Predicting Users' Requests on the WWW

I. Zukerman, D.W. Albrecht and A.E. Nicholson
School of Computer Science and Software Engineering
Monash University
Clayton, VICTORIA 3168, AUSTRALIA
phone: +61 3 9905-5202 fax: +61 3 9905-5146
{ingrid,dwa,annn}@csse.monash.edu.au

Abstract. We describe several Markov models derived from the behaviour patterns of many users, which predict which documents a user is likely to request next. We then present comparative results of the predictive accuracy of the different models, and, based on these results, build hybrid models which combine the individual models in different ways. These hybrid models generally have a greater predictive accuracy than the individual models. The best models will be incorporated in a system for pre-sending WWW documents.

1 Introduction

Users typically have to wait for information they require from the World Wide Web (WWW). The eventual aim of this project is to develop a system that reduces a user's expected waiting time by pre-sending documents s/he is likely to request (Nicholson et al., 1998,Albrecht et al., 1999). This requires the development of models which can anticipate a user's requests on the WWW. In this paper, we consider several such models (Sections 4 and 5), and compare their predictive power and their efficiency in terms of time and space consumption (Section 6). Our models are based on observing the behaviour patterns of many users, rather than modeling the requirements of an individual user. This is due to the constantly changing population of casual visitors to most WWW sites, in particular the site from which we gather our data (the School of Computer Science and Software Engineering at Monash University). Our models are generated by considering different combinations of two main features of our observations: the order in which documents are requested and the structure of the server site. These combinations yield four basic Markov models: Time, Space, Second-order Time, and Linked Space-Time (Section 4). The best of these models are then combined to yield hybrid models (Section 5).

In the next section we describe related research. We then describe the features of our domain, followed by our prediction models. Finally, we discuss our results, which motivate our hybrid models, and present concluding remarks.

2 Related Work

The recent growth in the WWW and on-line information sources has inspired research on agents that help users derive the most benefit from the vast quantities of available information. These agents may be broadly classified into *recommender systems*, which recommend information items that are likely to be of interest to the user, and *action systems*, which go one step further, performing actions on the user's behalf. Examples of recommender systems are WebWatcher (Joachims et al., 1997) and Letizia (Lieberman, 1995); examples of action systems are those described in (Bestavros, 1996,Balabanović, 1998). Both types of systems require a prediction model which anticipates a user's preferences, including documents a user may find interesting,

or his/her future actions. These models are generally obtained by applying machine learning techniques to identify these preferences or future actions based on the preferences or actions of (1) the users themselves (Davison and Hirsch, 1998,Joachims et al., 1997,Lieberman, 1995), (2) a group of similarly-minded users (Alspector et al., 1997), or (3) the general population (Bestavros, 1996,Albrecht et al., 1998).

Our system, which predicts web pages of interest to a user based on behaviour patterns of the general population, is most closely related to Bestavros'. However, Bestavros' system features one prediction model only – a Time-Markov model that predicts the probability of a future document request. In contrast, we are interested in comparing the accuracy of different predictive models. The prediction of the next request, rather than a future request, is the simplest basis for this comparison. The most accurate model will then be used to predict future requests.

3 The Domain

Analysis of information obtained from our WWW server yields the following features. (1) We can observe only one type of action performed by a user, namely a document request in the WWW (and our aim is to predict the user's next request). (2) It is extremely difficult to obtain a perspicuous representation of the domain. Typically there are huge numbers of documents located on a server and many links between them (there are also links to and from pages in external locations); the existence, location and size of documents are subject to continual change, as are the links between documents. (3) There is no obvious clear objective that applies to all users — some users may be browsing, others may be seeking specific information; also, there may be many ways to achieve an objective, since there may be many paths from a document to the desired information. (4) The sequence of requests from a user observed by the server providing the documents is only a partial record of the user's movements through the internet, since not all the user's movements to external locations are observed. (5) Finally, most WWW browsers and proxy servers cache documents received by a user. Thus, a user's requests for previously supplied documents that are still in the cache will not be observed.

The server logs the document requests which were satisfied, where a request takes the form {`client referer requestedDoc time size`}. The `client` is the internet server site that made the request. The `referer` is the current internet location of the user requesting the document, which may have one of the following values: (1) the http address of a local location, i.e., a (previously requested) web page on our server site; (2) the http address of an external location, i.e., a web page on another internet site; or (3) empty (represented by `'-'`), because the information has not been provided. The `requestedDoc` is the http address of the document being requested by the client. The `time` indicates when the request was received (measured in seconds elapsed since the startup of the system). The `size` is the number of bytes in the requested document. The requests are grouped into sessions, so that each session contains the temporal sequence of requests from a single client. This grouping supports the development of request models based on the temporal sequence of requested documents, i.e., Markov models (Section 4).

