



## Next Web Page Prediction by Utilization of Content and Web-Log Features with Markov Model

**Pinki Jain**  
M.TECH. (CSE)  
ICOT, Bhopal, India

**Harish Patil**  
M.TECH. (CSE)  
ICOT, Bhopal, India

**Jay Prakash Maurya**  
A.P. (CSE)  
ICOT, Bhopal, India

---

**Abstract**— *In prediction of next page several models are prepared which utilize different techniques. For this new technique different outcomes are obtained which have probability to predict the chance of the next page. So targeting the most probable page makes the very low precision. The basic component here is to collect different features of the web so that prediction is more effective than the single and unorganized one. For the optimization of the prediction it is required that all the features of the website should be utilized. In this paper, the content of web with Markov model is utilized for the prediction of next page. Result shows that by the use of Markov Model with web content and web log feature is better for prediction.*

---

**Index Terms**— *Information Extraction, Text Analysis, Ontology, feature extraction, text categorization, clustering*

---

### I. INTRODUCTION

Recent times the massive use of web has increased the traffic in network and also the load the web servers manages. Even though nowadays web users have made higher bandwidth connections, they still perceive high latencies when navigating the web due to overloaded elements, long message transference times, and the Round Trip Time [4]. As a result, the reduction of the users perceived latency when browsing the web is still a crucial

Research issue. The reduction of the web users' perceived latency has become the subject of many research efforts over the last few years. The widely used techniques proposed to reduce this latency are web caching, geographical replication, and pre-fetching. Caching techniques are widely implemented now days since they achieve important latency savings. Many Multinational companies implement web replication by using Content Delivery Networks [2] to reduce their websites access time, but this solution is not feasible as it is expensive and many small companies, organizations cannot afford it. Web pre-fetching techniques are mutually independent to caching and replication techniques, so that they can be applied together to achieve a better web performance. Caching and replication techniques have been widely implemented in real world, some studies have also investigated web pre-fetching in real environments [10].

Web pre-fetching is used to pre-process object requests before the user makes an explicit demand of those objects, in order to reduce the users' perceived latency. Web pre-fetching consists of two steps mainly; first, it is a necessity to make accurate prediction then next user accesses. These predictions are usually made based on previous experience about users' accesses and preferences, and the relative hints are provided to a pre-fetching engine, second, the pre-fetching engine makes a decision which objects from the predicted hints are going to be pre-fetched.

The elements of web architecture such as the client, the proxy or the server where the prediction engine has been located, this information can be used by the prediction engine. One user access pattern is used for performing the predictions when the location is at client side [4]. However, multi user and multi server patterns are used to gather information if the predictor is at the proxy server [6]. If the location of engine is at the side of server predictions made are based on multi-user accesses to the same website [1, 3]. Finally, the predictions can be performed by several elements in collaboration.

### II. RELATED WORK

There are few research works to make comparison of the performance between different prediction algorithms this is mainly because of the difficulty in reproducing environments and workloads[1]. Two algorithms based on Markov models, proposed by Zukerman [5] and by Bestavros. The comparison is only performed at the algorithmic level, without giving consideration to the details related to the user's perceived latency.

The first appropriate study that analyzed the potential benefits of web pre-fetching was done out by Kroeger et al. [4]. The results of analysis gave the limits on latency savings that caching and pre-fetching reached in several scenarios. Experiments were performed with a perfection of-line prediction algorithm and using traces obtained from a proxy server in the year 1996. The result of that research work was that pre-fetching doubles the latency reduction achieved by caching, which is limited by the frequent update of objects in the web. However, their results are hard to be compared due to the assumptions made about the environment and the workload characteristics.

In [2] discovered, using traces from, the limits on the benefits that a perfect prediction algorithm located at the proxy server could achieve. As in [2] stated that the results obtained in web pre-fetching experiments are highly dependent on the architecture or model assumptions, and that the parameters that influence web pre-fetching are in close relation with the web architecture itself.

