

In recent years, the problem of modeling and predicting a user's browsing behavior on a Web site has attracted a lot of research interest as it can be used to improve the Web cache performance [Schechter et al. 1998; Bestravos 1995; Padmanabham and Mogul 1996], recommend related pages [Dean and Henzinger 1999; Pirolli et al. 1996], improve search engines [Brin and Page 1998], understand and influence buying patterns [Chi et al. 1998], and personalize the browsing experience [Pitkow and Pirolli 1999]. The significance of this problem is evidenced by the fact that at the SIGKDD 2000 Conference [Kohavi and Brodley 2000], the problem of predicting and understanding a

This work was supported by NSF CCR-9972519, EIA-9986042, ACI-9982274, ACI-0133464, NASA NCC 21231, and Army High Performance Computing Research Center contract number DAAD19-01-2-0014.

Authors' addresses: M. Deshpande (deshpand@cs.umn.edu); G. Karypis (karypis@cs.umn.edu).

user's browsing and purchasing behavior was the topic of the KDDCup 2000 Competition.

Markov models [Papoulis 1991] have been used for studying and understanding stochastic processes and shown to be well suited for modeling and predicting a user's browsing behavior on a Web site. In general, the input for these problems is the sequence of Web pages accessed by a user and the goal is to build Markov models that can be used to predict the Web page that the user will most likely access next.

In many applications, lower-order Markov models are not very accurate in predicting the user's browsing behavior, since these models do not look far into the past to correctly discriminate the different observed patterns. As a result, higher-order models are often used. Unfortunately, these higher-order models have a number of limitations associated with high state-space complexity, reduced coverage, and sometimes, even worse, overall prediction accuracy. One simple method to overcome some of these problems is to train varying order Markov models and use all of them during the prediction phase, as is done in the All-Kth-Order Markov model proposed in Pitkow and Pirolli [1999]. However, this approach further exacerbates the problem of state-space complexity. To address this problem, an alternate approach was developed by Pitkow and Pirolli [1999] that identifies patterns of frequent accesses and uses them for prediction. Unfortunately, even though this approach was able to reduce the state-space complexity by up to an order of magnitude, it also reduced the prediction accuracy of the resulting models.

In this article, we present techniques for intelligently combining different order Markov models so that the resulting model has a low state-space complexity and, at the same time, retains the coverage and the accuracy of the All-Kth-Order Markov models. The key idea behind our techniques is that many of the states of the different order Markov models can be eliminated without affecting the performance of the overall scheme. In particular, we present three schemes for pruning the states of the All-Kth-Order Markov model, called (i) frequency pruning, (ii) confidence pruning and (iii) error pruning. Our experiments on four different datasets have shown that the proposed pruning schemes consistently outperform the All-Kth-Order Markov model and other single-order Markov models. For many problems, our schemes prune up to 94% of the states from the All-Kth-Order Markov model, while achieving comparable or better prediction accuracies than the All-Kth-Order Markov model.

Even though our algorithms were developed in the context of Web usage data, we have successfully used these techniques for prediction in different applications as well. For example, these models were used to predict the next-command-typed by the user on a word processor based on his/her past sequence of commands and for predicting the alarm state of telephone switches based on its past states. These applications will be discussed in detail in Section 4.1.

The rest of this article is organized as follows. Section 2 presents an overview of Markov models, followed by a brief overview of the problem of predicting a user's browsing behavior. Section 3 presents a detailed description of our selective Markov models. Section 4 provides a detailed experimental evaluation of our algorithms on a variety of datasets. Section 5 describes related research in this area. Finally, Section 6 offers some concluding remarks.

2. MARKOV MODELS FOR PREDICTING NEXT-ACCESSED PAGE

The act of a user browsing a Web site is commonly modeled by observing the set of pages that he or she visits [Srivastava et al. 2000]. This set of pages is referred to as a *Web session* ($\mathcal{W}$),[1] and is represented by the *sequence* of pages $\mathcal{W} = \langle \mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_l \rangle$, that were accessed. In this sequence, $\mathcal{P}_1$ represents the first page that was accessed, $\mathcal{P}_2$ the second, and so on. Note that $\mathcal{P}_i$ (uppercase) is a random variable representing the ith page in the user's Web session, whereas the actual realization of this random variable, the Web page in a user's Web session, will be represented by p_i(lower-case). Given such a Web session, the *next-page prediction problem* is that of predicting the Web page that will be accessed by the user next [Pitkow and Pirolli 1999]. That is, given $\mathcal{W}$, predict the next page $\mathcal{P}_{l+1}$ of the user's Web session.

This formulation can be used to solve many prediction problems that often arise in the Web and e-commerce domain, such as whether or not a user will further explore the information associated with a particular topic, buy a particular product, view certain advertisements, or leave the Web site—providing valuable clues about the user's interests and behavioral patterns. For example, we can solve the problem of predicting whether a user has the intention to buy a given product or not by predicting if that user will visit the *add-to-shopping-cart* page immediately after visiting the *product description* page of that product. Such formulations are possible because most pages in a Web site have a certain meaning associated with them and a user, by visiting a particular page, indicates his or her interest in that page. For example, e-commerce sites contain pages related to the products they sell (e.g., product description/specification pages, customer ratings and reviews, comparative pricing, etc.), pages related to order processing (e.g., shopping card management, payment information, etc.), and pages related to various policies (e.g., return/exchange policy, privacy policy, etc.). Being able to predict the potential interests (or disinterests) of a user while he or she is still undecided, can help in taking actions to affect their behavior.

We will be studying this problem in a typical machine-learning setting where the prediction model will be built on a set of Web sessions, referred to as a *training set*. This model will then be evaluated for accuracy on a previously unseen set of Web sessions, called the *test set*.

2.1 Markov Models for Web Sessions

The next-page prediction problem can be solved using a probabilistic framework as follows. Let $\mathcal{W}$ be a user's Web session of length l (i.e., it contains l pages), and let $P(p_i|\mathcal{W})$ be the probability that the user visits page p_i next. Then, the

[1] World Wide Web Committee: Web usage characterization activity, http://www.w3.org/WCA.

page p_{l+1} that the user will visit next is given by

$$p_{l+1} = \underset{p \in \mathbb{P}}{\operatorname{argmax}}\{P(\mathcal{P}_{l+1} = p|\mathcal{W})\} = \underset{p \in \mathbb{P}}{\operatorname{argmax}}\{P(\mathcal{P}_{l+1} = p|\mathcal{P}_l, \mathcal{P}_{l-1}, \ldots, \mathcal{P}_1)\}, \quad (1)$$

where $\mathbb{P}$ is the total set of pages present on the Web site. Essentially, this approach for each page p_i computes its probability of being accessed next, and then selects the page that has the highest probability.

The key step in determining p_{l+1} from Equation 1 is to be able to compute the various conditional probabilities given the user's current Web session $\mathcal{W}$. In general, it is not feasible to accurately determine these conditional probabilities because (i) the Web sessions can be arbitrarily long, and (ii) the size of the training set is often much smaller than that required to accurately estimate the various conditional probabilities for long Web sessions. For this reason, the various conditional probabilities are commonly estimated by assuming that the sequence of Web pages visited by the user follows a *Markov process* [Sarukkai 2000; Pitkow and Pirolli 1999]. That is, the probability of visiting a page p_i does not depend on all the pages in the Web session, but only on a small set of k preceding pages, where $k \ll l$. Using the Markov process assumption, the page p_{l+1} that the user will visit next is given by

$$p_{l+1} = \underset{p \in \mathbb{P}}{\operatorname{argmax}}\{P(\mathcal{P}_{l+1} = p|\mathcal{P}_l, \mathcal{P}_{l-1}, \ldots, \mathcal{P}_{l-(k-1)})\}. \quad (2)$$

The number of preceding pages (i.e., observations) k that the next page depends on is called the *order* of the Markov model, and the resulting model $\mathcal{M}$ is called the *kth-order Markov model* [Papoulis 1991].

Besides making the next-page prediction problem computationally tractable, the Markov process assumption correctly captures certain aspects of the overall process that a user follows as he or she browses a Web site. First, the page that a user will visit next does strongly depend on the page that the user is currently visiting, because the user can only reach pages that are directly linked (via forward or backward hyperlinks) from that page. Second, in cases of long Web sessions, pages that the user visited much earlier tend not to influence the user's current actions because they tend to reflect different information needs [Chi et al. 2000]. Nevertheless, there will be cases in which a user's Web site browsing process is not Markovian and, in these cases, such an assumption will lead to inaccurate modeling.

