Appeared in *Journal of Web Engineering*, Vol. 20, Issue 5, pp. 1279-1296, 2021

Web Usage Mining by Neural Hybrid Prediction with Markov Chain Components

Arpad Gellert

Computer Science and Electrical Engineering Department, Lucian Blaga University of Sibiu, Romania, arpad.gellert@ulbsibiu.ro

Abstract.

This paper presents and evaluates a two-level web usage prediction technique, consisting of a neural network in the first level and contextual component predictors in the second level. We used Markov chains of different orders as contextual predictors to anticipate the next web access based on specific web access history. The role of the neural network is to decide, based on previous behaviour, whose predictor's output to use. The predicted web resources are then prefetched into the cache of the browser. In this way, we considerably increase the hit rate of the web browser, which shortens the load times. We have determined the optimal configuration of the proposed hybrid predictor on a real dataset and compared it with other existing web prefetching techniques in terms of prediction accuracy. The best configuration of the proposed neural hybrid method provides an average web access prediction accuracy of 86.95%.

Keywords. Web prefetching, webpage prediction, Markov chains, neural networks, browser extension.

1. INTRODUCTION

Prefetching is a technique which allows a browser to fetch into its cache the web resources which have a high probability of being accessed next. If the web resources are already in the cache at the time of the access, the browser can load them faster, thus reducing the latency. When the prefetching is applied to all accesses, it can increase network traffic, as it can fetch unnecessary resources, too. In order to reduce the ratio of unnecessary fetches, the web resources are classified dynamically into unpredictable and predictable, the prefetching being performed only in predictable states. Thus, the next resource is predicted only if the current resource is in a predictable state. This selective approach significantly improves the prediction accuracy. In [10], we have already compared web prefetching applied through graph algorithms, Markov chains, and Hidden Markov Models (HMM), respectively. We have also investigated web prefetching through prediction by partial matching (PPM) in [11] and through dynamic decision trees (DDT) in [12].

In this work, a neural hybrid web prefetching method is presented and evaluated. The proposed two-level predictor uses a neural network in its first level and contextual predictors in its second level. Our contextual predictors are Markov chains of different orders. The role of the neural network from the first level is to select the predictor from the second level. The web resources anticipated by the selected predictor are prefetched if the current web resource is in a predictable state. Since we observed that the correct prediction is not always provided by the same predictor (it can differ among users, or it can change in time even for a certain user), a hybrid predictor could exploit the advantage of all its components.

Artificial neural networks are connectionist models composed of neurons organized on layers and they can learn from examples to associate outputs with the given inputs. Each connection between two neurons has a weight, which indicates how strong the connection is. The inputs applied to the neural network traverse the layers and are transformed according to their weight. If the output is wrong, the error can be backpropagated through the layers to adjust the weights correspondingly. Since the computation of the output for the given inputs does not require a specific algorithm, neural networks have been widely used to solve high-complexity practical problems. Its role in our application is to decide which predictor is better in a specific web browsing context. Thus, it will select a Markov chain of a specific order. The Markov predictor uses a limited web access history and provides a context-based prediction of the subsequent web access. The Markov prediction is efficiently generated, by performing simple searches for the context within the web access history, as in [11], instead of using high complexity graph- or transition table-based modelling. The neural network has good adaptability and can dynamically select the best predictor. We expect this novel neural hybrid web prefetcher to outperform the previously developed techniques in terms of prediction accuracy and rendering speedup. The comparative evaluations were performed on the Boston University (BU) dataset, collected by the "Ocean Group Research" [4].

The remainder of this paper is structured as follows. Section 2 reviews the related work in the web prefetching domain. Section 3 describes the proposed neural hybrid web prefetcher. Section 4 presents and discusses the experimental results. Finally, Section 5 concludes the paper and suggests further work directions.

2. RELATED WORK

Markov chains were used for link prediction in [27]. In [17], Kaushal used a Markov chain for webpage prediction, together with the k-nearest neighbour classifier algorithm. In [22], Pamutha et al. combined all-kth order Markov chains with fuzzy adaptive resonance theory for webpage prediction, but they obtained an accuracy under 40%. In [6], Dubey and Mishra proposed a hybrid webpage predictor which combines support vector machines, association rule mining, and Markov chains, obtaining an accuracy of around 63%. In [2], Brala and Dhanda combined first-order Markov chains, association rule mining, and clustering for webpage prediction. In [21], Narvekar and Banu combined Markov chains and HMMs. In [1], Adelaja and Akinola proposed a web page recommendation system based on HMMs combined with page ranking.