During pre-processing we perform the following actions to reduce the distortion of prediction models due to server traffic generated by certain WWW phenomena, the existing client-server protocol, or the configuration of the WWW at the server site. (1) We remove data generated by search engines and sessions identified as originating from a web-crawler client. (2) We remove instances of self-referring documents, since the requested document is already in the client's cache. (3) We infer implicit document requests within our server site; these are requests which were not logged by our current data-logging protocol (since the requested document is already

in the client's cache), but must have occurred to enable a particular sequence of events to take place. Inferred requests can be incorporated into our document prediction model (see Section 4.2 for an example), but not into our time prediction model (Section 4.1), since we do not know when an inferred request was made. (4) Finally, we take into account documents embedded in a main document, e.g., images embedded in text. These embedded documents are automatically requested by the main document within a few seconds after the main document is requested. Embedded documents must be identified when building a prediction model, since on one hand, they can almost never be pre-sent before they are requested (hence their incorporation in prediction models does not enhance these models, while slowing down the computations), and on the other hand, these documents must be pre-sent when pre-sending any document which contains them.

Our web site has over 200 personal pages plus hundreds of pages which contain coursework, research and administration information. These documents are organized in a complex lattice with many links connecting between pages of different types. The results presented in this paper are based on logs of web-page requests recorded by our server over a 50-day time window. After pre-processing, the following data were obtained: 1,095,730 document requests, where 59,486 clients requested 17,332 documents from 21,692 referer locations. The data include 14,023 referers which are also requested documents, and 103,972 different referer/document combinations.

4 Prediction Models

The models described in this section predict the document requested next. That is, they estimate $P(D_{R_1}, T_{R_1}|\text{previous requests})$, where D_{R_1} is the next document requested, and T_{R_1} is the time D_{R_1} is requested. In order to make the prediction problem computationally tractable, we assume that the distribution of the time for requesting a document is independent of the actual document that is requested, that the next document requested depends only on the previous documents requested, and that the time of the next request depends only on the time of the last request, T_R. This last assumption over-simplifies our domain, since the size of a document affects both its transmission time and the user's reading time, thereby influencing the time of the next request. In the future, we intend to factor the size of a document into the estimation of the time of the next request. According to our assumptions

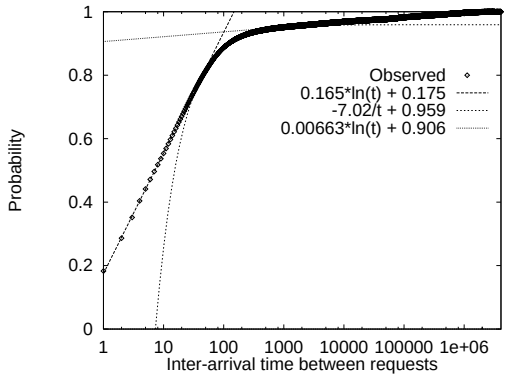
$$P(D_{R_1}, T_{R_1}|\text{previous requests}) = P(D_{R_1}|\text{previous documents}) \times P(T_{R_1}|T_R) \ .$$

The estimation of $P(T_{R_1}|T_R)$ is described in Section 4.1 and that of $P(D_{R_1}|\text{previous documents})$ in Section 4.2.

4.1 Next document is requested at a particular time

For our current database (based on 50 days of data), the time between successive requests from a client ranges from 0 to 4,100,910 seconds ($\sim$ 47 days): $0 \leq T_{R_1} - T_R \leq 4,100,910$.

Figure 1(a) shows the cumulative frequency distribution of the inter-arrival time between consecutive requests (plotted against a log scale). This distribution indicates that approximately 90% of document requests from a client are made within 122 seconds of the previous request, 95% are made within 874 seconds, and 99% within 343,412 seconds. As shown in Figure 1(a), a combination of three functions provides a good fit for the data (these functions were found using a weighted least-squares method). Therefore, we use the probability function in Figure 1(b) to estimate the probability of receiving a request at a particular time.