2.1.1 *Estimating Model Parameters.*

In order to use the kth-order Markov model, we need to learn $\mathcal{P}_{l+1}$ for each sequence of k pages, $\mathcal{S}_j^k = \langle \mathcal{P}_{l-(k-1)}, \mathcal{P}_{l-(k-2)}, \ldots, \mathcal{P}_l \rangle$, by estimating the various conditional probabilities, $P(\mathcal{P}_{l+1} = p|\mathcal{P}_l, \mathcal{P}_{l-1}, \ldots, \mathcal{P}_{l-(k-1)})$, of Equation 2. We will refer to the various k-length sequences, $\mathcal{S}_j^k$, as the *states* of the Markov model.

In a Web site with $\mathbb{P}$ pages, there are a total of $\Theta(|\mathbb{P}|^k)$ states in a kth-order Markov model, and a total of $\Theta(|\mathbb{P}|^{k+1})$ conditional probabilities that need to be estimated[2] from the training set. The most commonly used method for

[2] Note that the Θ notation is used because our definition of a state does not allow for the same page to appear in consecutive locations of the k-length sequence.

estimating these conditional probabilities is to use the maximum likelihood principle [Duda et al. 2000]. However, other techniques like Bayes estimation [Duda et al. 2000], capable of incorporating background information, can also be used.

Using the maximum likelihood principle, the conditional probability $P(p_i|\mathcal{S}_j^k)$ is computed by counting the number of times sequence $\mathcal{S}_j^k$ occurs in the training set, and the number of times page p_i occurs immediately after $\mathcal{S}_j^k$. The conditional probability is the ratio of these two frequencies, that is,

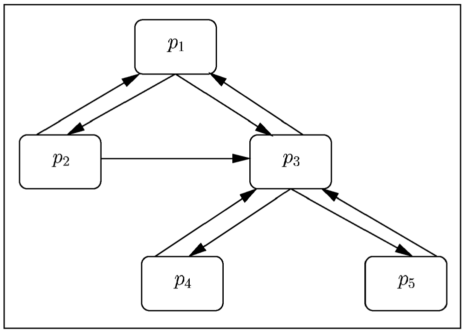
$$P\left(p_i|\mathcal{S}_j^k\right) = \frac{Frequency\left(\langle\mathcal{S}_j^k, p_i\rangle\right)}{Frequency\left(\mathcal{S}_j^k\right)}. \tag{3}$$

An example of the various steps involved in building, learning, and using a first- and second-order Markov model is shown in Figure 1. Figure 1(a) shows the site map for a sample Web site as a directed graph. The nodes of this graph correspond to the Web pages and the arrows to the hyperlinks between these pages. Figure 1(b) displays a set of Web sessions that were generated on this Web site. These set of Web sessions are divided into training set and test set. Figures 1(c) and 1(e) display the frequencies of different states for first- and second-order Markov models, computed on the training set of Web sessions. Figure 1(d) displays how these models are used to predict the most probable page for Web session $\mathcal{W}_{t_1}$, using the first- and the second-order Markov model.

2.1.1.1 *Issues Associated with Parameter Estimation from Web Session Data.* The above example also helps to illustrate some of the characteristics of the Markov models for predicting Web page accesses, and the challenges in properly estimating the parameters of these models. Studying the observed frequencies shown in Figures 1(c) and (e), we can see that a number of them are zero. For example, in Figure 1(c), the frequency for $\langle\mathcal{S}_4^1, p_1\rangle$ and $\langle\mathcal{S}_4^1, p_2\rangle$ is zero. Since the frequency of visiting those pages from state $\mathcal{S}_4^1$ is zero, the associated conditional probabilities will be zero. Such zero frequency counts can be observed for two reasons.

First, it can be due to the structure of a Web site and how the pages on a Web site are connected by hyperlinks. In the Web site in Figure 1(a), it is highly unlikely for a user to visit page p_1 from page p_4, as there is no hyperlink connecting these pages. In such scenarios, a standard technique to obtain nonzero conditional probabilities for such states is known as smoothing [Durbin et al. 1998], which adds a small nonzero number to the frequencies of all the states. Even though smoothing techniques are widely used in other applications of Markov models, it is not of great concern for the next-page prediction problem, as we only need to compute the page that has the highest value of conditional probability.

Second, it can be because the sequence of pages associated with those states is not present in the training set. For example, the second-order state associated with sequence $\langle p_2, p_4\rangle$ does not occur in the training set. If a state is absent in both the training set and the test set, then this is not a serious problem. However, if a state is absent in the training set but occurs in the test set,



(a)

Training set
$\mathcal{W}_1 : \langle p_1, p_3, p_4 \rangle$
$\mathcal{W}_2 : \langle p_1, p_2, p_3, p_5 \rangle$
$\mathcal{W}_3 : \langle p_1, p_2 \rangle$
$\mathcal{W}_4 : \langle p_1, p_3, p_4, p_3, p_1, p_2 \rangle$
$\mathcal{W}_5 : \langle p_1, p_2, p_3, p_5, p_3 \rangle$
$\mathcal{W}_6 : \langle p_1, p_3, p_1, p_2, p_1, p_3, p_4 \rangle$

Test set
$\mathcal{W}_{t_1} : \langle p_1, p_2, p_3, ? \rangle \ \langle p_5 \rangle$
$\mathcal{W}_{t_2} : \langle p_1, p_2, p_3, p_5, p_3, ? \rangle \ \langle p_4 \rangle$

(b)

1^{st}-*Order St.*	*Fr.*	p_1	p_2	p_3	p_4	p_5
$\mathcal{S}_1^1 = \langle p_1 \rangle$	9	0	5	4	0	0
$\mathcal{S}_2^1 = \langle p_2 \rangle$	3	1	0	2	0	0
$\mathcal{S}_3^1 = \langle p_3 \rangle$	7	2	0	0	3	2
$\mathcal{S}_4^1 = \langle p_4 \rangle$	1	0	0	1	0	0
$\mathcal{S}_5^1 = \langle p_5 \rangle$	1	0	0	1	0	0

(c)

First-Order Markov model:

$$\begin{aligned}\mathcal{P}_{l+1} &= \text{argmax}(P(p_1|p_3), P(p_4|p_3), P(p_5|p_3)),\\ &= \text{argmax}(p_1 \Rightarrow 0.29, p_4 \Rightarrow 0.42, p_3 \Rightarrow 0.29),\\ &= p_4.\end{aligned}$$

Second-Order Markov model:

$$\begin{aligned}\mathcal{P}_{l+1} &= \text{argmax}(P(p_5|p_3, p_2)),\\ &= \text{argmax}(p_5 \Rightarrow 1.0),\\ &= p_5.\end{aligned}$$

(d)

2nd- Order State	*Freq.*	p_1	p_2	p_3	p_4	p_5
$\mathcal{S}_1^2 = \langle p_1, p_2 \rangle$	3	1	0	2	0	0
$\mathcal{S}_2^2 = \langle p_1, p_3 \rangle$	4	1	0	0	3	0
$\mathcal{S}_3^2 = \langle p_1, p_4 \rangle$	0	0	0	0	0	0
$\mathcal{S}_4^2 = \langle p_1, p_5 \rangle$	0	0	0	0	0	0
$\mathcal{S}_5^2 = \langle p_2, p_1 \rangle$	1	0	0	1	0	0
$\mathcal{S}_6^2 = \langle p_2, p_3 \rangle$	2	0	0	0	0	2
$\mathcal{S}_7^2 = \langle p_2, p_4 \rangle$	0	0	0	0	0	0
$\mathcal{S}_8^2 = \langle p_2, p_5 \rangle$	0	0	0	0	0	0
$\mathcal{S}_9^2 = \langle p_3, p_1 \rangle$	2	0	2	0	0	0
$\mathcal{S}_{10}^2 = \langle p_3, p_2 \rangle$	0	0	0	0	0	0