In [1] researched on the influence of the web architecture on the limits of latency reduction. The result was obtained concluding that the main constraint to obtain the upper bound for saving latency savings is due to the location in which a user access cannot be predicted, e.g., the first access of a session and the first time the predictor sees an object. Their results proved that there is a dependency between latency reduction and location of the predictor. They stated that if the predictor is located at server side, client or proxy, respectively the latency can be reduced by 36%, 54%, and 67% whereas the reductions in latency can be higher than 90% if the predictor worked collaboratively at different elements of the architecture.

In [3] suggested a calculative model for a web pre-fetching architecture. The result of which showed that pre-fetching produced profit even with the presence of a good caching system.

In the TLPM [9], in level one, Markov model is used to predict the next possible category which will be browsed by the user. In level two, Bayesian theorem is used to predict the next possible page which belongs to the predicted category of level one to archive the goal of reducing prediction scope more efficiently through the two-level framework. The experiment result proves that TLPM can archive the purpose and improve the efficiency of prediction by the way of finding out the important category in level one and decreasing the candidate page set in level two.

### III. BACKGROUND

Whole work is divide into two steps first is the training modules this can be understand as the preparation of the model for the page prediction such as creating the semantic network, ontology, analyzing the web logs, etc. After this step next step is to test the model where different web logs are pass in the model and it will predict the next page by the model.

#### Model Preparations:

In order to prepare the model for the page prediction all the feature of the web mining are use such as content, structure and logs. Different steps which is requiring for the prediction are shown in fig 1 and 2 and explanation of each are given below:

**Step 1:** In this step first Collect the content of all the Web-pages from the website this then store it in the text file which will be read from that file and utilize the words present in the page for the effective prediction as this represent the user interest.

**Step 2:** In this step the semantic network of the keywords are prepare this is term as – TermNetWP [12]. So in order to prepare the TermNetWP (semantic network). H matrix need to utilize as this can be seen as the graph where nodes act the page number and keywords while edge act as the link between the nodes. Here one has H matrix that contain both the keyword as well as the page number, it is possible that one keyword is present in more than one page. So the keyword is share by both the page and the link is made between those pages one more information is maintains is the number of occurrence of the keyword.

**Step 3:** In this step a new feature is introduce that is web log which is use for the construction of Domain Ontology (DomainOntoWP). In order to utilize this one has to pre-process this data as well. As the Web user dataset is a collection of the date, time, and sequences of pages. Now in

Order to work on this data, preprocessing is required for taking the required data [9, 13].

**Step 4:** Here in this step Frequent Web Access Pattern (FWAP) are generate with the new approach from the [12] where it was use for the link prediction by utilizing the access feature that is Markov model. In this technique let the third order Markov is use it means that the page are in sequence, then as per the different sequences obtain the most visited page is present for the pattern which act the most promising pattern. In this way it will give a rank to the different page sequence from the web log that can utilize for the filtering of the next page as well.

**Step 5:** Here in this step Frequently Viewed Term Pattern (FVTP) of a web log is constructing from the matrix which collect the terms in the matrix page wise. So a Bag of words having current pages terms are collect here, which can be use for the next page prediction as well. This can be understood as if the page contains a term that contains most of the Frequently Viewed Term Pattern (FVTP) then it will be next page in this sequence.

**Testing Phase:** Here the dataset is again preprocessing for the web log portion in order to get the logs that are use for testing the built model. Pre-Processing steps are similar as done in step 3 of model preparation. The only difference here is that pre-processed logs are break such that each sequence first few pages are in the testing part and the next page after that sequence is store for the evaluation of result [8, 9].

#### Proposed Algorithm: Next page prediction:

Input: Web\_log, Web\_Content

OutPut: Prob[a,b], G[a,b]

Step I: Web\_Content  $\leftarrow$  Pre\_process(Web\_Content )

1. Loop i = 1: Number of Page
2. H[i] = pre\_process( W, H[i])
3. End Loop
4. Loop a  $\leftarrow$  Web\_Content
5. S = Support(H[a]);
6. If w\_supp > S
7. T[] = H[a]
8. End If
9. End Loop

Step II: Build the semantic network – TermNetWP

1. Loop a  $\leftarrow$  T[1...t] // represent terms
2. Loop b  $\leftarrow$  H[1...i] // Collection of Tems pagewise
3. G[a,b]  $\leftarrow$  compare(H[b], T[a])
4. End Loop
5. End Loop

