Journal of Web Engineering, Vol. 16, No.5&6 (2017) 524-536

WEB ACCESS MINING THROUGH DYNAMIC DECISION TREES WITH MARKOVIAN FEATURES

ARPAD GELLERT
Computer Science and Electrical Engineering Department, Lucian Blaga University of Sibiu, Romania
E-mail: arpad.gellert@ulbsibiu.ro

Received June 2, 2016
Revised May 25, 2017

In this work we propose a hybrid web access prediction method consisting in a dynamic decision tree and different order Markov predictors as components. The predictions generated by the Markov chain components are used as features within the dynamic decision tree. Our goal is to use this hybrid technique in order to anticipate and prefetch the web pages and files accessed by the users through browsers, reducing thus the load times. We use a decision tree to select the most predictive features from a considered feature set and based on those selected features we generate predictions. In our application, the feature set includes the current link, the type of the current link as well as the predictions of different order Markov chains. The optimal configuration of the proposed hybrid technique provides an average web page prediction accuracy of 72.57%.

Key words: Web page prediction, web prefetching, Markov chains, dynamic decision tree, browser extension.
Communicated by: B. White & M. Bielikova

1 Introduction

Several low-latency web access techniques have been proposed during the last years. Caching exploits the temporal locality principle by keeping the accessed pages and files in a cache structure. Prefetching anticipates the next accesses and loads the corresponding web pages or files into the cache. If the user accesses a web page or a file which is already in the cache, the browser can load it without any delays. Thus, when the users have long browsing sessions, prediction-based prefetching can be very effective by minimizing the access latencies. In this work, we investigate web prefetching through dynamic decision tree (DDT) which acts like a hybrid web predictor having different order Markov chains as components. The idea of using such a hybrid predictor is based on our observation that in different log files, due to different user behaviours, Markov chains of different orders can predict better. Thus, we expect that a hybrid predictor, which can dynamically select the most appropriate component, could provide better prediction accuracy than any of the components considered alone. This dynamic component selection is performed, in our case, by the DDT. The novelty of the proposed method is that we use the predictions generated by the component Markov predictors as DDT features to anticipate the next accessed link.

The decision tree is a commonly used inductive inference machine learning method and due to its simplicity and accuracy it is best suited for our needs. Decision trees can classify items by keeping them sorted from the root to the leaf nodes [15]. Each node in the tree denotes a test of some feature and each

descending branch corresponds to one possible value of that certain feature. An item is classified by testing the feature specified by the root and moving down on the branch corresponding to the value of that feature. The process is repeated in a recursive manner on the current subtree. The value of the leaf node situated at the end of the selected path represents the output of the decision tree.

The Markov chains are using as input the web access history of a certain size consisting in a codified web sequence. We consider as context the most recently accessed web files. Thus, a Markov chain of order R is using as context the R previous web files accessed by the user and predicts the web file that followed that context with the highest frequency as being the next access. For efficiency, the Markov chain is implemented as simple searches in the observation sequence, as in [8], instead of using high complexity tree-based modelling.

We use the decision tree to dynamically select the most predictive features from a considered feature set and we consider only the selected features to generate predictions. In our application the feature set includes the current link, the type of the current link (HTML, non-HTML) as well as the predictions of different order Markov chains. The first step of the algorithm is to choose the root of the tree from the existing attributes. That chosen attribute has the highest information gain on the entire set of examples and it does best classify this set. After that, for each possible value of the selected attribute a new attribute will be recursively chosen from the remaining attributes as the root of the subtree that will be created as a branch. After the tree is learned we will use it to predict and prefetch the next web page or file based on the current context. For a higher prediction accuracy, we have also attached a dynamic confidence mechanism to the predictor. We have evaluated the prediction accuracy of the proposed hybrid predictor on the Boston University benchmark set (BU), generated by the "Ocean Group Research" [4].

The rest of this paper is organized as follows. Section 2, is reviewing the related work in web prefetching with the focus on Markov chains, decision trees and hybrid methods. Section 3 is describing our proposed Markovian DDT predictor. In Section 4 we are presenting the experimental results. The conclusions as well as the further work directions are discussed in Section 5.

2 Related Work

In our previous work [7] we compared Markov chains, Hidden Markov Models and graph algorithms as web page prediction methods from their accuracy point of view. We applied multi-page prediction by prefetching all the web files occured in the user access history after the considered context. The best prediction accuracy has been obtained with a hybrid predictor consisting in a HMM and a graph-based predictor. The proposed hybrid predictor prefetched the web pages anticipated by both component predictors. In contrast, in this work we use DDT-based prediction which combines different order Markov chains, exploiting thus the advantage of each one. In order to keep low network traffic, in this work we predict and eventually prefetch only one web file at a time.

