Integrating Markov Model with KNN Classification for Web Page Prediction

J.S.Raikwal
Department of Information Technology
I.E.T., DAVV, Indore, M.P.
INDIA

Rahul Singhai, Ph.D
Department of Computer Application
IIPS,DAVV,Indore, M.P.
INDIA

Kanak Saxena, Ph.D
Department of Computer Application
S.A.T.I,Vidisha M.P., INDIA

ABSTRACT
World Wide Web is growing rapidly in recent years. User's experience on the internet can be improved by minimizing user's web access latency. This can be done by predicting the next step taken by user towards the accessing of web page in advance, so that the predicted web page can be prefetched and cached. So to improve the quality of web services, it is required to analyze the user web navigation behavior. Analysis of user web navigation behavior is achieved through modeling web navigation history. Markov model is widely used to model the user web navigation sessions. Although traditional Markov models have helped predict user access behavior, they have serious limitations In this paper, we analyze and study Markov model and all-Kth Markov model in Web prediction. We propose new two-tier prediction frameworks that classify the user sessions, based on the KNN algorithm and then the Kth Markov Model is applied to predict the next web page. We show that such framework can improve the prediction time without compromising prediction accuracy and provides better performance over build time, search time, memory used and error rate.

Keywords
Marko model, Kth Marko model, Web page prediction, User's browsing behavior, Classification Algorithms, Web mining.

1. INTRODUCTION
Analysis and discovery of useful information from the World Wide Web is called as web mining. Based on the history of previous visit, the Web mining is used to predict patterns from web data that helps in personalizing the web sites. Such knowledge of user's history of navigation within a period of time is referred to as a session. These sessions, which provide the source of data for training, are extracted from the logs of the Web servers, and they contain sequences of pages that users have visited along with the visit date and duration.

Various Prediction models are used for addressing the Webpage prediction problem (WPP). These are generally classified as point-based and path-based prediction models. Path-based prediction is based on user's previous and historic path data, while point-based prediction is based on currently observed actions. Researchers have proved that the accuracy of point-based models is low due to the relatively small amount of information that could be extracted from each session to build the prediction model. Thus in the proposed work, we have focused on the path based prediction process.

The Markov Model is one of the widely used path based approach for web page prediction. The basic concept of Markov model is to predict the next action depending on the result of previous actions. In Web page prediction, the next action corresponds to predicting the next page to be visited. The previous actions correspond to the previous pages that have already been visited. The simplest Markov model predicts the next action by only looking at the last action performed by the user. In this model, also known as the first-order Markov model, each action that can be performed by a user corresponds to a state in the model. A somewhat more complicated model computes the predictions by looking at the last two actions performed by the user. This is called the second-order Markov model, and its states correspond to all possible pairs of actions that can be performed in sequence. This approach is generalized to the Kth-order Markov model, which computes the predictions by looking at the last K actions performed by the user, leading to a state-space that contains all possible sequences of K actions. Thus the First-order Markov models are not successfully used for WPP, because these models do not look far into the past to correctly differentiate the different behavioral modes of the different users. In order to obtained good predictions higher-order models must be used. But the higher-order models also have a number of limitations such as:

1. high state-space complexity
2. reduced coverage
3. Sometimes even worse prediction accuracy.

Thus we proposed a hybrid markov model that minimizes the above problem and gives better results and performance in terms of model building and prediction time.

2. RELATED WORK
The research for predicting user navigation behavior is one of the popular research problems in Web mining. A number of contributions in this field are available in the literature. Xing and Shen [2] proposed a hybrid-order tree-like Markov model that can predict Web access accurately, providing high coverage and good scalability. This HTMM model merges two methods: a tree-like structure Markov model method that aggregates the access sequences by pattern matching and a hybrid-order method that combines varying-order Markov models.

A trust prediction model for service Web is presented by Sherchan [3]. This prediction model uses Hidden Markov Model with multivariate Gaussian probability density functions for adjusting the effects of different QoS parameters on the reputation. In 2011, Khanchana and Punithavalli [4] have introduced a page rank based web page prediction system by considering the length of time spent on visiting the page and the frequency of the visited page. There experiments

incorporated higher order Markov models for successful predictions. Three different schemes for web prefetching and caching proposed by Nigam and Jain [5] are prefetching only, Prefetching with Caching and Prefetching from Caching. Prediction is done by modelling the web log using Dynamic Nested Markov model. This Dynamic Nested Markov model is analyzed on these three schemes. Nigam [7] also presented and analysed this Dynamic Nested Markov model on the basis of time complexity and coverage of the prediction state.