2nd-Order State	*Freq.*	p_1	p_2	p_3	p_4	p_5
$\mathcal{S}_{11}^2 = \langle p_3, p_4 \rangle$	1	0	0	1	0	0
$\mathcal{S}_{12}^2 = \langle p_3, p_5 \rangle$	1	0	0	1	0	0
$\mathcal{S}_{13}^2 = \langle p_4, p_1 \rangle$	0	0	0	0	0	0
$\mathcal{S}_{14}^2 = \langle p_4, p_2 \rangle$	0	0	0	0	0	0
$\mathcal{S}_{15}^2 = \langle p_4, p_3 \rangle$	1	1	0	0	0	0
$\mathcal{S}_{16}^2 = \langle p_4, p_5 \rangle$	0	0	0	0	0	0
$\mathcal{S}_{17}^2 = \langle p_5, p_1 \rangle$	0	0	0	0	0	0
$\mathcal{S}_{18}^2 = \langle p_5, p_2 \rangle$	0	0	0	0	0	0
$\mathcal{S}_{19}^2 = \langle p_5, p_3 \rangle$	0	0	0	0	0	0
$\mathcal{S}_{20}^2 = \langle p_5, p_4 \rangle$	0	0	0	0	0	0

(e)

Fig. 1. (a) Site map of a sample Web site. (b) Sample training and test Web sessions, the hidden page for each Web session in the test set is displayed separately. (c) Frequency table for first-order Markov model. (d) Computing prediction for $\mathcal{W}_{t1}$. (e) Frequency table for second-order Markov model.

then the Markov model will not be able to make a prediction for this state. This problem is exacerbated as the order of the Markov model increases, and is especially common in Web domains where the Web site contains thousands of pages, and some portions of the Web site are accessed very infrequently. This situation occurs for the second Web session, $\mathcal{W}_{t_2}$, in the example shown in Figure 1(d). The second-order state corresponding to the page sequence $\langle p_5, p_3 \rangle$ does not occur in the training set, therefore, it has all outgoing frequencies set to zero. In such a case, the model has to resort to a default prediction. In our models the default prediction is the most frequently occurring page in the training set.

2.1.2 *Performance Measures for Markov Models.*

There are several performance measures that we will use to compare different Markov model-based techniques for solving the next-page prediction problem.

The first is the *accuracy* of the model that measures the predictive ability of the model. The accuracy of a model is determined using a separate set of Web sessions (i.e., test set) that was not used during training. In this article, this is done by hiding the last page in each of the test set's Web sessions, and using the model to make a prediction of the resulting *trimmed* Web session. The accuracy is defined as the ratio of the number of Web sessions for which the model is able to correctly predict the hidden page to the total number of Web sessions in the test set. For the example displayed in Figure 1, the accuracy of the first-order Markov model is 50.0%, and the accuracy of the second-order Markov model is also 50.0%.

The second is the *number of states* of the model which measures the space- and time-complexity of learning, and applying the model. A model that requires a large number of states can significantly limit its applicability in cases in which predictions are needed in real time, that is, while the user is still browsing the Web site. The number of states of a Markov model is defined as the total number of states for which a Markov model has estimated (from the training set) the most probable page to be accessed next (using Equation 2). For a Web site with $\mathbb{P}$ pages, the number of states is $\Omega(|\mathbb{P}|^k)$. However, as discussed in Section 2.1.1, many of these states will not occur in the training set and we will not be counting them towards this measure. For the example shown in Figure 1, the first-order Markov model has 5 states (i.e., $\langle p_1\rangle$, $\langle p_2\rangle$, $\langle p_3\rangle$, $\langle p_4\rangle$, $\langle p_5\rangle$), while the second-order Markov model has 8 states (i.e., $\langle p_1, p_2\rangle$, $\langle p_1, p_2\rangle$, $\langle p_1, p_3\rangle$, $\langle p_2, p_1\rangle$, $\langle p_3, p_1\rangle$, $\langle p_3, p_4\rangle$, $\langle p_3, p_5\rangle$, $\langle p_4, p_3\rangle$).

The third is the *coverage* of the model that measures the number of times a Markov model was able to compute a prediction without resorting to the default prediction. The coverage of a Markov model is evaluated on a test set. It is defined as the ratio of the number of Web sessions whose state required for making a prediction was found in the model to the total number of Web sessions in the test set. For example, the coverage of the model displayed on the test set displayed in the Figure 1(b) is 100.0% for the first-order Markov model, and 50.0% for the second-order Markov model.

The last is the *model accuracy*, defined as the accuracy on the portion of the test set for which the model was able to locate the state required for prediction (i.e., it did not perform any default predictions).

2.2 All *K*th-Order Markov Models

In many applications, lower-order Markov models (e.g., first- and/or second-order) are not successful in accurately predicting the next page to be accessed by the user. This is because, in these models, the Markov process assumption that is being made is incorrect. In other words, these models do not look far into the past to correctly discriminate the different behavioral modes of the

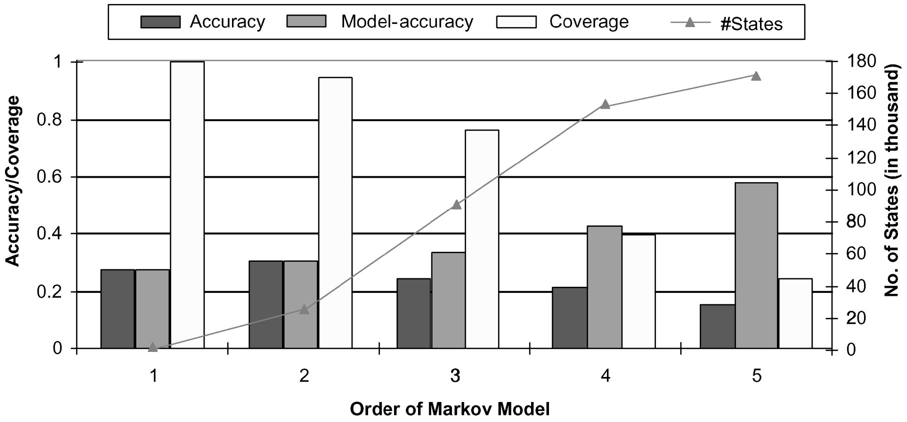


Fig. 2. Plot comparing accuracy, coverage, model accuracy and model size with the order of Markov model.

different users. As a result, in order to obtain better predictions, higher-order models must be used (e.g., third and fourth). Unfortunately, these higher-order models have a number of limitations: (i) high state-space complexity, (ii) reduced coverage, and (iii) sometimes even worse accuracy as a result of the lower coverage.

To better understand these shortcomings we conducted an experiment using a dataset obtained from an e-commerce Web site (the EC1 dataset described in Section 4.1). We compared various order Markov models starting from the first, all the way to the fifth-order model. For each of the models we computed accuracy, coverage, number of states, and model accuracy.

The results are plotted in Figure 2, and we can see that as the order of the model increases, the accuracy of the model decreases, accompanied by a decrease in the coverage. However, the model accuracy of the model continues to increase. This indicates that though higher-order model can locate states for only a small set of Web sessions, they are able to predict these Web sessions with a greater accuracy than lower-order models do. It is worth noting that as the order of the model increases, the number of states used for the model also increases dramatically. However, the rate of increase reduces for higher-order models because of the fact that the structure of the Web site constrains the space of possible states.

One method to overcome the problem of low coverage on the test set is to train varying order Markov models and then combine them for prediction [Pitkow and Pirolli 1999]. In this scheme, for each test instance, the highest-order Markov model that covers the instance is used for prediction. For example, if we build the first-, second-, and third-order Markov models, then given a test instance, we first try to make a prediction using the third-order model. If this model does not contain the corresponding state, then we try to make a prediction using the second-order model, and so on. This scheme is called the *All-Kth-Order Markov model* [Pitkow and Pirolli 1999]. Note that even though the All-*K*th-Order Markov model solves the problem of low coverage, it exacerbates the problem of model size as the states of all the different order Markov models are now part of the model.

3. SELECTIVE MARKOV MODELS

Despite their limitations, the All-Kth-Order Markov model typically holds the promise of achieving higher prediction accuracies and improved coverage on the test set than any single-order Markov model does. However, this increased power comes at the expense of a dramatic increase in the state-space complexity. This led us to develop techniques to intelligently combine different order Markov models so that the resulting model has low state complexity, retains the coverage of the All-Kth-Order Markov model, and achieves comparable or better prediction accuracies.