In [8] we have analyzed the prediction by partial matching (PPM) algorithm for web prefetching. We have implemented it in an efficient way, allowing thus long web access histories and higher order component Markov chains at low complexity. The evaluations performed on the BU dataset have shown that the optimal PPM configuration was the 4^{th} order one, with a prediction accuracy of 71.11%. The PPM is a hybrid predictor with prioritization-based static component selection. In contrast, in this work

we use a DDT-based dynamic selection in order to use the best component predictor at a certain moment, for higher prediction accuracy.

Link prediction based on Markov chains was presented in [21]. In [9] the author applied the Markov model together with the k-nearest neighbor classifier algorithm to enhance the performance of traditional Markov chains in predicting web pages. He obtained lower consumed memory, quite similar build time and evaluation time and higher accuracy. In [10] and [11] clustering is used to group homogeneous user sessions. Low order Markov chains are built on these clustered sessions and when Markov models cannot make clear predictions the association rules are used. The prediction accuracy was around 65%. In [12] the authors presented a survey of web page ranking for web personalization. They concluded that low order Markov models have higher accuracy and lower coverage, whereas the higher order models have a number of limitations associated with: higher state complexity, reduced coverage and sometimes even worse prediction accuracy. In [18], the authors combined All-k^{th} order Markov models with fuzzy Adaptive Resonance Theory for web page request prediction and obtained a prediction accuracy less than 40%. In contrast, we dynamically select the order of the Markov chain using a DDT and we expect a better adaptability to different users and also to changes in the behaviour of the same user.

In [5] the authors proposed a hybrid web page prediction method which combines support vector machines (SVM), association rule mining and Markov chains in order to enhance the efficiency. Their experimental results showed that the proposed hybrid method provided a prediction accuracy of about 63%, outperforming the individual predictors. Their method needs a pretraining stage, while our method is based only on run-time training and we expect faster adaptation to user behaviour changes. In [1] the authors proposed a three level technique in which they combined Markov models, association rule mining and clustering. The prefetching and prediction is done by preprocessing the user accesses and integrating the three mentioned techniques. They used only a first order Markov model, therefore we expect higher prediction accuracy with our superior order models. In [16] the authors proposed another hybrid predictor which combines Markov model and Hidden Markov Models (HMM). In our opinion, the complexity of a HMM-based prediction algorithm increases dramatically in the case of higher order model and high number of different accessed links.

In [17], Pabarskaite proposed decision trees for web user behaviour analysis including prediction of future actions. The author used the C4.5 decision tree algorithm which determines a set of rules that classify a data item into one of a set of predefined classes. Since the classes are associated to links, taking into account the high number of different visited web pages, some pages were grouped. The difference between our work and [17] is that we use also Markovian features and we predict and prefetch pages or files from any website instead of only one certain site. To keep the decision tree at reasonable size we limit the used web access history instead of grouping web pages. This web access history limitation can also help our DDT to adapt faster to the user behaviour changes. In [14], Lowd and Davis improve Markov network structure learning with decision trees. In their approach, a tree is learned to predict the value of each variable and then the trees are converted into sets of conjunctive features. All the learned features are merged into a single global model. The weights of these features are then learned globally by a Markov network. In contrast, in this work we include the predictions of different order Markov chains as features within a dynamically adapted decision tree which provides the final prediction. A temporal modelling of web user behaviour has been proposed by Radinsky et al. in [20] for future behaviour prediction. They presented several modelling techniques based on time-series to capture

trends, periodicities and surprises in web usage behaviour. A decision tree is learned during a training stage to choose the best performing temporal model. In contrast, in our work we use different order Markov predictors in a decision tree which dynamically learns which one is better in each context.

In [22] Temgire et al. presented a review on web prefetching techniques. Other recent studies on web prediction and prefetching have been published in [2], [3], [13], [23] and [24].

3 Web Prediction using Markovian DDT

The browser extension, presented in Figure 1, collects and preprocesses the links accessed by the user: each link is codified with a unique number, ports and relative parts are eliminated from the links, if there are two consecutive accesses of the same link only one is considered, and the links having *.gif* and *.xbm* extension, which are usually small images, are eliminated. The browser extension keeps a link history of size *H*. When the current link is accessed, it is introduced into the link history which can be truncated to the last *H* unique web accesses.

The goal is to anticipate the next link using the DDT algorithm based on the history of the previously visited links as well as the type of the current link. For this, the proposed hybrid technique is composed of two prediction levels. The link history is translated into Markovian features consisting in the predictions using Markov chains of different orders. These Markovian features are used together with the link type feature (HTML and non-HTML) and the current link feature as input for the DDT algorithm to generate the final prediction. The predicted web page or file is prefetched into the cache in order to be available if the user will access it in the future.