(a) Cumulative frequency distribution

$$\Pr(T_{R_1} - T_R < t) =$$

$$\begin{cases} 0.165 \times \ln(t) + 0.175 & 1 \leq t \leq 45 \\ -7.02/t + 0.959 & 45 < t \leq 960 \\ \min\{0.00663 \times \ln(t) + 0.906, 1\} & t > 960 \end{cases}$$

(b) Fitted probability function

Figure 1. Document requests at a particular time: (a) the cumulative frequency distribution plotted against a log scale of the inter-arrival time between requests (i.e., $T_{R_1} - T_R$) and fitted with three functions; and (b) the fitted probability function for $T_{R_1} - T_R$.

4.2 A particular document is requested next

In our earlier work, we considered a *Time Markov* model, which predicts a user's next request based only on the document that was requested last (Nicholson et al., 1998). Further analysis of the data logs of access to our site points to the importance of the structure of the site for building accurate prediction models. In this section, we introduce three additional prediction models, *Space Markov, Second-order Time Markov* and *Linked Space-Time Markov* (also called *Linked Markov*), and give a graphical representation for all four models. The Time and Second-order Time Markov models consider temporal information only; the Space Markov model considers structural information; and the Linked model combines temporal and structural information.

The **Space Markov model** was motivated by the observation that normally people follow links on web pages. Hence, in this model, the probability of a client requesting a document depends only on the referring document, which has a link to the requested document. In the **Second-order Time Markov model**, the probability of a client requesting a document depends on both the last requested document and the document requested before that. Finally, in the **Linked Space-Time Markov model**, the probability of a client requesting a document depends on both the last visited document and the referring document of the last visited document. Like the Second-order Time Markov model, this model considers two information items, but these items may be obtained from a single request record.

We use a graphical representation for each document prediction model, where the graph represents the probability that a document D_i is requested after an event E_{i-1}. The graph corresponding to each model contains a vertex for each event E_{i-1} and each requested document D_i observed in the training data. If a client's request for D_i was preceded by event E_{i-1} during a session, then there is an arc in the graph from E_{i-1} to D_i. In this case, we say that D_i is a successor of E_{i-1} ($D_i \in succ(E_{i-1})$). For the Time Markov model, the event of interest is the last document requested (D_{i-1}), with an arc from D_{i-1} to D_i indicating that document D_i was requested after D_{i-1}. For the Space Markov model, E_{i-1} is the referring document of D_i (D_{Ref_i}), with an arc from D_{Ref_i} to D_i indicating that D_i was reached through a link from D_{Ref_i}. For the Second-order Time Markov model, E_{i-1} is a tuple which contains the last two docu-

ments requested ($\{D_{i-2}, D_{i-1}\}$), with an arc from $\{D_{i-2}, D_{i-1}\}$ to D_i indicating that a request for document D_i was preceded by a request for D_{i-1} which in turn was preceded by a request for D_{i-2}. Finally, for the Linked Space-Time Markov model, E_{i-1} is a tuple that contains the last document requested and its referer ($\{D_{Ref_{i-1}}, D_{i-1}\}$), with an arc from $\{D_{Ref_{i-1}}, D_{i-1}\}$ to D_i indicating that a referral from $D_{Ref_{i-1}}$ to D_{i-1} was followed by a request for D_i.

Each arc from event E_{i-1} to D_i has an associated weight, $w(E_{i-1}, D_i)$, which is the frequency of an event-document pair across all sessions. Thus, after observing an event E, the probability that the next requested document is D can be computed as follows.

$$\Pr(D_{R_1} = D|E) = \frac{w(E,D)}{\sum_{D_j \in succ(E)} w(E,D_j)} .$$

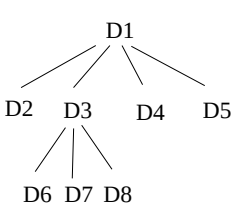
To illustrate these models and the manner in which they are built, consider a fragment of training data from a client who visits the documents in the WWW site shown in Figure 2(a) in the order indicated in Figure 2(b). The document to the left of each arrow is the referring document, the document to the right of the arrow is the next document, and the time stamp indicates when the requests were made. Note that the first and last requests have no referring documents, because the client's browser has not supplied this information to the server. For this example, we assume that D6 and D7 were always visited whenever D3 was visited, and these visits were performed shortly after visiting D3. Hence, D6 and D7 are considered embedded documents of D3, forming one document with D3. This is not the case for D8, which is visited sometime after D3, and not every time D3 is visited. After applying the pre-processing operations described in Section 3 to this sequence of requests, we obtain the extended log in Figure 2(c), where the steps inferred from those actually logged appear in boldface, i.e., the referrals **D2** $\rightarrow$ **D1** and **D8** $\rightarrow$ **D1**, and the embedded documents, i.e., *D3;D6;D7*, appear in italics. This extended log is used to build the graphs corresponding to our four models as follows.