Our schemes were motivated by the observation that given a sequence of pages for which we need to predict the next most probable page, there are multiple states in the All-Kth-Order Markov model that can be used to perform this prediction. In fact, there can be as many states as the number of the different order Markov models used to form the All-Kth-Order Markov model. Now, depending on the particular set of states involved, each of them can have different prediction accuracies. Based on this observation, we can then start from the All-Kth-Order Markov model and eliminate many of its states that are statistically expected to have low prediction accuracy. This allows us to reduce the overall state complexity without affecting the prediction accuracy of the overall scheme.

The starting point for all of our algorithms is the All-Kth-Order Markov model obtained by building a sequence of increasing order Markov models. However, instead of using this model for prediction, we use a number of techniques to eliminate certain states across the different order Markov models. The set of states that survive this step, then become the final model that is used for prediction. The goals of this *pruning* step is primarily to reduce the state complexity and secondarily, improve the prediction accuracy of the resulting model. We will refer to these models as *selective Markov models* (SMM).

Given an SMM, the actual prediction algorithm is similar to that used by the All-Kth-Order Markov model. For a given sequence of pages, we first identify all the states in the model that can be used to predict the next page,[3] and then use the highest-order state among them to compute the prediction.

The key step in our algorithm is the scheme used to determine the potential accuracy of a particular state. We developed three different schemes that have an increasing level of complexity. The first scheme simply eliminates the states that have very low occurrence frequency. The second scheme uses a technique to identify states for which the conditional probabilities of the two most prominent pages are not significantly different; such states are pruned. Finally, the third scheme uses an estimated-error-based approach to eliminate states with low prediction accuracy. These schemes are described in the rest of this section.

[3] Note that these states will be part of Markov models of different orders.

3.1 Frequency-Pruned Markov Model

The *frequency-pruned Markov model* (FPMM) is based on the observation that states that occur with low frequency in the training set, tend to also have low prediction accuracies. This is primarily due to the fact that, for such states, the maximum likelihood estimations of the conditional probabilities will not be reliable. Consequently, these low frequency states can be eliminated without affecting the accuracy of the resulting model. The amount of pruning in the FPMM scheme is controlled by the parameter ϕ, referred to as the *frequency threshold*. Using this parameter, FPMM eliminates all the states of the different kth-order Markov models for $k > 1$, that occur in fewer than ϕ training set instances. Note that FPMM will never prune a state from a first-order Markov model that will not reduce the coverage of the original model.

There are a number of observations to be made about the FPMM scheme. First, the same frequency threshold is used for all the models regardless of their order. Second, this pruning policy is more likely to prune higher-order states since higher-order states occur with lower frequency. Third, the frequency threshold parameter ϕ, specifies the actual number of training set instances that must be supported by each state and not the fraction of training set instances as is often done in the context of association rule discovery [Agrawal et al. 1993]. This is done primarily because the trustworthiness of the estimated conditional probabilities of a particular state depends on the actual number of training set instances and not on its frequency relative to the size of the dataset.

3.2 Confidence-Pruned Markov Model

One of the limitations of the FPMM scheme is that it does not capture all the parameters that influence the accuracy of the state. In particular, the probability distribution of outgoing pages from a state is completely ignored. For example, consider a Markov state that has two outgoing pages, such that one of them is substantially more probable than the other. Even if the overall occurrence frequency of this state is somewhat low, the predictions computed by this state will be quite reliable because of the clear difference in the outgoing probabilities. On the other hand, if the outgoing probabilities in the above example are very close to each other, then, in order for that difference to be reliable, they must be based on a large number of training instances. Ideally, we would like the pruning scheme to not only consider the occurrence frequency of the state, but also weigh the probability distribution of the outgoing pages before making its pruning decisions.

This observation led to us to develop the *confidence-pruned Markov model* (CPMM) scheme. CPMM uses a technique to determine for each state if the probability of the most frequently accessed page is significantly different from the probabilities of the other pages that can be accessed from this state. If the probability differences are not significant, then this state is unlikely to give high accuracy and it is pruned. In contrast, if the probability differences are significant, the state is retained.

The CPMM scheme retains a state if the probability difference between the most probable page and the second most probable page is above a certain threshold (ϕ_c), called the *confidence threshold*. If the probability difference between the two most probable pages is below the confidence threshold, the state is pruned. The confidence threshold is computed as follows: if $\hat{p}$ is the probability of the most probable page, then

$$\phi_c = \hat{p} - z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \tag{4}$$

where $z_{\alpha/2}$ is the upper $\alpha/2$ percentage point of the standard normal distribution, and n is the frequency of the Markov state.

The degree of pruning in CPMM is controlled by $z_{\alpha/2}$ (confidence coefficient). As the value of $z_{\alpha/2}$ increases, the size of the confidence threshold (ϕ_c) decreases, resulting in increased pruning. Note that if a state occurs with a high frequency, then Equation 4 will compute a tighter (smaller) confidence threshold. As a result, even if the difference in the probabilities between the two most probable pages is relatively small, the state will most likely be retained. This behavior is desirable because in cases in which a state has a large number of outgoing pages, even small preferences toward one of these pages conveys significant information.

3.3 Error-Pruned Markov Model

In the previous schemes we used either the frequency of a state or the probability distribution of its outgoing pages to gauge the potential error associated with a state. However, the error of a state can also be automatically estimated and used to prune that state. A widely used approach to estimate the error associated with each state is to perform a *validation step*. During the validation step, the entire model is tested using part of the training set (*validation set*) that was not used during the model-building phase. Since, we know the actual pages visited in Web sessions of the validation set, we can easily determine the error rate for each state and use it for pruning. Note that the validation set is created from the training set and the Web sessions present in the test set are never seen during the model-building phase.

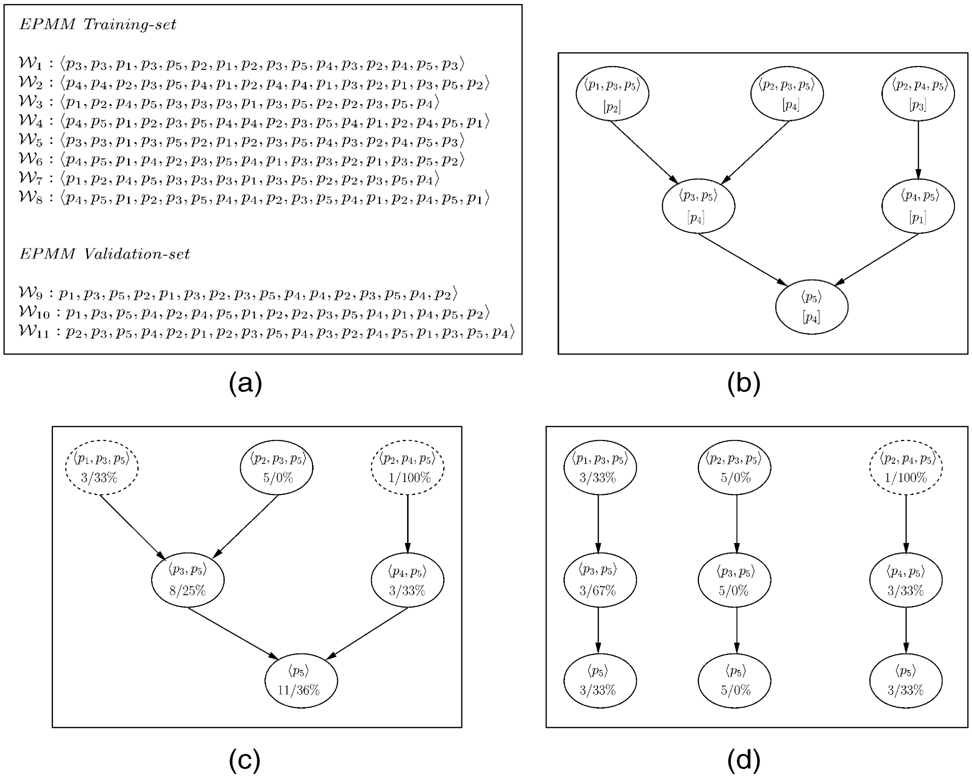
This observation led us to develop the *error-pruned Markov model* (EPMM) scheme, in which the final predictions are computed by using only the states of the model that have the smallest estimated error rate. Though EPMM also uses the All-Kth-Order Markov model as its starting point, like FPMM and CPMM, it differs from those two schemes in the way pruning is done. In both FPMM and CPMM schemes, a single parameter (ϕ or ϕ_c) is computed and the whole Markov model is pruned using this parameter. However, in the case of EPMM a higher-order state is pruned by comparing its error rate with the error rate of its lower-order states. For example, to prune the state $\mathcal{S}_q^3$ (associated with page sequence $\langle p_i, p_j, p_k\rangle$), its error rate will be compared with the error rate for states $\mathcal{S}_r^2$ (associated with page sequence $\langle p_j, p_k\rangle$), and state $\mathcal{S}_s^1$ (associated with page sequence $\langle p_k\rangle$); the state $\mathcal{S}_q^3$ will be pruned if its error rate is higher than any of them.