Step III Web\_log  $\leftarrow$  Pre\_Process(Web\_log )

Step IV Prob[Pattern, page]  $\leftarrow$  Markov\_model(Web\_log)

In proposed algorithm

H = Set of terms present in content may have repeated value

T = Unique Set of terms present in content

Min\_supp = Minimum frequency of the term in the page

Prob = High score page set for the input page pattern

G = Terms pages graph

### Algorithm for Testing: Next Page Prediction

Input: Prob[a,b], G[a,b], Web\_log

OutPut: Final\_log

Step I: Generate testing logs for prediction

[Web\_log\_test, Web\_log\_result]  $\leftarrow$  Pre\_Process(Web\_log)

Loop a  $\leftarrow$  Web\_log\_test

Step II: Identifies a set of currently viewed terms

1. F = FVTP (Web\_log\_test[a], G[a,b])
2. Step 2 Find next viewed terms on the 1st-order TermNavNet
3. F\_T = TermNavNet(F);

Step III: Predict next pages mapped to next term on DomainOntoWP.

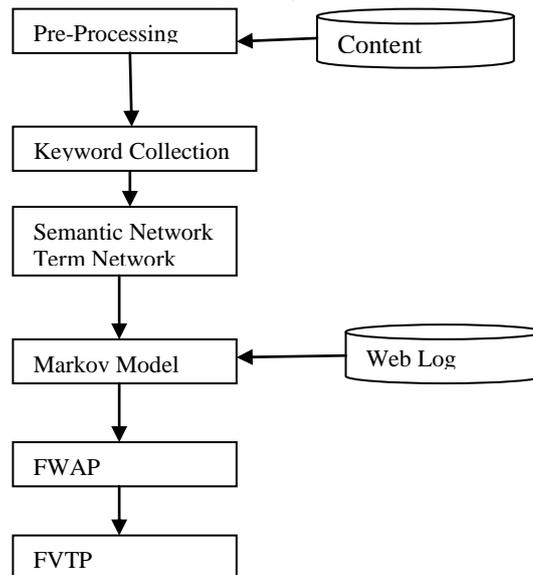


Fig. 1 Representing the Training phase.

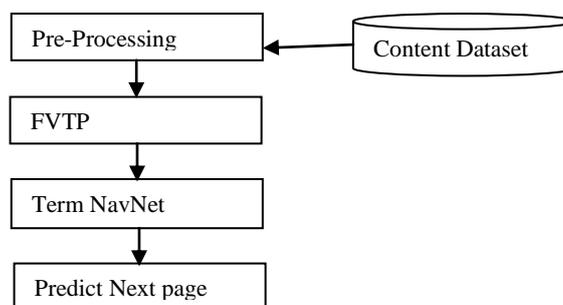


Fig. 2 Representing the Testing phase.

1. Predict\_pages  $\leftarrow$  (Prob[a,b], Web\_log\_test[a])
2. Final\_Page  $\leftarrow$  similar(Predict\_pages, F\_T)

End Loop  
 In above algorithm  
 F = Set of Frequent viewed terms  
 F\_T = Next Viewed Terms evaluate by the TermNavNet  
 Predict\_pages = High score page set for the input log  
 Final\_Page = Result of the system or predict page of the current log

#### IV. EXPERIMENT AND RESULT

##### Experiment Setup

This work used the Apriori algorithm [11, 14], which is a common algorithm to extract frequent rules. All algorithms and utility measures were implemented using the MATLAB tool. The tests were performed on an 2.27 GHz Intel Core i3 machine, equipped with 4 GB of RAM, and running under Windows 7 Professional.

##### Evaluation Parameter:

In order to evaluate this work there are different parameter present for the different techniques. The best parameter which suit this work is the precision where it give the value which is a measure of the prediction which is correctly identify by proposed model to the all the logs pass in the experiment [10, 12, 4]. The other important measure is the Satisfaction which is include new in this era. This is the ration of the prediction page which are satisfactory but not the actual or the correct prediction to the total number of logs pass in the model. This can be understand part of the web log pass for the test will predict one page then that page is compare with the next to next page of the log if it same then consider it as the satisfactory result.