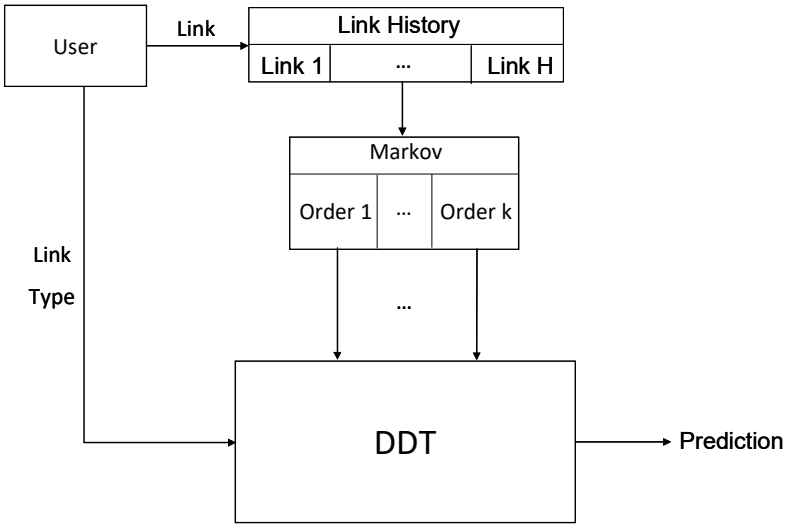

Figure 1. The structure of the application

For a higher prediction accuracy, the predictor is enhanced with a confidence mechanism as in [7], consisting in 4-state saturating counters attached to all the links kept in the history. The saturating counter associated to the current link is incremented on correct prediction and decremented on misprediction. A prediction is generated only if the confidence counter of the current link is in a

predictable state. For a low traffic level, we predict and eventually prefetch only one web file at a time, instead of a multi-page prediction presented in [7]. In our application, the states within the Markov chain are represented by the links (web pages and files). In a Markov chain of order R, the current link depends on R previous links [7]:

$$P_i[q_t = L_i \mid q_{t-1},\ ...,\ q_{t-H}] = P_i[q_t = L_i \mid q_{t-1},\ ...,\ q_{t-R}],\ i \in [1, N] \quad (1)$$

where q_t is the link at time t and P_i is the probability of the link (web page) L_i at time t, N is the number of distinct web pages and H is the size of the used web access history.

The Markov chain is implemented as in [8] as simple searches for the current context within the observation sequence, represented by the history of visited web pages. The web page that followed the context with the highest frequency is the predicted one. In a Markov chain of order R, the context consists in the last R web pages. The prediction of the R^{th} order Markov chain is computed as follows:

$$M_R = L_i,\ P_i = \max_{1 \le k \le N} P_k[q_t = L_k \mid q_{t-1},\ ...,\ q_{t-R}] \quad (2)$$

The Java implementation of this Markov predictor is given in [6]. Now we will describe the DDT method for predicting the next web page. We have used the ID3 learning algorithm [19] to create the decision tree from a training set of examples. Then we have used this tree to predict the next web page to be prefetched.

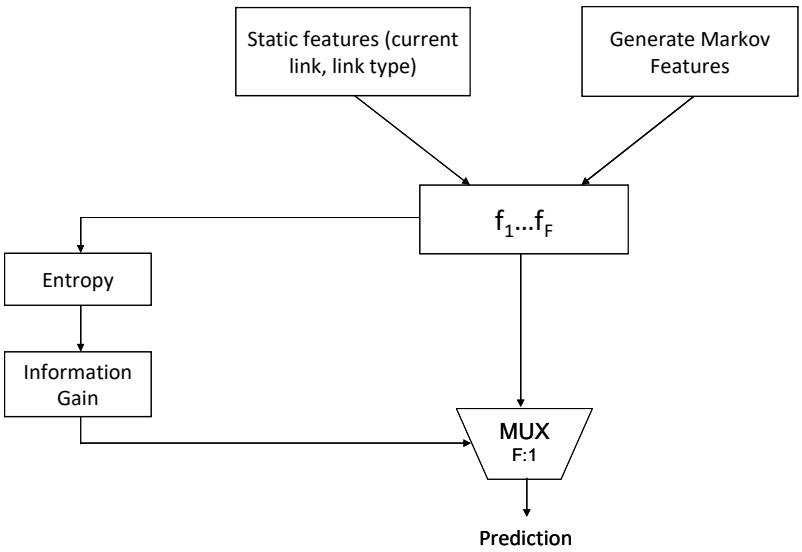

Figure 2. The structure of the DDT

Figure 2 presents the structure of our DDT. As it can be seen, we use two types of features, the static ones, which are the current link and the link type taken from the server logs, and the dynamic ones generated at runtime which are the predictions provided by the Markov models. We considered in our evaluations Markov chains with the order ranging between 0 and 7 and we observed that the best results

were obtained with four Markov features: M_1, M_2, M_3 and M_4 (with M_R defined in equation (2)). As we detail below, the entropy and the information gain are used for feature selection.