We have developed two error-based pruning strategies. Both of these methods follow a similar approach of pruning but differ in the way the error rate is computed for each state. In the first scheme, referred as *overall error pruning*, every lower-order Markov state has a single error rate value that is computed over the entire validation set. In the second scheme, that we will refer to as *individual error pruning*, each of the lower-order states has many error rates associated with it, one for each one of its corresponding higher-order states. The details of these two schemes and our approach for estimating the error rates are described in the rest of this section.

3.3.1 *Overall Error Pruning.* In this scheme, each one of the K-Markov models of the All-Kth-Order Markov model is individually validated using the validation set. For each Web session in the validation set, the state of the Markov model used for prediction is identified, and the result of the prediction is recorded for that state. After evaluating all the Web sessions in the validation set, the error rate for each state is computed. During the pruning step, for each state in the Markov model, we identify its corresponding lower-order states. For example, if the higher-order state corresponds to the page sequence $\langle p_i, p_j, p_k \rangle$, then the lower-order states that are identified correspond to page sequences $\langle p_j, p_k \rangle$ (second-order), $\langle p_k \rangle$ (first-order). Now if the error rate of the higher-order state is greater than *any* of its corresponding lower-order sates, the state is pruned. The same procedure is repeated for all the states in the lower-order Markov models as well, except from the first-order Markov model. The states from the first-order Markov model are never pruned in order to retain the coverage of the resulting model.

Figure 3 illustrates this pruning strategy, using a small example. Figure 3(a) shows a set of Web sessions used for training and validating an All-Kth-Order Markov model with $K = 3$. Figure 3(b) shows some of the states from the different order Markov models, along with the most probable page for each state. Figure 3(c) shows the frequency and the error rate for these states after the validation-step. For example, the states corresponding to page sequences $\langle p_1, p_3, p_5 \rangle$ and $\langle p_2, p_4, p_5 \rangle$ have error rates of 33% and 100% respectively, which are higher than the corresponding second-order states, and are pruned.

3.3.2 *Individual Error Pruning.* In the overall error pruning scheme, a particular state $\mathcal{S}_q^3 = \langle p_i, p_j, p_k \rangle$, is pruned if its corresponding lower-order state, $\mathcal{S}_r^2 = \langle p_j, p_k \rangle$, has a smaller error rate. This error rate is estimated by taking into account all the Web sessions C_q and C_r that can be predicted by $\mathcal{S}_q^3$ and $\mathcal{S}_r^2$, respectively. However, the set of Web sessions predicted by $\mathcal{S}_r^2$ is a superset of the Web sessions predicted by $\mathcal{S}_q^3$. As a result, there may be cases in which $\mathcal{S}_r^2$ achieves a lower error rate because it generates better predictions for the Web sessions in $C_r - C_q$, even if its predictions for the Web sessions in C_q are actually worse than the corresponding predictions by the $\mathcal{S}_q^3$ state. This can easily happen if the cardinality of the set $C_r - C_q$ is large. In such cases, the overall error pruning scheme will incorrectly decide to eliminate $\mathcal{S}_q^3$ in favor of $\mathcal{S}_r^2$. This limitation of the overall error pruning scheme stems from the fact that it assigns a single overall error rate to each state. Thus, it cannot discriminate between cases in which a particular state performs well (or poorly) for a certain well-defined subset of Web sessions.



EPMM Training-set

$\mathcal{W}_1 : \langle p_3, p_3, p_1, p_3, p_5, p_2, p_1, p_2, p_3, p_5, p_4, p_3, p_2, p_4, p_5, p_3 \rangle$
$\mathcal{W}_2 : \langle p_4, p_4, p_2, p_3, p_5, p_4, p_1, p_2, p_4, p_4, p_1, p_3, p_2, p_1, p_3, p_5, p_2 \rangle$
$\mathcal{W}_3 : \langle p_1, p_2, p_4, p_5, p_3, p_3, p_3, p_1, p_3, p_5, p_2, p_2, p_3, p_5, p_4 \rangle$
$\mathcal{W}_4 : \langle p_4, p_5, p_1, p_2, p_3, p_5, p_4, p_4, p_2, p_3, p_5, p_4, p_1, p_2, p_4, p_5, p_1 \rangle$
$\mathcal{W}_5 : \langle p_3, p_3, p_1, p_3, p_5, p_2, p_1, p_2, p_3, p_5, p_4, p_3, p_2, p_4, p_5, p_3 \rangle$
$\mathcal{W}_6 : \langle p_4, p_5, p_1, p_4, p_2, p_3, p_5, p_4, p_1, p_3, p_3, p_2, p_1, p_3, p_5, p_2 \rangle$
$\mathcal{W}_7 : \langle p_1, p_2, p_4, p_5, p_3, p_3, p_3, p_1, p_3, p_5, p_2, p_2, p_3, p_5, p_4 \rangle$
$\mathcal{W}_8 : \langle p_4, p_5, p_1, p_2, p_3, p_5, p_4, p_4, p_2, p_3, p_5, p_4, p_1, p_2, p_4, p_5, p_1 \rangle$

EPMM Validation-set

$\mathcal{W}_9 : p_1, p_3, p_5, p_2, p_1, p_3, p_2, p_3, p_5, p_4, p_4, p_2, p_3, p_5, p_4, p_2 \rangle$
$\mathcal{W}_{10} : p_1, p_3, p_5, p_4, p_2, p_4, p_5, p_1, p_2, p_2, p_3, p_5, p_4, p_1, p_4, p_5, p_2 \rangle$
$\mathcal{W}_{11} : p_2, p_3, p_5, p_4, p_2, p_1, p_2, p_3, p_5, p_4, p_3, p_2, p_4, p_5, p_1, p_3, p_5, p_4 \rangle$

(a)

$\langle p_1, p_3, p_5 \rangle$ $[p_2]$
$\langle p_2, p_3, p_5 \rangle$ $[p_4]$
$\langle p_2, p_4, p_5 \rangle$ $[p_3]$
$\langle p_3, p_5 \rangle$ $[p_4]$
$\langle p_4, p_5 \rangle$ $[p_1]$
$\langle p_5 \rangle$ $[p_4]$

(b)

$\langle p_1, p_3, p_5 \rangle$ 3/33%
$\langle p_2, p_3, p_5 \rangle$ 5/0%
$\langle p_2, p_4, p_5 \rangle$ 1/100%
$\langle p_3, p_5 \rangle$ 8/25%
$\langle p_4, p_5 \rangle$ 3/33%
$\langle p_5 \rangle$ 11/36%

(c)

$\langle p_1, p_3, p_5 \rangle$ 3/33%
$\langle p_2, p_3, p_5 \rangle$ 5/0%
$\langle p_2, p_4, p_5 \rangle$ 1/100%
$\langle p_3, p_5 \rangle$ 3/67%
$\langle p_3, p_5 \rangle$ 5/0%
$\langle p_4, p_5 \rangle$ 3/33%
$\langle p_5 \rangle$ 3/33%
$\langle p_5 \rangle$ 5/0%
$\langle p_5 \rangle$ 3/33%

(d)

Fig. 3. (a) Training and validating Web sessions. (b) Various order Markov states with their maximum frequency page. (c) Overall error rates for Markov states. The states with dotted lines are pruned. (d) Individual error rates for Markov states for *order* = 3. The states with dotted lines are pruned.

This observation led us to develop the *individual error pruning scheme*. The key idea behind this scheme is, when deciding whether or not to prune a particular state in favor of its corresponding lower-order state, to estimate the error rate only on the Web sessions that can be predicted by both of these states. For instance, in our previous example, when comparing states $\mathcal{S}_q^3$ and $\mathcal{S}_r^2$ we will use only the set C_q for estimating the error rate of the two states. This approach allows us to directly compare the prediction performance of the two states on the same set of Web sessions.