In [11], web prefetching was performed through prediction by partial matching (PPM), a hybrid predictor with static prioritisation of its components. It starts trying to predict with a Markov predictor of a specific order, iteratively decreasing the order in case of failure. The experimental results obtained on the BU dataset have shown that the fourth-order PPM was the best configuration, with a prediction accuracy of 71.11%. In contrast, the neural hybrid predictor proposed in this work, applies dynamic prioritisation adapted to the user behaviour changes.

In [12], a decision tree was used to select dynamically the most appropriate features employed to predict webpages. The feature set consisted of the current link and its type and also the predictions provided by Markov chains of different orders. The decision tree was built top-down recursively, by choosing the feature that best classifies the set of examples at each node. The decision tree was then used to anticipate the subsequent web resource based on the current context. The average prediction accuracy was 72.57%. Decision trees were used for web usage prediction also in [23].

Neural networks have also been used for web prefetching. In [8], Feng et al. applied the adaptive resonance theory (ART) neural network, widely used for classification or clustering. The main idea of the cluster-based approach is that if a webpage is requested from a specific cluster, the other webpages from that cluster are likely to be accessed in the near future. The authors proposed the so-called pART1 algorithm, modifying the ART1 neural network (a type of ART which works with binary inputs) to obtain two-way weights between host clusters and webpages, and used these weights for prefetching decisions. In [14] and [31], the authors proposed a neural network relying on the keywords from the anchor texts of the current uniform resource locator (URL) to predict the next URLs, thus providing semantic prefetching capabilities. In [20], Mandal and Azad used artificial neural networks for webpage prediction, in combination with the K-means clustering algorithm. The neural network uses the clusters formed using the K-means algorithm as inputs. In [29], Tian et al. proposed a neural network which anticipates the next web resource based on the following data extracted from the current web resource: document type, number of previous hits, relative access frequency, document size, and retrieval time. Their evaluations, performed on the same BU dataset as ours, have shown a prediction accuracy of up to 82%. The main difference between all these methods relying on neural networks and our method is that we do not use the neural network directly for web prediction. In our approach, the prediction is provided by a specific contextual predictor selected by the neural network.

In [24], Setia et al. presented a threshold-based web prefetching control mechanism which relies on a neural network to decide when prefetching can be triggered and to establish the number of pages to be prefetched so that the network bandwidth is used efficiently. In contrast, the neural network presented in this paper decides which component predictor to use for web prediction. The method investigated by Setia et al. was validated on the NS-2 simulator, showing an optimized trade-off between latency and bandwidth usage.

Deep learning was applied to web prefetching in [32]. The authors combined a long short-term memory (LSTM) model with the Skip-Gram embedding method and a two-layer neural network to predict the subsequent accessed links. All the links are classified into prefetching and skipping categories, according to their size and latency, prefetching thus only the web objects larger than 50 KB or whose latency is longer than 1.5 seconds. The proposed complex prediction model uses three input information: the link, the webpage title and the IP together with a timestamp. The link is encoded by a Skip-Gram layer and the obtained output vector is applied to the LSTM. The webpage title and the IP with the timestamp are encoded by other embedding layers and applied together with the LSTM's output to the two-layer neural network, which then provides the final prediction. The evaluations performed on the same BU dataset as ours have shown a prediction accuracy of up to 80%. In [18], Kim proposed a web prefetching framework which applies weblog pre-processing – through a Hadoop-based MapReduce programming model – followed by webpage prediction using a deep learning algorithm.

In [25], Shyamala and Kalaivani proposed a prefetching mechanism relying on the Monte Carlo prediction algorithm. A user navigation graph is built in the first phase of the algorithm. The prediction is performed in the second phase based on a Monte Carlo search applied on a user navigation graph for frequently accessed web pages, which are then sent to the browser as prefetch hints. In [26], Shyamala and Kalaivani applied the enhanced Monte Carlo prediction algorithm, which includes a rank for each predicted page through a feedback process, thus improving the accuracy. In [13], Hong et al. presented another method which predicts the next web pages based on access pattern graphs and then prefetches them into a non-volatile random-access memory that stores resources for browser caching. In [5], the authors presented an approach of webpage access prediction using the CPT+ and PageRank algorithms, which reduced the space complexity by 15%. In [15], Joo et al. presented a predictive prefetching scheme using interaction events to determine the client's context and intention. In [28], Sisodia et al. applied the most interesting pattern-based parallel FP-growth algorithm to extract frequent patterns from long weblogs and to predict the browsing behaviour of the users. A comprehensive review of probabilistic techniques used for caching in web browsers is presented in [16].

3. NEURAL HYBRID WEB USAGE PREDICTION

The proposed browser extension collects the links accessed by the user, eliminating the ports and relative parts and the links to small images like GIF and XBM. In the case of multiple consecutive occurrences of a link, only one is kept. The collected links are codified with unique numbers. The browser extension stores such a codified link history of size H, updated after each web access, and uses it to anticipate the next link.