The ID3 algorithm receives as parameters the set of examples S and the features F. It is a greedy algorithm that builds the decision tree T from top to down, by selecting at each node the feature that best classifies the training examples. The ID3 algorithm is given in the following pseudocode, where the union operation denotes node insertion:

```
ID3(S, F)
   T := Ø
   If F= Ø
      T := T ∪{the most common prediction in S}
   Endif
   Else
      Foreach Fk ϵ F
         Gmax := 0
         E(S) := Σ_{i=1}^{c} (−pi log2 pi)
         G(S, Fk) := E(S) − Σ_{v∈Values(Fk)} (|Sv|/|S|) E(Sv)
         If G(S, Fk) > Gmax
               N := Fk
               Gmax := G(S, Fk)
         Endif
      Endforeach
      T := T ∪{N}
      Foreach possible v of N
         If Sv = Ø
            T := T ∪{the most common prediction in S}
         Endif
         Else
            T := T ∪ID3(Sv, F-{N})
         Endelse
      Endforeach
   Endelse
   Return T
End
```

where c is the number of possible values of the target feature, N is the selected node, F denotes the features, S_v is the subset of the example set S for which feature F has the value v, E is the entropy and G is the information gain. The entropy represents the impurity of an example set. At the computation of the entropy, p_i is the proportion of S belonging to class i. The information gain is computed for each feature based on the entropy and it shows how well an attribute separates the training examples. The first feature that will be chosen does best split the set of examples S and will be the root of the tree. For each possible value of the selected feature a branch will be created whose descendent node will be recursively chosen from the remaining eligible features. The feature that has the highest information gain (meaning the lowest entropy) will be chosen as a node for the current branch. The tree is

reconstructed after 100 links and the tree root can change, so it can provide good knowledge on the current situation. After this learning step, we use the DDT to predict the next link.

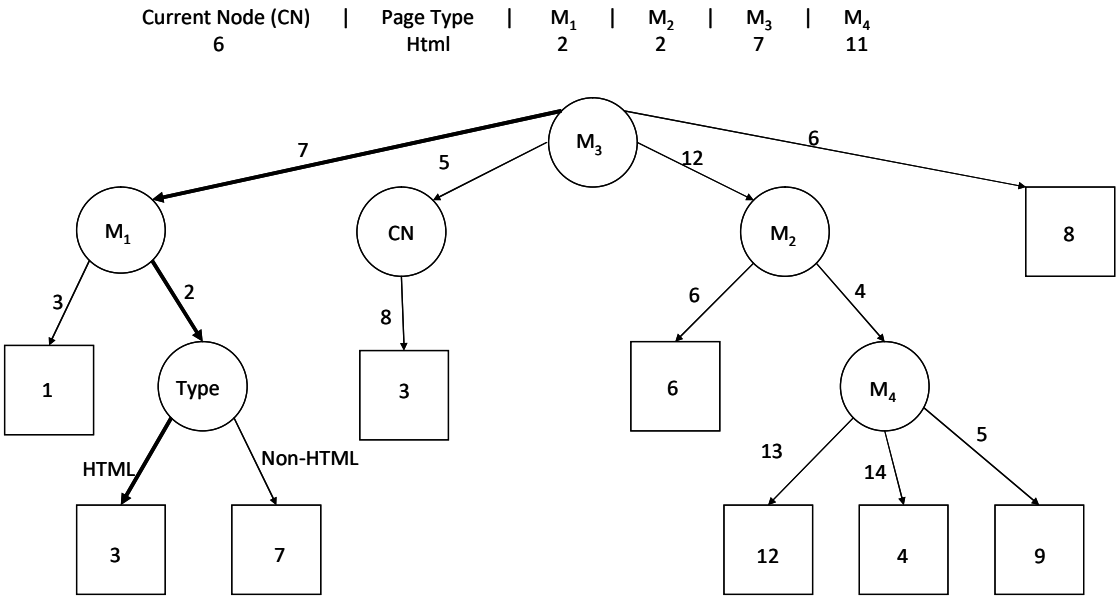

Figure 3. DDT Example

Figure 3 depicts a small example showing how a DDT-based prediction mechanism works by emphasizing the path from the root to the predicted leaf. The internal nodes (represented by circles) denote tests of the input features, the branches (represented by arrows) are the outcomes of the tests, whereas the leaf nodes (represented by squares) are the output classes. The feature with the highest information gain is M_3 and, when the input data is available, that feature will be checked first choosing thus the branch that corresponds to that specific value. In this case, we have for the feature M_3 the value 7, which corresponds to the left sub-tree in which the root is M_1. The value for the M_1 feature in this example is 2 and we go through that branch reaching the *Type* node. The value of this feature is HTML, meaning that we go through the left branch and reach the leaf node 3 which will be the predicted value for this case. Basically, this step is a tree traversal having as input the static data provided by the link and the dynamic data provided by the Markov chains.

4 Experimental Results

We used the BU dataset to evaluate the proposed method which was implemented in Java. The BU benchmark set was generated by the "Ocean Group Research" from Boston University [4]. Each log file name contains a user ID number, the machine on which the session took place and the Unix timestamp when the session started. Each line in a log corresponds to a single URL requested by the user; it contains the machine name, the timestamp when the request was made, the URL, the size of the document in bytes (including the overhead of the protocol) and the object retrieval time in seconds.