The individual error pruning scheme is performed level-by-level, starting with the states of the highest-order Markov model. For each state, we identify all the Web sessions of the validation set that it can predict, and use them to estimate the error rate for itself and its lower-order states. If the error rate of the lower-order states is smaller, then the higher-order state is pruned, otherwise it is retained. As was the case with the overall error pruning method, the states from the first-order Markov model are never pruned so as to retain the coverage of the resulting model. From the above description, we can see that in the new scheme, for each lower-order state, we estimate multiple error rates, one for each of its corresponding higher-order states. This allow us to accurately evaluate the performance of each state, within the *context* of its various higher-order states.

The individual error pruning strategy is illustrated in Figure 3(d) for the fragment of the All-Kth-Order Markov model model shown in Figure 3(b). Figure 3(d) displays the error rate for third-order states. Note that the error rate for all the states are computed only on those examples that are supported by the third-order states. In this case, the state $\langle p_1, p_3, p_5\rangle$ will not be pruned because its error rate is lower for the Web sessions that it can predict, compared to the error rate of its corresponding second-order state. However, the $\langle p_2, p_4, p_5\rangle$ state will still be pruned as its error rate is still higher than its second-order state $\langle p_4, p_5\rangle$.

3.3.3 *Additional Considerations.* One of the problems associated with naively applying the above error-pruning approaches is that not all states in the Markov model will be thoroughly validated, as only a part of the training set is used during the validation step. To overcome this problem we repeat the training and validation steps multiple times by dividing the dataset into training and validation sets in a different fashion each time. This can be thought of as performing cross validation on the training set.

4. EXPERIMENTAL RESULTS

We experimentally evaluated the performance of the proposed selective Markov models on a variety of datasets. In the rest of this section, we briefly describe these datasets, our experimental methodology, and present the results obtained by our schemes.

4.1 Datasets

We evaluated the performance of the proposed schemes on the following four datasets.

—*ECommerce Web Logs*: We used the Web server logs from two e-commerce companies for our analysis. The first dataset was from the Web site `Gazelle.com`, and was used as part of the KDDCup 2000 competition [Kohavi et al. 2000]. The second dataset was from the Web site `Fingerhut.com` (`http://www.fingerhut.com`). These Web logs were first cleaned using the WebSIFT system [Cooley et al. 1999] that preprocesses the Web logs and identifies Web sessions. Each session corresponds to the sequence of Web pages accessed by the user during his/her visit to the site. Note that the session contains only the accesses to Web pages—accesses to images are ignored. These two datasets will be referred to as *EC1* and *EC2*.

—*OWL Dataset*: This dataset contains the log of editing commands typed by different users in Microsoft Word over a period of 2 years [Linton 2000]. The goal of the model built on this dataset was to predict the next-command-typed by the user based on the user's past sequence of commands. Such a predictive system could be used to make online recommendations to the user about commands that could be typed next. This dataset will be referred to as *OWL*.

Table I. Dataset Statistics

Dataset	# Sessions	Avg. Ses. Length	# unique pages	Average Branching Factor 1st-Order	2nd-Order	3rd-Order
EC1	90,693	6.66	900	12.74	2.15	1.49
EC2	113,723	8.74	6,154	33.52	4.00	1.80
OWL	14,025	4.76	170	10.79	3.05	1.97
TC1	78,280	4.05	164	14.42	3.06	1.72

In this dataset, each session corresponds to the sequence of commands typed by a user on a particular document in one sitting. The different commands typed by the users constitute the pages of the Markov model.

—*Telephone Switch Dataset*: This dataset was obtained from a large telecommunications company that maintains nationwide telephone networks. The dataset contains the log of different alarms generated by a telephone switch over a period of one month. Each session in this dataset corresponds to the sequence of alarms given out by the switch that is related to the same problem. For this dataset, the alarm generated by the switch is considered a page. This dataset will be referred as *TC1*.

The characteristics of these datasets are shown in the Table I. All sessions in the dataset have a minimum length of 2. To get a better understanding of the distribution in the dataset, we define a measure called *average branching factor*. This measure is computed on the Markov model built on training set, and is defined as the average number of total outgoing pages from each state in the Markov model. For each dataset, Table I displays the average session length, number of unique pages or alphabet size, and the average branching factor for different orders of Markov models.

4.2 Experimental Design and Measures

To make the evaluation of different schemes manageable, we limit ourselves to the problem of predicting just the last page of a test example/session. For evaluation purposes, we divide the complete dataset into training set and test set. Depending on the model, the training set may further be divided into a validation set. During the testing step, the model is given a *trimmed session* for prediction in which the last page of the session is hidden. The prediction made by the model is then compared with the hidden page of the session to compute the accuracy of the model. In some cases, Markov model-based schemes are unable to make a prediction for a session in the test set. This could be either because the length of test session is less than the order of the model, or the model has not seen a similar session in the training step, in which a case the model makes a default prediction. The default prediction is the most frequently occurring page in the training set. In all of our experiments, for both the All-Kth-Order Markov model and selective Markov model schemes, we combined first-, second-, and third-order Markov models, i.e., $K = 3$.

The overall performance of the various Markov model-based algorithms were evaluated in terms of their accuracy and their model size (number of states); both these measures are defined in Section 2.1.2.

Table II. The Accuracy and the Model Size of FPMM for Different Values of Frequency Threshold(ϕ). Boldfaced Entries Correspond to the Highest Obtained Accuracy Levels

Freq.	EC1		EC2		OWL		TC1	
Thr.	Accuracy	# states	Accuracy	# states	Accuracy	# states	Accuracy	# states
0	30.24	126464	58.27	232003	46.00	7944	**76.34**	6074
2	30.68	44528	58.47	72271	46.12	3170	76.32	3148
4	31.32	20914	58.79	32862	47.11	1598	76.20	1905
6	31.56	14164	58.86	22111	**47.34**	1116	76.13	1458
8	31.65	10899	58.90	17279	47.21	899	76.04	1223
10	31.71	8952	**58.91**	14629	47.12	767	75.95	1059
12	**31.74**	7661	58.90	12898	47.27	669	75.86	950
14	31.73	6716	58.90	11689	47.32	600	75.78	868
16	31.72	5969	58.90	10850	47.19	552	75.75	808
18	31.72	5389	58.89	10206	47.14	513	75.75	761
20	31.72	4965	58.87	9695	47.05	487	75.75	718
22	31.67	4609	58.86	9297	46.98	458	75.68	675
24	31.67	4296	58.86	8963	46.95	437	75.67	644

4.3 Results for Frequency-Pruned Markov Model

The goal of our first set of experiments was to study the effect of the frequency threshold (ϕ) parameter on the performance of the frequency-pruned Markov model. Toward this goal, we performed an experiment in which we used different values for ϕ ranging from 0 (no pruning) up to 24 in increments of two, and measured both the accuracy and the number of states in the resulting Markov model. These results are shown in Table II for the four datasets in our experimental test bed. Note that the frequency threshold(s) that achieve the highest accuracies are shown using a boldface font.

A number of interesting observations can be made from Table II. First, for three of the four datasets, initially an increase in the frequency threshold, is accompanied by an increase in the accuracy. Second, as we continue to further increase the frequency threshold, the accuracy achieved on the different datasets starts to decrease. This decrease in the overall accuracy with increasing values of the ϕ is because some useful Markov states are being pruned from the model, affecting its overall accuracy. Third, each one of the four datasets achieve their maximum accuracy levels at different values of the frequency threshold, indicating that the *optimal* value of ϕ is dataset dependent. However, the overall accuracy tends to change smoothly with ϕ, making it possible to use a validation set approach to estimate its optimal value for each dataset.

The effect of the frequency threshold is very pronounced on the number of states in the Markov model. The size of the model drops drastically as the frequency threshold increases. For example, in the case of the *EC2* dataset, the number of states in the FPMM scheme with highest accuracy is almost 6% of the number of states of the All-Kth-Order Markov model. The dramatic reduction in the number of states and the accompanied modest improvement in accuracy are due to the fact that the predictions made by states that have low occurrence frequency are not reliable and these states should be eliminated.