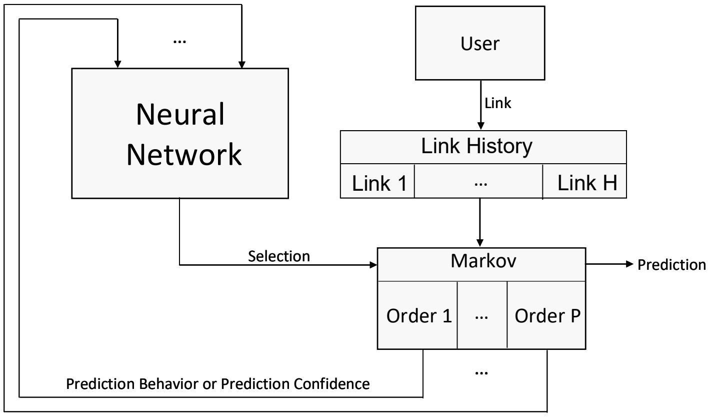

Figure 1. The structure of the browser extension

Based on the last behaviours of the component Markov chains of different orders, the neural hybrid prediction selects the one to be used to generate the prediction. The behaviours of a specific Markov model can be provided as codified confidence counters or as a limited history of correct / wrong predictions. The final prediction is then provided by the selected Markov model. The predicted web resources are prefetched into the cache of the browser to be available if the user accesses them again in the future. The structure of the browser extension is presented in Figure 1.

For higher prediction accuracy and lower traffic level (which are the main characteristics of efficient prefetchers [3]), each component Markov predictor relies on a confidence mechanism as in [10], implemented using 4-state saturating counters for all the web links kept in the web usage history. The saturating counter of the current web link is incremented on correct predictions and decremented in case of mispredictions (see Figure 2). The predicted link is used for prefetching only if the confidence counter of the current web link is in a predictable state.

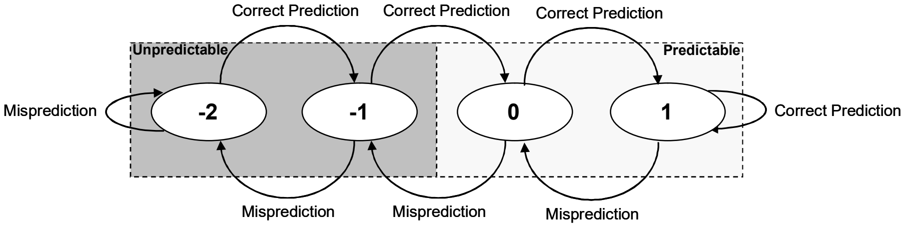

Figure 2. The 4-state confidence counter

To further decrease the traffic level, we prefer to predict and prefetch only one web object at a time. Further on, we will describe the components of the proposed hybrid weblink predictor. A Markov chain can be described as follows [10]:

$$P[q_t \mid q_{t-1}, \ ..., \ q_{t-H}] = P[q_t \mid q_{t-1}, \ ..., \ q_{t-R}] \tag{1}$$

where q_t in our application is the web link at time t and P is its probability, whereas H is the size of the considered limited link history and R is the order of the Markov chain. The last two are the input parameters of the model. As we can observe in (1), in a Markov chain of order R, the current web link depends only on the R previous weblinks, also called the context. Thus, Markov chains can be used as context-based predictors. The Markov model implementation determines the prediction by applying simple searches for the current context in the limited web link history, as in [11]. The model predicts the next web link as being the one which followed the current context the most frequently. Thus, the probabilities of the web links are modelled through their occurrence frequencies in a specific web usage context. A generally applicable implementation of this Markov predictor in the Java programming language is presented in [9].

Next, we will describe the neural network which has the role of the selector in the proposed hybrid predictor. We used a multi-layer perceptron with one hidden layer. The structure is presented in Figure 3.

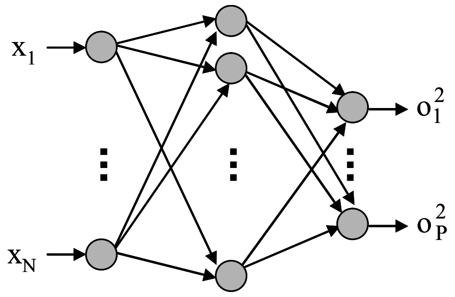

Figure 3. The structure of the neural network

For better results we used the hyperbolic tangent activation function:

$$F(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{2}$$

Thus, the input information must be codified using the values -1 and 1 and the output of the neural network is obtained in the [-1, 1] interval.