The graph corresponding to the Time Markov model (Figure 2(d)) is built by following the sequence of documents to the right of the arrow, viz D1, D2, **D1**, *[D3;D6;D7]*, D8, **D1**, D4, D5.[1,2] The graph that represents the Space Markov model (Figure 2(e)) is built using the {referer, document} pairs in each line, viz $-$ $\rightarrow$D1, D1$\rightarrow$D2, **D2**$\rightarrow$**D1**, D1$\rightarrow$*[D3;D6;D7]*, *[D3;D6;D7]*$\rightarrow$D8, **D8**$\rightarrow$ **D1**, D1$\rightarrow$D4, $-$ $\rightarrow$D5. This graph, which represents structural information in the WWW server site, is the same as the graph in Figure 2(d) when the links in the current site are followed; the graphs differ when there is no referral log for a requested document, as in the first and last lines in Figure 2(c), or when the referer is an external location. The graph that represents the Second-order Time Markov model (Figure 2(f)) is built using event-document pairs, where the events are composed of two consecutive documents to the right of the arrow, which in turn precede the next document, i.e., {D1, D2} **D1**, {D2, **D1**} *[D3;D6;D7]*, {**D1**, *[D3;D6;D7]*} D8, {*[D3;D6;D7]*, D8} **D1**, {D8, **D1**} D4, {**D1**, D4} D5. Finally, the graph that represents the Linked Markov model (Figure 2(g)) is built from {preceding-referer, preceding-request} events, which precede the next requested document, viz {$-$ $\rightarrow$ D1} D2, {D1$\rightarrow$D2} **D1**, {**D2**$\rightarrow$**D1**} *[D3;D6;D7]*, {D1$\rightarrow$*[D3;D6;D7]*} D8, {*[D3;D6;D7]*$\rightarrow$D8} **D1**, {**D8**$\rightarrow$**D1**} D4, {D1 $\rightarrow$ D4} D5.

To illustrate the calculation of the probability of requesting a particular document after observing an event, let us reconsider the Time Markov model in Figure 2(d), and assume that

[1] This graph also contains weights for the arcs, which are obtained from frequency counts of pairs of consecutively requested documents in the training data. Similar frequency counts may be obtained for the other models, but are not required for this exposition.

[2] The inferred links are drawn in thick lines in the graphs representing the Time and Space Markov models.



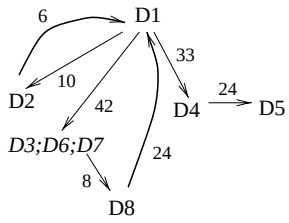
(a) Site structure

referer	requestedDoc	time
—	D1	261
D1	D2	270
D1	D3	300
D3	D6	301
D3	D7	301
D3	D8	320
D1	D4	350
—	D5	396

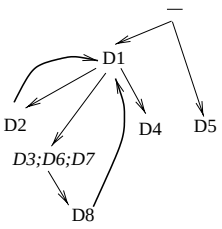
(b) Record of document requests

referer	requestedDoc	time
—	D1	261
D1	D2	270
D2	**D1**	
D1	*D3;D6;D7*	300
D3;D6;D7	D8	320
D8	**D1**	
D1	D4	350
—	D5	396

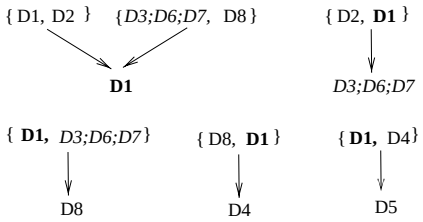
(c) Extended record of document requests