Table III. The Accuracy and Model Size of CPMM for Different Values of $z_{\alpha/2}$. Boldfaced Entries Correspond to the Highest Obtained Accuracy Levels

	EC1		EC2		OWL		TC1	
$z_{\alpha/2}$	Accuracy	# states	Accuracy	# states	Accuracy	# states	Accuracy	# states
0	30.24	126464	58.27	232003	46.00	7944	76.34	6074
0.75	31.80	18244	58.91	38672	47.80	1831	76.38	2272
0.84	31.84	16984	58.92	37628	47.77	1779	76.39	2222
0.93	31.90	15277	**58.93**	35475	47.75	1669	76.41	2149
1.03	**31.93**	13697	**58.93**	32953	**47.81**	1537	**76.44**	2056
1.15	31.90	13452	**58.93**	32752	**47.81**	1523	76.43	2042
1.28	31.89	11054	58.92	28261	47.62	1302	76.37	1825
1.44	31.84	10532	58.89	27710	47.62	1269	76.35	1791
1.64	31.76	9707	58.84	26640	47.46	1206	76.34	1737
1.96	31.65	8824	58.75	25282	47.10	1133	76.39	1647
2.57	31.44	7443	58.61	22942	46.70	1006	76.28	1472

4.4 Results for Confidence-Pruned Markov Model

To study the effect of the confidence threshold (ϕ_c) used by the CPMM scheme on the accuracy and state-space complexity of the resulting model, we performed a sequence of experiments in which we varied $z_{\alpha/2}$ from 0.75 to 2.57.[4] Note that from Equation 4, $z_{\alpha/2}$ controls the value of confidence threshold ϕ_c, that is, as the value of $z_{\alpha/2}$ increases, the value of confidence threshold decreases, resulting in increased pruning. The accuracy values and the model size for different values of the $z_{\alpha/2}$ are shown in Table III.

As we can see from the table, the value of the $z_{\alpha/2}$ has a similar effect on the accuracy as that of the frequency threshold parameter used in FPMM. Initially, the accuracy improves as we increase the value of the $z_{\alpha/2}$ (i.e., reduce the value of confidence threshold (ϕ_c)); however, after a certain point, the accuracy starts to decrease. The reason for this performance degradation is due to the fact that CPMM prunes a larger fraction of the states as the value of the confidence threshold decreases. Consequently, some reasonably high accuracy states are getting pruned as well.

Looking at the effect of $z_{\alpha/2}$ on the model size, we observe that as the value of the $z_{\alpha/2}$ increases, the number of states in a model decreases. However, the model size reduction achieved by CPMM is somewhat lower than that achieved by FPMM. We believe that this is because the CPMM scheme tends to retain some of the states that would have been pruned by FPMM, because their most probable outgoing pages are sufficiently more frequent than the rest. Comparing the CPMM and FPMM schemes in terms of prediction accuracy, we can see that, as expected, the CPMM achieves somewhat higher accuracies when compared to the FPMM schemes corresponding to ϕ values that lead to similar state-space complexities. However, as our statistical analysis in Section 4.6 will show, these differences are in general not statistically significant.

[4] Equation 4 was developed for the confidence-interval equation used for testing if the probability of the most probable page is significantly different from the probability of the second most probable page. The values of $z_{\alpha/2}$ are varied corresponding to z values associated with the different confidence levels, starting form 55% having the $z_{\alpha/2}$ value of 0.75, all the way to 99% which has the $z_{\alpha/2}$ value of 2.57.

Table IV. The Accuracy and Model Size of Error-Pruned Markov Models for Different Datasets

	EC1		EC2		OWL		TC1	
Model	Accuracy	# states	Accuracy	# states	Accuracy	# states	Accuracy	# states
All Kth	30.24	126464	58.27	232003	46.00	7944	76.34	6074
O.EPPM	32.40	6253	58.94	14034	49.04	746	76.06	888
I.EPPM	32.32	2237	59.02	7024	49.10	323	76.55	398

Table V. The Accuracy and Model Size of Different Markov Models on the Four Datasets. Boldfaced Entries Correspond to the Highest Obtained Accuracy Levels

	EC1		EC2		OWL		TC1	
Model	Accuracy	# states	Accuracy	# states	Accuracy	# states	Accuracy	# states
First	28.28	876	57.59	6102	42.91	165	55.64	160
Second	31.66	27371	58.13	74390	44.20	2143	59.13	1578
Third	28.65	98216	57.17	151510	40.95	5635	79.52	4335
All Kth	30.24	126464	58.27	232003	46.00	7944	76.34	6074
FPMM	31.74	7661	58.91	14629	47.34	1116	76.34	6074
CPMM	31.93	13697	58.93	32752	47.81	1523	76.44	2056
O.EPPM	**32.40**	6253	58.94	14034	49.04	746	76.06	888
I.EPPM	32.32	2237	**59.02**	7024	**49.10**	323	**76.55**	398

4.5 Results for Error-Pruned Markov Model

In this section, we compare the performance of the overall and individual error-pruning schemes discussed in Section 3.3. The accuracy and model size achieved by these schemes on the four datasets are shown in Table IV. The rows labeled "O.EPPM" and "I.EPPM" correspond to the overall and individual error-pruned schemes, respectively.

From these results, we can see that both I.EPMM and O.EPMM schemes achieve a substantial reduction in the number of states in the model, while still maintaining a reasonably high accuracy. Comparing the accuracies achieved by the two schemes, we can see that, in general, the individual error-pruned schemes leads to somewhat better predictions, even though those improvements are not statistically significant (as discussed later). Comparing the model size of the two schemes, we can see that the individual error-pruned scheme leads to models that have 36% to 50% fewer states than the corresponding models of the overall error-pruned scheme. These results suggest that the individual error-pruned scheme is more aggressive in pruning states.

4.6 Overall Comparison of Different Schemes

Our last set of experiments compares the performance achieved by our different selective Markov model schemes against that achieved by the first-, second-, third-, and All-Kth-order Markov models. These results are shown in Table V. Note that the results for FPMM and CPMM correspond to the results obtained using the optimal frequency threshold (ϕ) and the optimal confidence threshold (ϕ_c), respectively, for each one of the four datasets.

From the results in Table V, we can conclude that even though pruning schemes eliminate a large fraction of the states from the All-Kth-Order Markov model, the accuracy of the resulting schemes is not affected. Though the accuracies of the error-pruned Markov model are numerically higher than most

Table VI. Statistical Significance Comparisons Between the Different Schemes. Each Cell Shows the Number of Datasets the Scheme of the Row Does Better, Equal, or Worse than the Scheme on the Column, Respectively

Model	First	Second	Third	All *K*th	FPMM	CPMM	O.EPMM	I.EPMM
First	—	0, 2, 2	0, 3, 1	0, 2, 2	0, 2, 2	0, 2, 2	0, 1, 3	0, 1, 3
Second	2, 2, 0	—	1, 2, 1	0, 3, 1	0, 3, 1	0, 3, 1	0, 3, 1	0, 3, 1
Third	1, 3, 0	1, 2, 1	—	1, 2, 1	1, 0, 3	1, 0, 3	1, 0, 3	1, 0, 3
All *K*th	2, 2, 0	1, 3, 0	1, 2, 1	—	0, 4, 0	0, 3, 1	0, 3, 1	0, 3, 1
FPMM	2, 2, 0	1, 3, 0	3, 0, 1	0, 4, 0	—	0, 4, 0	0, 4, 0	0, 4, 0
CPMM	2, 2, 0	1, 3, 0	3, 0, 1	1, 3, 0	0, 4, 0	—	0, 4, 0	0, 4, 0
O.EPPM	3, 1, 0	1, 3, 0	3, 0, 1	1, 3, 0	0, 4, 0	0, 4, 0	—	0, 4, 0
I.EPPM	3, 1, 0	1, 3, 0	3, 0, 1	1, 3, 0	0, 4, 0	0, 4, 0	0, 4, 0	—

of the models, the difference is not statistically significant as the variance for the accuracy is of the order of 0.5%. Comparing the model size for different schemes, we observe that schemes based on error pruning have the smallest number of states, with I.EPPM having 1.7%, 3.0%, 4.0%, and 6.5% of All-*K*th-Order Markov model states on the four datasets *EC1, EC2, OWL* and *TC1*, respectively.