We used as measures the prediction accuracy, the coverage and the prediction rate. The prediction accuracy is computed by dividing the number of correctly predicted links to the number of predicted links. The coverage is the division between the number of correctly predicted links and the total number of links. The prediction rate is the number of predicted links divided to the total number of links. The performance of prefetching techniques can be measured based on the hit rate and the additional bandwidth. In this work, the hit rate is expressed as prediction accuracy (PA). Since the bandwidth increase is directly proportional with the prediction rate (PR), we can approximate the relative prefetching benefit of a certain method 1 compared to another method 2, as being:

$$B = \frac{\frac{PA_1}{PR_1}}{\frac{PA_2}{PR_2}} \tag{3}$$

Obviously, there is a benefit if B is greater than 1.

For all the evaluated methods we used a 4-state confidence mechanism as in [7]. First, we analyzed the proposed Markovian DDT algorithm from the prediction accuracy point of view, by varying its input parameters.

Figure 4 compares DDTs composed of different order Markov predictors using a history of 1000 links and 4-state confidence counters. DDT-M1 contains only a Markov predictor of order 1, DDT-M1-2 contains Markov predictors of orders 1 and 2, and so on. We denote DDT-Mx-y a decision tree using as features the predictions generated by Markov chains having the order between x and y. As Figure 4 shows, the inclusion of Markov chains having the order higher than 4 into the hybrid predictor is not improving the prediction accuracy, meaning that usually the web access patterns are not longer than 4. The DDT-M1-4 which uses the Markov predictors having the orders between 1 and 4 is the optimal since the inclusion of higher order Markov chains is not improving significantly the accuracy.

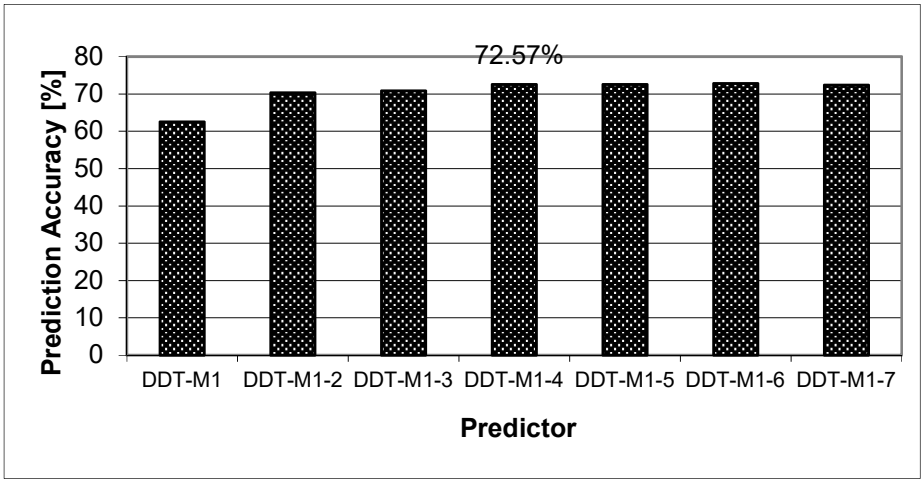

Figure 4. Average prediction accuracies obtained with different DDT predictors

Figure 5 compares the optimal DDT-M1-4 predictor with the Markov predictors having the orders from 0 to 7. All these compared predictors are using a history of 1000 links and 4-state confidence counters. As Figure 5 shows, the prediction accuracy is increasing together with the order of the Markov chain up to order 2 after which we can observe a decrease tendency, especially beyond order 4, which explains again the superiority in accuracy of DDT-M1-4 in Figure 4. In fact, we observed that starting with the Markov chain of order 6, the number of predictions was 0 on multiple benchmarks because the too rich contexts were not found within the web access history and/or the confidence counters were in unpredictable states.

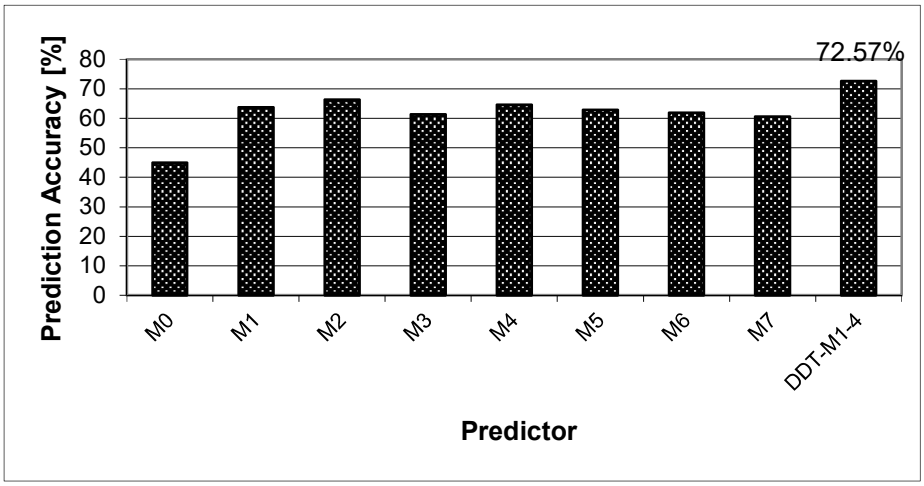

Figure 5. Average prediction accuracies obtained with Markov predictors of different orders and the optimal DDT-M1-4 predictor