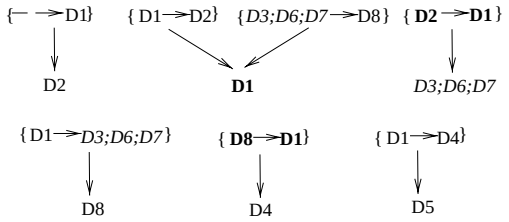
(d) Time Markov model



(e) Space Markov model



(f) Second-order Time Markov model



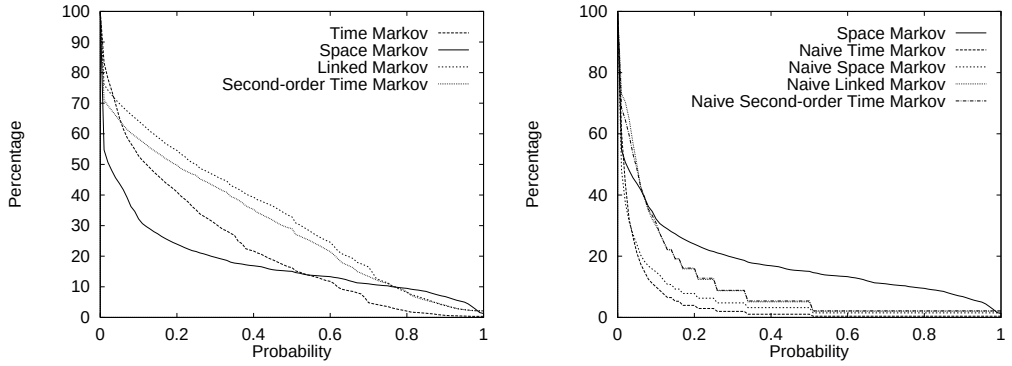
(g) Linked Space-Time Markov model

Figure 2. Sample Time, Space, Second-order Time and Linked Space-Time Markov models.

document D1 has just been requested. The model would then assign a zero probability to the next requested document being D8, D5 or a document not seen in training, and it would predict $\Pr(D_{R_1} = D2|D1) = \frac{10}{10+42+33} = 0.12$, $\Pr(D_{R_1} = \{D3; D6; D7\}|D1) = \frac{42}{10+42+33} = 0.49$, and $\Pr(D_{R_1} = D4|D1) = \frac{33}{10+42+33} = 0.39$.

5 Results

The results presented in this section were obtained from 50 days of data logged by our server. All the models were tested using 80% of the sessions for training and 20% for testing, with results showing averages from 10 runs. Differences noted in the results for the various prediction models are significant at the 5% level.



(a) Basic Markov models (b) Naive Markov models

Figure 3. Comparison of the prediction probabilities obtained with: (a) the basic Markov models, and (b) the naive Markov models.

The prediction models were assessed in terms of the probability with which they predict the actual next request. Figure 3(a) compares the prediction probabilities obtained with our four Markov models. The x-axis shows the probability with which a model predicts the next request made by the user. The y-axis shows the average percentage of predictions whose probability is greater than or equal to the probability shown on the x-axis. For example, the Linked Markov model predicts the actual next request with probability greater than or equal to 0.5 32.9% of the time, while the Second-order Time Markov model predicts the next request with probability greater than or equal to 0.5 29.0% of the time.

The results in Figure 3(a) indicate that the predictive performance of the Time Markov model is better than that of the other models when the probability of this prediction is low (≤ 0.06). When the prediction probability is between 0.06 and 0.77, the Linked model has the highest predictive accuracy. The relative performance of the Space Markov model improves from a prediction probability of 0.06 to 0.77, at which point it overtakes the Linked Markov model. This means that the Space Markov model is more accurate than the other models when its predictions have a high probability (> 0.77). Still, the overall performance of the Linked Markov model is better than that of the other models.

Since the predicted probabilities shown in Figure 3(a) seem quite low, we validate our models by comparing these probabilities with the predictions made by the naive counterparts of our models. The naive version of a model predicts the request which follows an event using a uniform distribution; that is, each of the n successors of a vertex which represents this event is predicted with a probability of $1/n$. This probability differs for each vertex, since each vertex may have a different number of successors. Figure 3(b) compares these naive predictions with those obtained using the Space Markov model, which generally gives the lowest predictions of all the basic Markov models.[3] For probabilities greater than 0.08, the Space Markov model is clearly a

[3] The variation in the predictive performance of the naive models is due to differences in the number of vertex successors typically found in the graph corresponding to each model. For instance, 92% of the vertices have 5 or less successors in the Second-order model, 6 or less successors in the Linked model, 9 or less in the Space model, and 15 or less in the Time Markov model. Although these branching factors seem low, each graph has many vertices with hundreds of successors.

better predictor than all the naive models, indicating that our models, which incorporate request frequency information into the arc weights, improve upon the naive predictions.