Regarding the input $\overline{X}$ of the neural network, we developed three different approaches. In the first approach, we used as input information the current link's 4-state confidences associated with the component Markov predictors (depicted in Figure 2), each codified on 1 bit. The input vector is built-up as follows: the input value corresponding to a certain component predictor is -1 if the associated confidence automaton is in unpredictable state (-2 or -1 in Figure 2), or 1 in the case of predictable state (0 or 1 in Figure 2). In the second approach, we codified the current link's 4-state confidence associated with each Markov predictor on 2 bits, as follows: {-1, -1} if the automation is in strong unpredictable state (-2 in Figure 2), {-1, 1} if the automaton is in weak unpredictable state (-1 in Figure 2), {1, -1} if the automaton is in weak predictable state (0 in Figure 2), or {1, 1} if the automaton is in strong predictable state (1 in Figure 2). In our third approach, we applied the behavioural history of each component Markov predictor as input information to the neural network. Such a behavioural history has a fixed length (which in this third approach is a parameter of the neural predictor) and keeps the last behaviours of a specific component predictor. The per component behavioural history codifies the correct predictions with 1 and the mispredictions with -1.

The output layer of the neural network has one neuron associated with each component Markov predictor. Thus, the number of output layer neurons is identical to the number of component predictors. The Markov predictor selected to provide the prediction is the one whose associated neural output has the highest value. For better results, we enable the prediction only if this highest neural output value exceeds a certain threshold. This prediction threshold, together with the already presented confidence mechanism, can help to apply the prediction selectively and with higher accuracy.

For a neural network with N input layer neurons, M hidden layer neurons and P output layer neurons, the weights w_{ij}^1 and w_{jk}^2, $i = \overline{1, N}$, $j = \overline{1, M}$, $k = \overline{1, P}$, are initialized with small random values belonging to the [-0.1, 0.1] interval, as it is recommended in [19]. The component predictor selection, relying on the Backpropagation algorithm [7], is applied as follows:

- Apply the input $\overline{X}$ to the neural network and compute its output $\overline{O^2}$:

$$\overline{O^1} = \overline{X} \cdot \overline{W^1} \tag{3}$$

$$\overline{O^2} = \overline{O^1} \cdot \overline{W^2} \tag{4}$$

- Based on the neural network's output $\overline{O^2}$, determine the order R of the Markov chain:

$$R = k,\, o = \max_{1 \le k \le P}\left(o_k^2\right) \tag{5}$$

- For each output layer unit k , $k = \overline{1, P}$, compute the error term δ_k^2 .

$$\delta_k^2 = \frac{1}{2} \cdot \left(1 - o_k^2 \cdot o_k^2\right) \cdot \left(t_k - o_k^2\right) \tag{6}$$

- Update the weights w_{jk}^2, $j = \overline{1, M}$, $k = \overline{1, P}$.

$$w_{jk}^2 = w_{jk}^2 + \alpha \cdot \delta_k^2 \cdot o_j^1 \tag{7}$$

- For each hidden layer unit j , $j = \overline{1, M}$, compute the error term δ_j^1 .

$$\delta_j^1 = \frac{1}{2} \cdot \left(1 - o_j^1 \cdot o_j^1\right) \cdot \sum_{k=1}^{P} w_{jk}^2 \cdot \delta_k^2 \tag{8}$$

- Update the weights w_{ij}^1, $i = \overline{1, N}$, $j = \overline{1, M}$.

$$w_{ij}^1 = w_{ij}^1 + \alpha \cdot \delta_j^1 \cdot x_i^1 \tag{9}$$

As we can observe in the above-presented algorithm, the neural network is started with randomly generated weights, and it is dynamically trained during the prediction process. Thus, after the forward step (3-4) and the Markov chain selection (5), the neural network is updated by adjusting its weights in the backward step (6-9), where α is the learning rate.

4. Experimental Results

For our evaluations, we have used the BU dataset generated by the "Ocean Group Research" from Boston University [4], composed of log files. The name of such a log file contains information like user ID, computer, and start-timestamp of the session. Each line of the log files is associated with an URL accessed by the user and contains the computer name, timestamp of the access, URL, size of the object and its retrieval time. We used prediction accuracy as an evaluation metric, which is computed by dividing the number of correctly predicted links by the number of predicted links. For all the prediction techniques evaluated in this work, we used a confidence mechanism as in [10]. As components for the neural hybrid predictors, we considered Markov chains of orders 1-4. This decision was made taking into account the evaluations of the DDT hybrid predictor [12], concluding that the inclusion of Markov predictors of order above 4 does not increase the prediction accuracy. In other words, web usage patterns are usually not longer than 4 links. Based on a rule determined in [30], we have preconfigured the neural network with $M = N + 1$ hidden layer neurons, where N is the number of input layer neurons. Additionally, we have set the link history to 1300. Smaller values caused prediction incapacity on specific benchmarks for some Markov chain components, as too long patterns could not be found within a too short web usage history. We have also set the neural output threshold to 0.5. With higher values, the neural network was again unable to predict on some benchmarks, whereas with lower values, the prediction accuracy was decreased. After this parametric pre-configuration, the neural hybrid web usage predictor could successfully be run on all the evaluated benchmarks. Therefore, we will keep the optimal values for these preconfigured parameters in all the evaluations of the neural hybrid predictor and we will systematically vary the other parameters. We are interested in finding the optimal configuration in terms of prediction accuracy.