Precision: In this evaluation parameter let us consider a Web\_log = {L1,L2, L3.....Ln}. Here L is the particular web page sequence such L1 = (P1, P2, P4, P5 , P6 , P9). In order to find the prediction the part of the log is pass in the system such as (P1, P2, P4) then for this pass correct prediction is P5 if the system generate the P5 value then consider it as the correct prediction otherwise consider it as the incorrect one.

So Precision = Rc/ R

Where Rc is the number of correct prediction

R is the total number of logs

Satisfaction : In this evaluation parameter let us consider a Web\_log = {L1,L2, L3.....Ln}. Here L is the particular web page sequence such L1 = (P1, P2, P4, P5, P6 , P9). In order to find the prediction the part of the log is pass in the system such as (P1, P2, P4) then for this pass satisfactory prediction is P6 if the system generate the P6 value then consider it as the satisfactory prediction otherwise consider it as the unsatisfactory one.

So Satisfaction = Rs/ R

Where Rs is the number of correct prediction

R is the total number of logs

##### Execution Time

Total Time for the execution of the algorithm in the prediction of the page base on the different size of dataset. This can be understand as the with the use of different number of web logs for initial training the total time of execution get differ. So if a dataset for training have more number of web log session then it will take more time, here time is calculated in seconds.

##### 5.1 Data Sets

We considered data sets, generate artificial by the website create for this work. This website consist of 16 pages which contain content of different field, this is use for the web content data. Then the structure of the website consists of the complete connection which can be understood as the total connection of one page with the other. Website has a function which automatically save the user behavior when it surf the different web page content.

Table I Dataset Summary

	Artificial
Total Session	10,000
Average Session Length	5
DataSet Time	Self

##### Prediction Setup

In order to evaluate proposed work method for prediction it is compare with the prediction algorithm use in [12] where most of the steps are same but the difference is of utilizing the whole content of the web. As in [12] only web heading or title is utilize for decision of choosing the page.

Given a testing session (t) of length L, we conduct prediction using the (L - 1)-gram Markov model and obtain the prediction to evaluate the precision of the model. Recall that the last page of t is the final outcome that we will evaluate the correctness of the mode against; hence, we use (L - 1)- gram. In case this longer than the highest N-gram used in the experiment, we apply a sliding window of size L on t. For example, suppose t = p1, p2, p3, p4, p5, if we use the third-order Markov model, then we break t into p1, p2, p3, p4.

Here precision is the measuring parameter which is taken as the on the data set with different training set.

Table1. Results representing prediction work in [12].

Dataset	60%	40%	20%
Precision	0.4719	0.4352	0.4096
Satisfaction	0.3544	0.4342	0.5397
Time	46.6869	23.30	18.9

Table2. Results representing proposed work.

Dataset	60%	40%	20%
Precision	0.4923	0.4452	0.4120
Satisfaction	0.3375	0.4335	0.5358
Time	55.063	35.976	19.9

As observed from the table 1 & 2 evaluation parameter values proposed work is getting good values from the previous while in case of the time as the whole content are evaluate here and out of which keywords are fetched so the proposed work execution time is little bit higher than previous but because of this precision value is quite high. This thing can be easily compromise by using efficient servers or by storing the page keywords in advance.

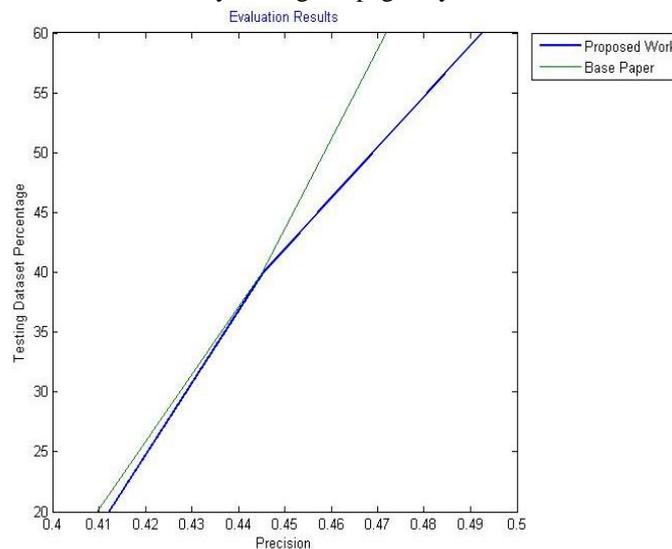


Fig . 3 Precision graphs for different size of data-set.