Hybrid Prediction Models Each prediction model is sometimes unable to make a prediction because the current situation was unseen during training ("% seen" column in Figure 4(b)). This adversely affects its predictive performance. We have designed two hybrid models, `maxHybrid` and `orderedHybrid`, to address this problem.

The `maxHybrid` model consults all the Markov prediction models, and makes its own prediction using the model which made a prediction with the highest probability, i.e., the model which has the most confidence in its most likely prediction.

The `orderedHybrid` model consults the Markov models in the following order, which was determined based on the relative performance of these models: Linked, Second-order, Time and Space. The first model which can make a prediction is selected.

Finally, as seen in Figure 3(a), when the Space Markov model predicts the actual next request with a high probability, its predictive performance is better than that of the other Markov models. This is the basis for the `spaceLinkedHybrid` model, which combines the Space and Linked Markov models as follows:

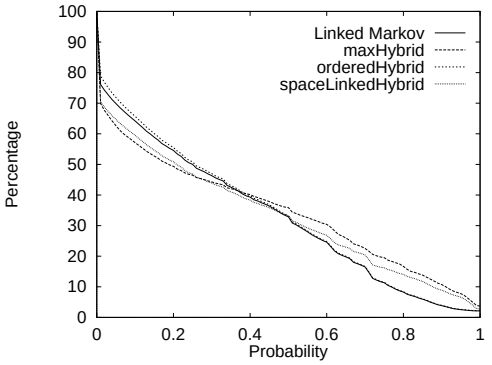
If the maximum prediction made by the Space Markov model is > 0.77, then use its predictions, otherwise use those of the Linked Markov model.

The results obtained with these hybrid models are shown in Figure 4(a) compared to the overall best of the individual prediction models, the Linked Space-Time Markov model. The predictive accuracy of the `orderedHybrid` model is higher than that of the other models until the probability reaches 0.39, at which point the `maxHybrid` model starts performing better than the other models. For probabilities greater than 0.5 all the hybrid models perform significantly better than the Linked Markov model.

We postulate that the `maxHybrid` model performs better when the prediction probability is relatively high (> 0.39), because such a probability indicates that the basic Markov model which was selected has some information about the user's behaviour. In contrast, a "highest" prediction probability that is relatively low indicates that all the candidate models are rather uninformed. In this case, the `orderedHybrid` model performs better because it primarily relies on the Linked Markov model, which is the best of the basic models (other models are consulted only when the Linked Markov model cannot make a prediction). As expected, the relative performance of the `spaceLinkedHybrid` model improves as the prediction probabilities increase. Still, even for high probabilities, its performance falls below the performance of the `maxHybrid` model. In summary, due to the improved predictive accuracy of the `maxHybrid` model in the higher probability ranges, we consider it the best prediction model for a document pre-sending system.

6 Discussion

The two main features of our domain are the spatial structure of the server site and the order in which documents are requested. The results in the previous section show that the Linked Markov model, which combines these two features, gives better predictions overall than the Markov models which incorporate only one of these features. The inability of the basic models to make a prediction some of the time motivated the development of the `orderedHybrid` and `maxHybrid` models, while the good performance of the Space Markov model when its predictions have a high probability motivated the `spaceLinkedHybrid` model. As shown in the previous section, the `maxHybrid` model is the most suitable for a document pre-sending system.



(a) Hybrid models

Markov Models			
model	#vertices	#arcs	% seen
Time	17,298	125,897	91.1
Space	25,000	103,972	60.3
Linked	101,110	271,022	78.6
Second-order	123,227	287,924	73.3

Hybrid Models			
model	#vertices	#arcs	% seen
`maxHybrid`	266,635	788,815	91.5
`orderedHybrid`	266,635	788,815	91.5
`spaceLinkedHybrid`	126,110	374,994	78.9

(b) Model size

Figure 4. (a) Comparison of the prediction probabilities obtained with hybrid models. (b) Features of the prediction models: maximum number of vertices and arcs, and average percentage of test data for which a model is able to make a prediction.

We now examine how the following factors affect the predictive performance of our models: time and space resources required, coverage of the test data, and model selection policy (for the hybrid models).