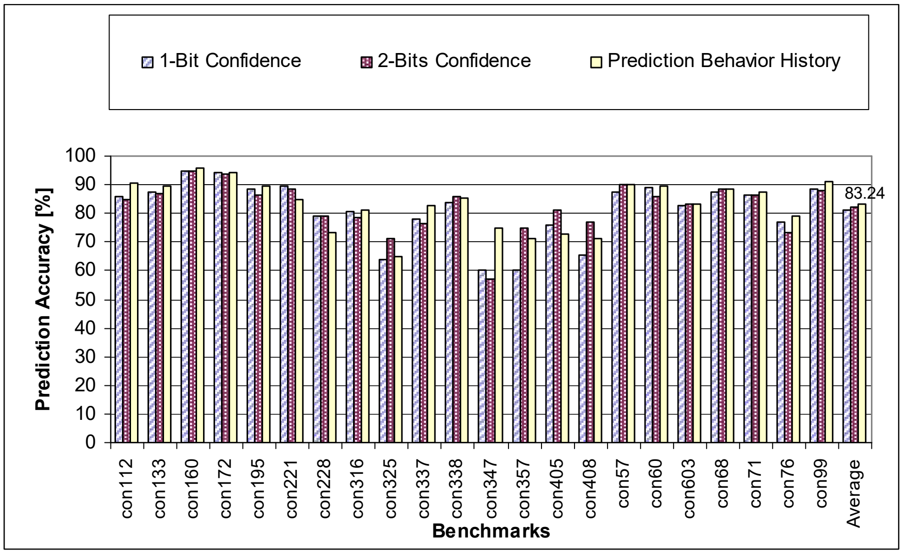

Figure 4. Analyzing neural input types

First, we studied the different input types which can be used by the neural network: the 4-state confidences codified on 1-bit per component, the 4-state confidences codified on 2-bits per component, or the behavioural history of the component Markov predictors (correct prediction or misprediction). For this initial run, we used a learning rate of 0.3 and an input length of 1. As Figure 4 shows, the behavioural history input was better on average than both confidence-based inputs. Thus, we will continue the evaluations of the neural network with behavioural history input.

The next evaluated parameter is the learning rate (L) which is the amount of error used to update the weights. We varied its value between 0.1 and 0.5. We can observe in Figure 5 a peak in the prediction accuracy for a learning rate of 0.2 and a constant degradation after this value. The poor results obtained for a larger learning rate can be the effect of a sub-optimal set of weights and of a real risk of falling into local minima [7]. On the other hand, the poor prediction accuracy obtained with a learning rate of 0.1 can be caused by the limited training set. Therefore, we will continue the evaluations by setting the learning rate to 0.1.

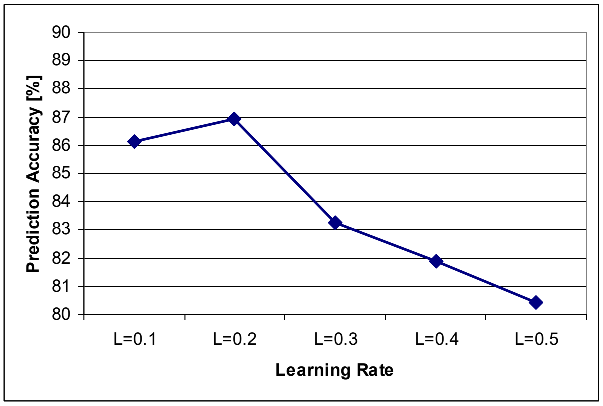

Figure 5. Varying the learning rate

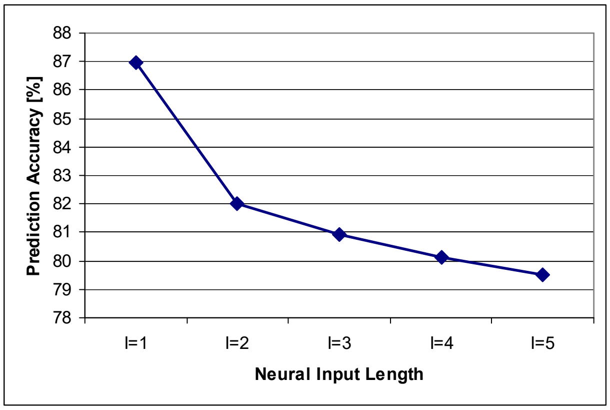

Figure 6. Varying the neural input length

Next, we will vary the neural input length (I). Since we have decided to use the behavioural history of the components as input information, this parameter decides how many last behaviours (correct/wrong predictions) of each component Markov chain are provided to the neural network. As Figure 6 shows, a behavioural input length of 1 is by far the best choice. The prediction accuracy is deteriorating as the input length is increasing. The explanation for lower prediction accuracy with longer input lengths can be a more complex and, thus, inefficient learning process. Consequently, we set the input length to 1, the neural network using the last behaviour of each component Markov chain.