From above precision Fig. 3 of different training set it has been observed as the training data increase the precision also increase but by very small fraction only, with the include of both the feature results are more accurate than using the log session feature alone.

## V. CONCLUSIONS

World Wide Web has necessitated the users to make use of automated tools to locate desired information resources and to follow and asses their usage pattern. Web page pre fetching has been widely used to reduce the user access latency problem of the internet; its success mainly relies on the accuracy of web page prediction. Markov model is the most commonly used prediction model because of its high accuracy. But this work introduce new feature of web whole content and Markov model for prediction which has shows good results It is a powerful method for arranging users' session into clusters according to their similarity.

## REFERENCES

- [1] Balamash, M. Krunz& P. Nain. Performance analysis of a client-side caching/pre-fetching system for Web traffic. Computer Network. vol. 51, no. 13, pages 3673-3692, 2007.
- [2] B. niney. Concept-Based Approach for Off-line Web Site Enhancements," in Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on, 2008, pages 234-241.
- [3] Giorgos Kollias, Efstratios Gallopoulos, and Ananth Grama Surfing the Network for Ranking by Multidamping. IEEE transactions on knowledge and data engineering, vol. 24, 2013.
- [4] Hongxin Hu, Member, IEEE, Gail-Joon Ahn, Senior Member, IEEE, and Jan Jorgensen Multiparty Access Control for Online Social Networks: Model and Mechanisms.. IEEE transactions on knowledge and data engineering, vol. 25, no. 7,2013.
- [5] I. Zukerman, D. W. Albrecht & A. E. Nicholson. Predicting user's requests on the WWW. Proc. of the seventh international conference on User modeling, pages 275-284, 1999.

- [6] J. Domenech, J. Sahuquillo, J. A. Gil & A. Pont. The Impact of the Web Prefetching Architecture on the Limits of Reducing User's Perceived Latency. Proc. of the International Conference on Web Intelligence, 2006.
- [7] L. Fan, P. Cao, W. Lin & Q. Jacobson. Web Pre-fetching Between Low-Bandwidth Clients and Proxies: Potential and Performance. Proc. of the ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, pages 178-187, 1999.
- [8] M. Awad, L. Khan, and B. Thuraisingham, "Predicting WWW surfing using multiple evidence combination," VLDB J., vol. 17, no. 3, pages. 401–417, 2008.
- [9] N. R. Mabroukeh and C. I. Ezeife, "Semantic-Rich Markov Models for Web Prefetching," in 2009 IEEE International Conference on Data Mining Workshops Miami, Florida, USA, 2009, pages. 465-470. Springer-Verlag, 1999.
- [10] T. M. Kroeger, D.E. Long & J. C. Mogul. Exploring the Bounds of Web Latency Reduction from Caching and Pre-fetching. Proc. of the 1st USENIX Symposium on Internet Technologies and Systems, 1997.
- [11] The Next Page Access Prediction Using Markov Model "International Journal of Electronics Communication and Computer Technology (IJECCCT) Volume 1 Issue 1 | September 2011 ISSN: 2249-7838.
- [12] Thi Thanh Sang Nguyen, Hai Yan Lu, Jie Lu " Web-page Recommendation based on Web Usage and Domain Knowledge" 1041-4347/13/\$31.00 © 2013 IEEE.
- [13] V. Vapnik, The Nature of Statistical Learning Theory. New York: springer 1999.
- [14] Vivek Tiwari , Vipin Tiwari, S. Gupta. Association Rule Mining: A Graph based approach for mining Frequent Itemsets. IEEE International Conference on Networking and Information Technology (ICNIT 2010) Manila, Philippines, IEEE Catalog Number: CFP1023K-PRT, ISBN:978-4244-7577-3, 2010.