The computation time required by the prediction models is as follows. The training time for each of the four basic Markov models is about 700 seconds of CPU time on a SGI Indy R5000. All the prediction models make fast predictions in the testing phase, from 1.6 to 2.0 ms/request.[4]

We measure the size of the prediction models by the number of vertices and arcs in their graphical representations (Figure 4(b)). While the larger models tend to have a better predictive performance than the smaller ones, size is not the only determining factor. For instance, the Linked Markov model is the best of the basic Markov models, even though it is not the largest.

The average percentages of the "seen" test data, i.e., the data for which the models are able to make a prediction, are shown in Figure 4(b). Increasing this percentage can improve predictive performance, e.g., the `maxHybrid` model compared to the Linked Markov model. However, as shown by the relatively poor performance of the Time Markov model, a high proportion of seen data does not in itself improve the predictive accuracy of a model.

The `maxHybrid` model and the `orderedHybrid` model, which have the same number of vertices and arcs and the same percentage of seen data, have the best predictive performance under different circumstances. The dynamic model selection performed by the `maxHybrid` model yields better results for higher prediction probabilities, while the static selection performed by the `orderedHybrid` model (based on the previous relative performance of the basic models) yields better results for lower probabilities.

In summary, the predictive performance of the models is affected by the following interrelated factors: (1) domain features – temporal order and spatial structure; (2) size – number of vertices and arcs in the models' graphical representations; and (3) coverage of test data – percentage of seen data. The model selection policies applied by the hybrid models also affect the performance of these models.

[4] We would expect the models with the lowest branching factor to be the fastest, however the Second-order model is the slowest due to the way in which the database lookup was implemented.

7 Conclusion and Future Work

We have compared prediction models that take into account two factors in isolation or in combination: (1) the order in which documents are requested, and (2) the structure of the server site. We have shown that the combination of these factors in the Linked Markov model yields the greatest predictive power among the basic prediction models. Further, combining more than one prediction model in a hybrid model overcomes the problem of unseen data to some extent, resulting in models that are better than the individual models. This improvement is achieved through the use of more space and time resources, however the space requirements are not prohibitive and the predictions can still be made in milliseconds.

We are currently investigating the incorporation of our best prediction model, the `maxHybrid` model, into a document pre-sending system (Albrecht et al., 1999). In the future, we intend to extend the best models to predict a user's future requests (rather than the next request). Such predictions are required to pre-send documents that are not immediately needed by a user, but may be required later on.

Finally, we plan to identify user profiles based on a classification developed using unsupervised learning techniques (such as those used by Albrecht et al. (1998) to classify actions). To this effect, we will consider attributes such as inter-request time, type of document requested, whether the referer is internal or external to our site, time of day, and depth of the document in the server site document hierarchy. The resulting classification will support the tailoring of prediction models to different types of user, which should yield improved predictive performance.

Acknowledgments

This research was supported in part by grant A49600323 from the Australian Research Council.

References

Albrecht, D. W., Zukerman, I., and Nicholson, A. E. (1998). Bayesian models for keyhole plan recognition in an adventure game. *User Modeling and User-Adapted Interaction* 8(1-2):5–47.

Albrecht, D. W., Zukerman, I., and Nicholson, A. E. (1999). Pre-sending documents on the WWW: A comparative study. In *IJCAI99 – Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*.

Alspector, J., Koicz, A., and Karunanithi, N. (1997). Feature-based and clique-based user models for movie selection: A comparative study. *User Modeling and User-Adapted Interaction* 7(4):279–304.

Balabanović, M. (1998). Exploring versus exploiting when learning user models for text recommendation. *User Modeling and User-adapted Interaction* 8(1-2):71–102.

Bestavros, A. (1996). Speculative data dissemination and service to reduce server load, network traffic and service time in distributed information systems. In *Proceedings of the 1996 International Conference on Data Engineering*.

Davison, B., and Hirsch, H. (1998). Predicting sequences of user actions. In *Notes of the AAAI/ICML 1998 Workshop on Predicting the Future: AI Approaches to Time-Series Analysis*.

Joachims, T., Freitag, D., and Mitchell, T. (1997). WebWatcher: A tour guide for the World Wide Web. In *IJCAI97 – Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, 770–775.

Lieberman, H. (1995). Letizia: An agent that assists web browsing. In *IJCAI95 – Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 924–929.

Nicholson, A. E., Zukerman, I., and Albrecht, D. W. (1998). A decision-theoretic approach for pre-sending information on the WWW. In *PRICAI'98 – Proceedings of the Fifth Pacific Rim International Conference on Artificial Intelligence*, 575–586.