After this evaluation stage, we can conclude that the optimal configuration of the hybrid neural predictor uses behavioural history of length 1 as input, a learning rate of 0.2, and we can also mention the preconfigured link history of 1300 and the neural output threshold of 0.5. Next, we will compare this optimal configuration of the neural hybrid predictor with all its component predictors, thus Markov chains of orders 1-4 (see Figure 7). As we expected, the proposed hybrid predictor outperforms all its components. This improvement is possible due to the applied two-level hybrid approach, which dynamically selects the best behaving component.

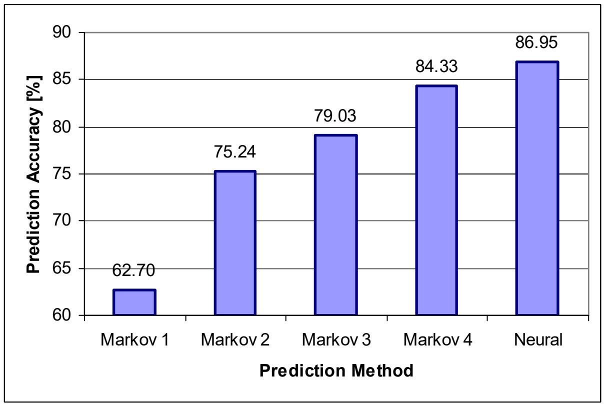

Figure 7. Comparing the neural hybrid predictor with its Markov chain components

Finally, we compared our best neural hybrid predictor with the optimally configured PPM [11] and DDT [12] hybrid web access predictors, on the same BU dataset. Figure 8 depicts the prediction accuracy of these three hybrid prediction methods. As Figure 8 shows, the proposed neural hybrid predictor significantly outperforms both the PPM-based and DDT-based hybrid web usage predictors. There is only one benchmark from the whole dataset, namely con405, whose prediction accuracy was slightly better with the PPM predictor. On all the other benchmarks, the neural predictor was the best, on two of them reaching a prediction accuracy of 100%. Its average prediction accuracy was 86.95%, in contrast with the 71.11% of the PPM predictor and the 72.57% of the DDT predictor, which is remarkable, taking into account that all these hybrid prediction methods are using Markov chains as components. The inferiority of the PPM predictor can be explained by the static prioritisation of its Markov chain components: it starts with the highest order and iteratively decreases the prediction order in case of unpredictability. Since both the neural and the DDT hybrid predictors are applying dynamic component prioritisation, the inferiority of the DDT predictor can be explained only through its lower adaptability to the changes in the users' behaviour.