To compare the different schemes across the different datasets better, we performed statistical significance tests on the accuracy values of the different models. These results are shown in Table VI. The details of these tests are explained in the Appendix. Each cell in Table VI contains three numbers, corresponding to the number of datasets that the scheme along the row does statistically better, equal, or worse than the scheme along the column, respectively. As we can see from these results, the selective Markov model schemes have similar accuracies as those displayed by the All-*K*th-Order Markov model, indicating that the extensive pruning done by the different pruning schemes does not affect the accuracy of the model.

5. RELATED RESEARCH

The problem of modeling users on the World Wide Web is an active research field that covers a wide range of problems. Providing a survey of the different problems and ongoing research efforts is beyond the scope of this section, and we will only focus on research that addresses the prediction problem, outlined in Section 2. The reader should refer to the excellent survey of Srivastava et al. [2000] that describes the research activities in this area.

Pitkow and Pirolli [1999] predict the next Web page by discovering the longest repeating subsequences in the Web sessions, and then using a weighted scheme to match it against the test Web sessions. They also propose the All-*K*th-Order Markov model as a baseline comparison for their models. Though their scheme is able to reduce the model complexity by an order of magnitude, the accuracy of the resulting scheme is marginally worse than the All-*K*th-Order Markov model. Sarukkai [2000], besides using the Markov model for predicting the next page in the Web session, also proposes two novel Web applications for Markov models: generating tours of a Web site, and identifying Hubs/Authorities [Kleinberg 1999] on a Web site. The accuracies for the prediction problem are reported to be around 50%.

Schechter et al. [1998] build a tree-like data structure that stores the sequence of pages accessed by the users. To reduce the complexity and size of the data structure, a sequence is added to the data structure only if it occurs with sufficient frequency in the training set. For predicting the next accessed page, a user's incomplete session is mapped on to the tree and the leaf nodes (suffixes) stored in the tree are used for prediction. They evaluate their technique on three different Web sites and report accuracies in the range of 50% to 70%.

Theusinger and Huber [2000] study the problem of predicting if a user will buy a product on a Web site. In their approach, they identify a set of about 40 features, mostly nonsequential, and represent each Web session in that feature space. They experimented with three predictive models: decision trees, regression analysis, and neural networks. Of the three techniques neural network displayed the highest predictive power.

Padmanabham and Mogul [1996] study the prediction problem for prefetching Web pages to reduce the Web cache latency. In their approach, a dependency graph is constructed to represent the pages that are accessed together. The nodes in the dependency graph consists of pages present on the site, an arc connecting node A to node B indicates that the page B is likely to be accessed within next w-page access, after A is accessed, where w is the window size. Since this approach is implemented at the Web cache level, it is evaluated using two measures: the time required to access a page from the Web cache server, and the increase in the network traffic due to prefetching. They show that their scheme significantly reduces the average access time for a page, with only a marginal increase in the network traffic.

The approach presented in this article is similar to the variable length Markov models approach (VLMM) used in the linguistics field. This approach is also referred to as a variable n-gram model. Ron et al. [1996] were the first to propose a variable length Markov model. In their approach, higher-order states are pruned by computing the Kullback-Leibler distance on outgoing conditional probabilities. Siu and Ostendorf [2000] propose an extension of VLMM specifically targeted towards speech modeling that has the ability to capture the characteristics of spoken English. Salzberg et al. [1998] propose a VLMM, called interpolated Markov model (IMM), to identify genes in microbial genomes. In an IMM, instead of using a single state to make a prediction, the predictions from all the matched states are combined to make a prediction.

6. CONCLUSIONS

In this article, we presented a class of Markov model-based prediction algorithms that are obtained by selectively eliminating a large fraction of the states of the All-Kth-Order Markov model. These state-pruning approaches were motivated by the observation than as the order of the Markov model increases, the limited availability of large training datasets makes it impossible to accurately estimate the conditional probabilities of all the higher-order states. For some of these incorrectly estimated states, we may get better prediction accuracies by using lower-order states instead. Our experiments on a variety of datasets have shown that the resulting Markov models have a very low state-space complexity and, at the same time, achieve comparable or better accuracies than those obtained by the traditional algorithms.

APPENDIX

Significance Test for Comparing Accuracies: To get a better understanding of the differences in accuracies, we use *p-test* for comparing the accuracy values between two schemes [Yang and Liu 1999]. If we have two schemes A and B, which produce accuracies p_a and p_b, respectively, and the size of the test dataset is given as n, then we can use the standard normal distribution for the statistic Z,

$$Z = \frac{p_a - p_b}{\sqrt{2p(1-p)/n}}$$

where,

$$p = \frac{p_a + p_b}{2}.$$

Please note that the same test set was used for evaluating both the schemes.

REFERENCES

AGRAWAL, R., IMIELINSKI, T., AND SWAMI, A. 1993. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*. ACM Press, 207–216.

BESTRAVOS, A. 1995. Using speculation to reduce server load and service time on www. In *Proceedings of the 4th ACM International Conference of Information and Knowledge Management*. ACM Press.

BRIN, S. AND PAGE, L. 1998. The anatomy of large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference*.

CHI, E., PIROLLI, P., AND PITKOW, J. 2000. The scent of a site: A system for analyzing and predicting information scent, usage, and usability of a web site. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI 2000)*. 161–168.

CHI, E., PITKOW, J., MACKINLAY, J., PIROLLI, P., GOSSWEILER, R., AND CARD, S. 1998. Visualizing the evolution of web ecologies. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI 98)*.

COOLEY, R., TAN, P.-N., AND SRIVASTAVA, J. 1999. Websift: The web site information filter system. In *Proceedings of the Web Usage Analysis and User Profiling Workshop*.

DEAN, J. AND HENZINGER, M. R. 1999. Finding related pages in world wide web. In *Proceedings of the 8th International World Wide Web Conference*.

DUDA, R., HART, P., AND STORK, D. 2000. *Pattern Classification*. John WIley and Sons.

DURBIN, R., EDDY, S., KROGH, A., AND G., M. 1998. *Biological sequence analysis*. Cambridge University Press.

KLEINBERG, J. M. 1999. Authoritative sources in a hyperlinked environment. *J. ACM 46*, 5, 604–632.

KOHAVI, R. AND BRODLEY, C. 2000. Knowledge discovery and data mining cup *SIGKDD 2000*. *http://www.ecn.purdue.edu/KDDCUP/*.

KOHAVI, R., BRODLEY, C., FRASCA, B., MASON, L., AND ZHENG, Z. 2000. KDD-Cup 2000 organizers' report: Peeling the onion. *SIGKDD Explor. 2*, 2, 86–98.

LINTON, F. 2000. Owl: A recommender system for organization-wide learning. *J. Inter. Forum Educat. Tech. Soc.*

PADMANABHAM, V. AND MOGUL, J. 1996. Using predictive prefetching to improve world wide web latency. *Comput. Commun. Rev*.

PAPOULIS, A. 1991. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill.

PIROLLI, P., PITKOW, J., AND RAO, R. 1996. Silk from a sow's ear: Extracting usable structures from the web. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI-96)*.

PITKOW, J. AND PIROLLI, P. 1999. Mining longest repeating subsequence to predict world wide web surfing. In *2nd USENIX Symposium on Internet Technologies and Systems*. Boulder, CO.

RON, D., SINGER, Y., AND TISHBY, N. 1996. The power of amnesia: Learning probabilistic automata with variable memory length. *Mach. Learn.*

SALZBERG, S. L., DELCHER, A. L., KASIF, S., AND WHITE, O. 1998. Microbial gene identification using interpolated markov models. *Nucleic Acids Research*.

SARUKKAI, R. R. 2000. Link prediction and path analysis using markov chains. In *9th International World Wide Web Conference*.

SCHECHTER, S., KRISHNAN, M., AND SMITH, M. D. 1998. Using path profiles to predict http requests. In *7th International World Wide Web Conference*.

SIU, M. AND OSTENDORF, M. 2000. Variable n-grams and extensions for conversational speech language modeling. *IEEE Trans. Speech Audio Process. 8*, 1 (Jan), 63–75.

SRIVASTAVA, J., COOLEY, R., DESHPANDE, M., AND TAN, P.-N. 2000. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explor. 1*, 2.

THEUSINGER, C. AND HUBER, K.-P. 2000. Analyzing the footsteps of your customers. In *WebKDD 2000*.

YANG, Y. AND LIU, X. 1999. A re-examination of text categorization methods. In *the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Received May 2002; revised December 2002; accepted February 2003