The optimal DDT-M1-4 hybrid method outperforms all the evaluated Markov predictors, being thus an efficient two-level predictor which provides better accuracy than all its components. The high average prediction accuracy of 72.57% is possible due to the 4-state confidence counters which classifies dynamically the web pages as predictable or unpredictable and allows to predict and prefetch selectively only from high confidence contexts. Figure 6 shows comparatively the prediction rates and coverages of the same methods.

As Figure 6 depicts, due to the selective approach, the prediction rates are low. Only 6.7% of the web pages are predicted and, as the coverage shows, 5.18% of the total number of web accesses are correctly anticipated with the DDT-M1-4 predictor. By focusing on these 6.7% web page contexts, which were dynamically classified as predictable, the proposed hybrid technique can provide the prediction accuracy of 72.57%, outperforming all its components (see Figure 5). It is also outperforming the PPM algorithm presented in [8], whose average prediction accuracy is 71.11% (with a prediction rate of 10%) on the same dataset. By making fewer predictions but with good accuracy, we can avoid a high number of additional useless prefetches which helps to keep the traffic at reasonable level. The

relative prefetching benefit of the optimal DDT-based method compared to the previous PPM-based technique, computed using equation (3), is 1.52.

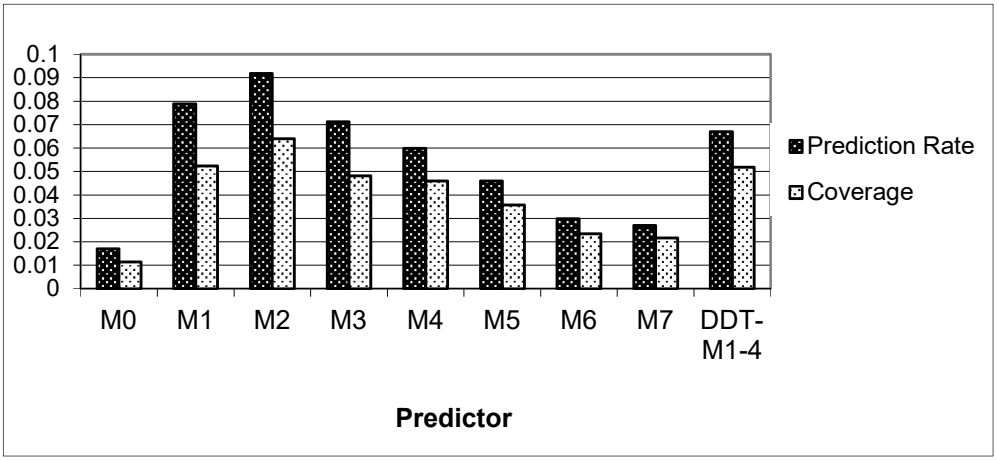

Figure 6. Comparing the average prediction rates and coverages on the BU dataset

Next we varied the history size. Figure 7 presents the prediction accuracies obtained on the BU benchmarks using the optimal DDT-M1-4 predictor with 4-state confidence mechanism. We can observe that the optimal history size is 1000. For a history of 1500 the prediction accuracy is just slightly better.

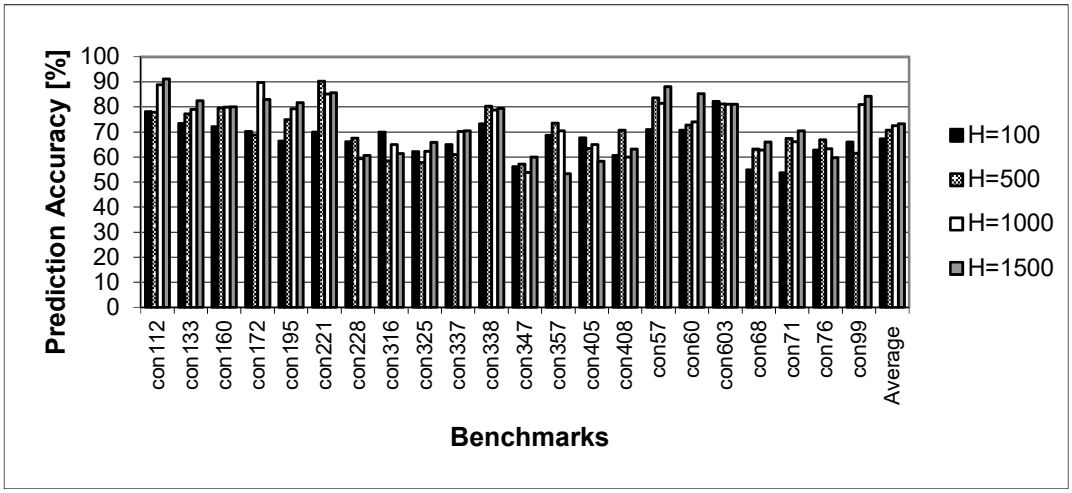

Figure 7. Prediction accuracies obtained with the DDT-M1-4 predictor using different link history lengths