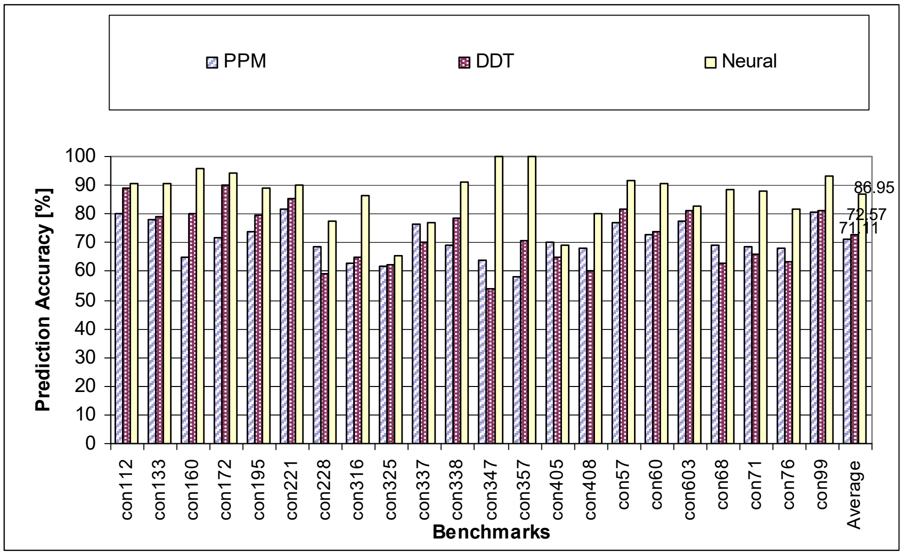

Figure 8. Comparing different hybrid web usage predictors

5. Conclusions and Further Work

In this paper, we proposed a two-level hybrid web usage prediction scheme, consisting of a neural network in its first level, and Markov predictors of different orders in its second level. The role of the neural network is to dynamically select the Markov chain, which is then used to predict and prefetch the next web page. For better results, we have attached a per link 4-state confidence automaton to each component Markov predictor, which can dynamically classify links as predictable or unpredictable. Thus, we can selectively apply the prediction process in predictable states. We have investigated three different neural input types. Two of them consist in the current link's confidence automata associated with the component predictors, each automaton state being codified on 1 bit or 2 bits, respectively. The other one is the recent behavioural history (correct or wrong prediction) of each component predictor.

The evaluations have shown on the BU dataset, that the behavioural history is the best input type. We have determined the optimal configuration of the neural network, considering the behavioural history of component predictors as input. The best prediction accuracy, 86.95% on average, was obtained using a link history of 1300 for the component Markov predictors of orders 1-4, a neural output threshold of 0.5, a behavioural history of length 1 as neural input, and a neural learning rate of 0.2. The experiments have also shown that the optimal neural hybrid web usage predictor outperforms the Markov predictors, the PPM [11], and the DDT [12]. It also outperforms the neural method evaluated in [29] and the deep-learning model presented in [32].

As further work, we intend to develop and evaluate hybrid web page prediction based on support vector machines. Finding similarities among users and exploiting them in the prediction schemes is another further work direction.

6. ACKNOWLEDGEMENTS

This work was partially supported by Lucian Blaga University of Sibiu & Hasso Plattner Foundation research grants LBUS-IRG-2020-06. Special thanks to Cosmin-Gabriel Mihalcea for providing his help in implementing the predictor.

7. REFERENCES

[1] T.T. Adelaja, S.O. Akinola, 'An Enhanced Web Page Recommendation System Using Hidden Markov Model and Page Rank Technique', Journal of Science and Logics in ICT Research, Vol. 1, 57-63, June 2017.

[2] M. Brala, M. Dhanda, 'An Improved Markov Model Approach to Predict Web Page Caching', International Journal of Computer Science and Communication Networks, Vol. 2, 393-399, 2012.

[3] L. Chun-jie, L. Hong-bo, L. Hua, Z. Li-ming, F. Chun-yan, G. Ji-ping, 'Research on Polynomial Regression Prefetching Model', IOP Conference Series: Earth and Environmental Science, Volume 332, Issue 2, 2019.

[4] C.A. Cunha, A. Bestavros, M.E. Crovella, 'Characteristics of WWW Client Traces', Boston University Department of Computer Science, Technical Report TR-95-010, 1995.

[5] N.T. Da, T. Hanh, P.H. Duy, 'Improving Webpage Access Predictions Based on Sequence Prediction and PageRank Algorithm', Interdisciplinary Journal of Information, Knowledge, and Management, Vol. 14, 27-44, 2019.

[6] S. Dubey, N. Mishra, 'Web Page Prediction using Hybrid Model', International Journal on Computer Science & Engineering, Vol. 3 Issue 5, 2170-2176, 2011.

[7] W. Ertel, 'Introduction to Artificial Intelligence', Second Edition, Springer, 2017.

[8] W. Feng, T.H. Kazi, G. Hu, 'Web Prefetching by ART1 Neural Network', Software and Network Engineering. Studies in Computational Intelligence, Vol. 413, 29-40, 2012.

[9] A. Gellert, A. Florea, 'Investigating a New Design Pattern for Efficient Implementation of Prediction Algorithms', Journal of Digital Information Management, Vol. 11, Issue 5, 366-377, 2013.

[10] A. Gellert, A. Florea, 'Web Page Prediction Enhanced with Confidence Mechanism', Journal of Web Engineering, Vol. 13, Issue 5-6, 507-524, 2014.

[11] A. Gellert, A. Florea, 'Web Prefetching through Efficient Prediction by Partial Matching', World Wide Web: Internet and Web Information Systems, Vol. 19, Issue 5, 921-932, September 2016.

[12] A. Gellert, 'Web Access Mining through Dynamic Decision Trees with Markovian Features', Journal of Web Engineering, Vol. 16, Issue 5-6, 524-536, 2017.

[13] S. Hong, K. Kim, T. Kim, 'A Prefetching Scheme for Improving the Web Page Loading Time with NVRAM', Journal of Semiconductor Technology and Science, Vol. 18, No. 1, 20-28, February 2018.

[14] T. Ibrahim, C.-Z. Xu, 'Neural Nets Based Predictive Pre-fetching to Tolerate WWW Latency', 20th IEEE International Conference on Distributed Computing Systems, 636-643, Taipe, Taiwan, April 2000.