Finally, we have extended the optimal DDT-M1-4 hybrid predictor with a Markov predictor component of order 0. A Markov chain of order 0 is not using any context information, it predicts the

most frequent link from the considered history. Figure 8 presents comparatively the prediction accuracies obtained using DDT-M1-4 and DDT-M0-4, respectively. We have used the optimal history length of 1000 links.

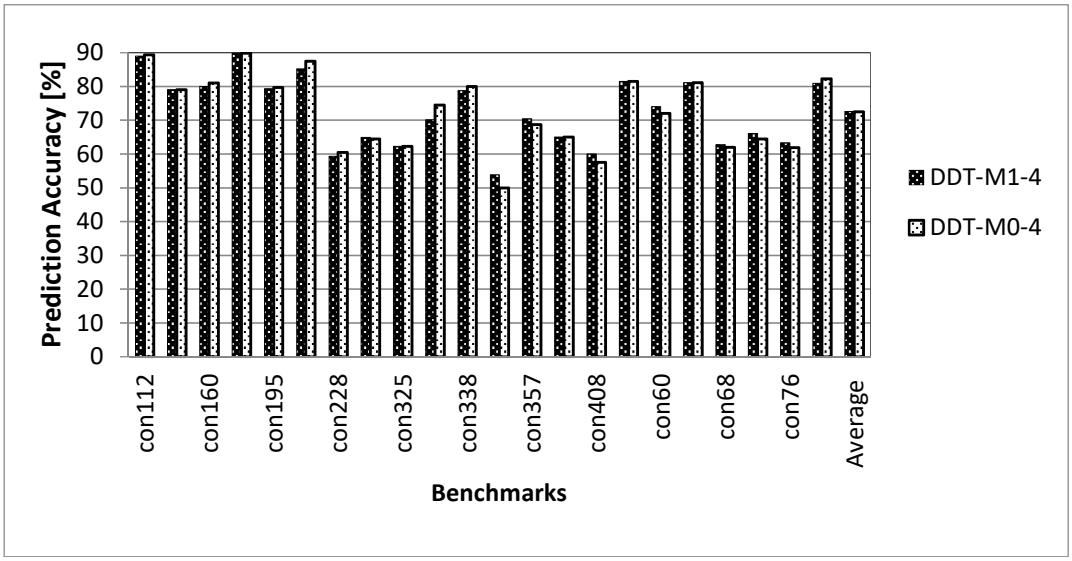

Figure 8. Extending the DDT-M1-4 hybrid predictor with a Markov predictor of order 0

As Figure 8 depicts, on some benchmarks is better to include the Markov predictor of order 0 and on others it is better without, such that the average prediction accuracies are almost the same: 72.57% without and 72.48% with 0 order Markov predictor, respectively. The Markov chain of order 0 cannot improve the average prediction accuracy due to its lack of context information. In this situation we prefer the DDT-M1-4 due to its simplicity with respect to the more complex hybridisation which additionally includes the 0 order Markov chain without any gain in the prediction accuracy. Thus, the optimal method remains DDT-M1-4 with an average prediction accuracy of 72.57%, but reaching almost 90% on some benchmarks (*con112*, *con172*).

5 Conclusions and Further Work

In this paper we presented a two-level hybrid web page prediction method consisting in a DDT with Markovian features. In the first level, the link history is translated into Markovian features represented by the predictions using Markov chains of different orders. These Markovian features are then used together with the static attributes, consisting in the current link and the link type, as input for the DDT algorithm to generate the final second level prediction. We have also used a confidence mechanism consisting in 4-state saturating counters, attached to all the links kept in the history, which allows selective prediction from high confidence contexts, improving thus the accuracy. The predicted web page or file is prefetched into the cache in order to be available for possible next accesses. The

experiments performed on the BU dataset show that the optimal method is the DDT which uses as features the current link, the link type and the predictions provided by the Markov chains of orders 1-4. This optimal predictor uses a history of 1000 web accesses and achieves an average accuracy of 72.57%, outperforming all the Markov chains applied separately. It is also outperforming the PPM algorithm presented in [8] whose average prediction accuracy is 71.11%, measured on the same dataset. The DDT-based method has a relative prefetching benefit over the PPM-based algorithm of 1.52, which is remarkable. Furthermore, it is significantly outperforming the methods presented in [5], [10], [11] and [18], whose prediction accuracies have been mentioned in Section 2. As further work, we intend to investigate other DDT features which can be relevant for even better prediction accuracy. Another further work direction consists in developing and evaluating SVM-based web page predictors.