[15] M. Joo, Y. An, H. Roh, W. Lee, ‘Predictive Prefetching Based on User Interaction for Web Applications’, IEEE Communication Letters, Vol. 25, Issue 3, 821-824, March 2021.

[16] A.S. Kakar, M.S. Rohie, ‘A Review of Probabilistic Techniques Used for Web Browsers’ Caching’, European Journal of Engineering research and Science, Vol. 5, Issue 7, 773-780, July 2020.

[17] P. Kaushal, ‘Hybrid Markov model for better prediction of web page’, International Journal of Scientific and Research Publications, Vol. 2, Issue 8, 2012.

[18] K.-C. Kim, ‘Framework for Efficient Web Page Prediction using Deep Learning’, Journal of the Korea Society of Computer and Information, Vol. 25, Issue 12, 165-172, 2020.

[19] M. Kubat, ‘An Introduction to Machine Learning’, Second Edition, Springer, 2017.

[20] O.P. Mandal, H.K. Azad, ‘Web Access Prediction Model using Clustering and Artificial Neural Network’, International Journal of Engineering Research & Technology, Vol. 3, Issue 9, 195-199, 2014.

[21] M. Narvekar, S.S. Banu, ‘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.

[22] T. Pamutha, S. Chimphlee, C. Kimpan, P. Sanguansat, ‘Web Page Access Prediction on Server Side’, Journal of Convergence Information Technology, Vol. 9, No. 5, September 2014.

[23] K. Radinsky, K.M. Svore, S.T. Dumais, M. Shokouhi, J. Teevan, A. Bocharov, E. Horvitz, ‘Behavioral Dynamics on the Web: Learning, Modeling, and Prediction’, ACM Transactions on Information Systems, Vol. 31, Issue 3, July 2013.

[24] S. Setia, Jyoti, N. Duhan, ‘Neural Network Based Prefetching Control Mechanism’, International Journal of Engineering and Advanced Technology, Vol. 9, Issue 2, 2019.

[25] K. Shyamala, S. Kalaivani, ‘Application of Monte Carlo Search for Performance Improvement of Web Page Prediction’, International Journal of Engineering & Technology, Vol. 7, No. 3.4, 133-137, 2018.

[26] K. Shyamala, S. Kalaivani, ‘Enhanced Webpage Prediction Using Rank Based Feedback Process’, New Trends in Computational Vision and Bio-inspired Computing, Springer, 567-576, September 2020.

[27] N. Singhai, R.K. Nigam, ‘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.

[28] D.S. Sisodia, V. Khandal, R. Singhal, ‘Fast Prediction of Web User BrowsingBehaviours Using Most Interesting Patterns’, Journal of Information Science, Vol. 44, No. 1, February 2018.

[29] W. Tian, B. Choi, V. Phoha, ‘An Adaptive Web Cache Access Predictor Using Neural Network’, Development in Applied Artificial Intelligence, Lecture Notes in Artificial Intelligence, Vol. 2358, 450-459, 2002.

[30] L. Vintan, A. Gellert, J. Petzold, T. Ungerer, ‘Person Movement Prediction Using Neural Networks’, Proceedings of the KI2004 International Workshop on Modeling and Retrieval of Context (MRC 2004), Vol. 114, Ulm, Germany, September 2004.

[31] C.-Z. Xu, T. Ibrahim, ‘A Keyword-Based Semantic Prefetching Approach in Internet News Services’, IEEE Transactions on Knowledge and Data Engineering, Vol. 16, Issue 5, 601-611, 2004.

[32] W. Zou, J. Won, J. Ahn, K. Kang, ‘Intentionality-related Deep Learning Method in Web Prefetching’, Proceedings of the 27th International Conference on Network Protocols, Chicago, USA, October 2019.

8. BIOGRAPHY

Arpad Gellert obtained his MSc (2003) and the PhD (2008) in Computer Science at Lucian Blaga University of Sibiu. He is currently working as an associate professor in the Computer Science and Electrical Engineering Department of the same university. He also worked as visiting researcher in Barcelona and Milano. Previously he was a Java developer at Multimedia Capital Romania. His research interests include computer architecture, smart buildings & factories, web mining and image processing. He published 5 books and over 50 scientific papers in some prestigious journals and international top conferences and acquired more than 300 citations. He was member of several research grants. He developed as a project manager a research grant supported by the Romanian National Council of Academic Research and three internal LBUS grants. Currently he is member of an ongoing Hasso Plattner Excellence Research Grant. He received in 2010 the “Ad Augusta Per Angusta” prize awarded by Lucian Blaga University of Sibiu for excellence in scientific research. His webpage can be found at http://webspace.ulbsibiu.ro/arpad.gellert.