Acknowledgements

We express our gratitude to the MSc student Ionut-Madalin Ghimis for providing his useful and competent help in implementing the ID3 algorithm.

References

1. Brala, M., Dhanda, M. An Improved Markov Model Approach to Predict Web Page Caching. International Journal of Computer Science and Communication Networks, Vol. 2, 393-399, 2012.
2. Canali, C., Colajanni, M., Lancellotti, R. Adaptive algorithms for efficient content management in social network services. 10th International Conference on Computer and Information Technology, 68-75, 2010.
3. Ciobanu, D., Dinuca, C.E. Predicting the next page that will be visited by a web surfer using Page Rank algorithm. International Journal of Computers and Communications, Issue 1, Vol. 6, 60-67, 2012.
4. Cunha, C. A., Bestavros, A., Crovella, M. E. Characteristics of WWW Client Traces. Boston University Department of Computer Science, Technical Report TR-95-010, 1995.
5. Dubey, S., Mishra, N. Web Page Prediction using Hybrid Model. International Journal on Computer Science & Engineering, Vol. 3 Issue 5, 2170-2176, 2011.
6. Gellert, A, Florea A. Investigating a New Design Pattern for Efficient Implementation of Prediction Algorithms. Journal of Digital Information Management, Vol. 11, Issue 5, 366-377, 2013.
7. Gellert, A., Florea, A. Web Page Prediction Enhanced with Confidence Mechanism, Journal of Web Engineering, Vol. 13, Issue 5-6, 507-524, 2014.
8. Gellert, A., Florea, A. Web Prefetching through Efficient Prediction by Partial Matching, World Wide Web: Internet and Web Information Systems, Vol. 19, Issue 5, 921-932, September 2016.
9. Kaushal, P. Hybrid Markov model for better prediction of web page. International Journal of Scientific and Research Publications, Vol. 2, Issue 8, 2012.
10. Khalil, F., Li, J., Wang, H. Integrating Recommendation Models for Improved Web Page Prediction Accuracy. Proceedings of the 31st Australasian Conference on Computer Science, Vol. 74, 91-100, 2008.
11. Khalil, F., Li, J., Wang, H. An Integrated Model for Next Page Access Prediction. International Journal of Knowledge and Web Intelligence, Vol. 1, No. 1/2, 48-80, 2009.
12. Khanchana, R., Punithavalli, M. Web Page Prediction for Web Personalization: A Review. Global Journal of Computer Science and Technology, Vol. 11, Issue 7, 39-44, 2011.

13. Kundu A., Guha S., Mitra A., Mukherjee T. A New Approach in Dynamic Prediction for User based Web Page Crawling. Proceedings of the International Conference on Management of Emergent Digital Ecosystems, 166-173, Bangkok, Thailand, October 2010.
14. Lowd, D., Davis, J. Improving Markov Network Structure Learning Using Decision Trees. Journal of Machine Learning Research, Vol. 15, 501-532, 2014.
15. Mitchell, T. Machine Learning. McGraw-Hill, 1997.
16. Narvekar, M., Banu, S.S. Predicting User's Web Navigation Behavior Using Hybrid Approach. International Conference on Advanced Computing Technologies and Applications, Vol. 45, 3-12, Mumbai, India, March 2015.
17. Pabarskaite Z. Decision Trees for Web Log Mining. Journal of Intelligent Data Analysis, Vol. 7, Issue 2, 141-154, April 2003.
18. Pamutha, T., Chimphlee, S., Kimpan, C., Sanguansat, P. Web Page Access Prediction on Server Side. Journal of Convergence Information Technology, Vol. 9, No. 5, September 2014.
19. Quinlan, J.R. Induction of Decision Trees. Machine Learning, Vol. 1, Issue 1, 81-106, 1986.
20. Radinsky K., Svore, K.M., Dumais, S.T., Shokouhi, M., Teevan, J., Bocharov, A., Horvitz, E. Behavioral Dynamics on the Web: Learning, Modeling, and Prediction. ACM Transactions on Information Systems, Vol. 31, Issue 3, July 2013.
21. Singhai, N., Nigam R.K. A Novel Technique to Predict Oftenly Used Web Pages from Usage Patterns. International Journal of Emerging Trends & Technology in Computer Science, Vol. 1, Issue 4, 49-55, 2012.
22. Temgire, S., Gupta. P. Review on Web Prefetching Techniques. International Journal of Technology Enhancements and Emerging Engineering Research, Vol. 1, Issue 4, 100-105, 2013.
23. Umapathi, C., Aramuthan M., Raja, K., Enhancing Web Services Using Predictive Caching, International Journal of Research and Reviews in Information Sciences, Vol. 1, No. 3, September 2011.
24. Wan, M., Jönsson, A., Wang, C., Li, L., Yang, Y. Web user clustering and Web prefetching using Random Indexing with weight functions. Knowledge and Information Systems, Vol. 33, Issue 1, 89-115, 2012