Borges and Levene [6] have given a method to evaluate the summarization of Variable Length Markov Chain (VLMC) model by applying a Sperman footrule metric. A comparison of traditional Markov models and new Markov models on the basis of prediction of user navigation behaviour is presented by Popa and Levendovszky [8]. This comparison suggests that new Markov models like Hybrid Order Tree Like model and Selective model provides good prediction. A model based on Hidden Markov Model for predicting the user browsing behaviour regarding e-business importance is proposed by Awad and Khalil [10].

In our system we are using Markov model to predict next web page, but we know that all Markov model are not able predict the next web page request by the user. But with some modifications, kth order Markov model is able to predict values for next web page access. To design and implement the effective and efficient model an enhanced approach for user's next web page prediction is proposed.

3. PROPOSED WORK

The central goal of this paper is to optimize the performance of the Markov model using a classifier named KNN. For experimental purpose training data set is used. This training data set is preprocessed using KNN classification algorithm. After classify the training data the kth order Markov model is applied to predict next page action.

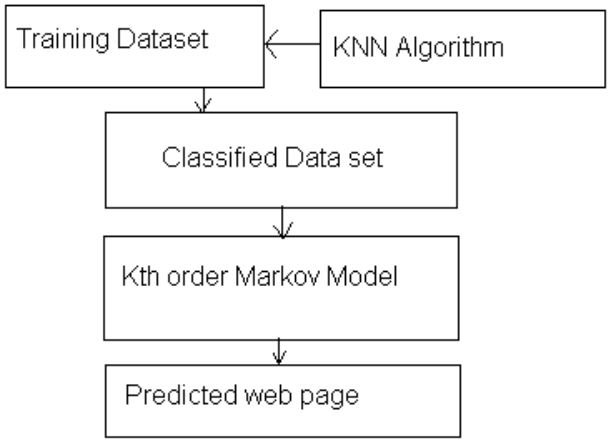


Fig 1: Proposed Model

The complete system (Fig 1) is divided into two parts. In the first part a training dataset is selected that is further classified into some classes using KNN classifier. This class indicates the pattern of data over given dataset. After classification the Markov Model is used to predict the next page request class.

The evaluation of prediction model is performed using n-cross validation process. It is mainly used in situations where the main goal of the model is prediction. It involves partitioning a sample of data into complementary subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (called the validation set or testing set). To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are averaged over the rounds.

4. IMPLEMENTATION

The implementation of the system is done using .net platform. Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source-control systems (like Subversion and Visual SourceSafe) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer). Due to this rich integrated development environment we select this framework for development.

To implement the proposed model the transactions are entered into the KNN algorithm to classify data. Transactions in this system are defined as pages which are visited during one user sessions. Suppose a web site has 5 pages and their names are home, about us, contact us, user registration, and user feed back respectably. Now if a user visiting sequence is like home → contact us → home → registration → about us → home. The recognition and implementation with this transaction steps are much complex so transactions are renamed in number format.

Home	0
About us	1
Contact us	2
User registration	3
And feed back	4

Thus above transaction is written as 0, 2,0,3,1. In the same manner suppose the transaction 1043 is selected for experimental study.

The collected data is now given in the following format

Transactions	States
1,2,2,0,1,2	3
0,2,1,3,1	4

KNN classifier is applied to these transactions. The classification of transaction is based over the below given pseudo code.

```
void classify(int k, Point[]
querySet, Point[] trainingSet) {
    // Loop 1.
    foreach (query in querySet) {
        // Loop 2.
        foreach (training in
trainingSet) {
```

```
            // Create a fixed sized
sorted map of length k,
            // where we map the
distance to a training point.
            SortedMap map = new
SortedMap(k);
 // Calculate distance of test point
to training point.
  double d = distance(query,
training);
 // Insert training point into sorted
list, discarding if training point
            // not within k nearest
neighbours to query point.
            map.insert(d, training);
        }
          // Do majority vote on k
nearest neighbours and
        // assign the corresponding
label to the query point.
        query.label =
majorityVote(map);
    }
}
 // We use the quadratic euclidean
distance.
double distance(Point query, Point
reference) {
    int sum = 0;
     // n is the number of dimensions
in the vector space.
    int n = query.vector.length;
     // Loop 3.
    for (int i = 0; i < n; i++) {
        int difference =
reference.vector[i] -
query.vector[i];
        sum += difference *
difference;
    }
   return sum;
}
```

After classification of data, these can now be grouped in the following manner.

Table 1. shows the dataset format

Transactions	Class	State
1,2,2,0,1,2	A	3
0,2,1,3,1	B	4
1,2,3,0,1,1	C	4
1,2,0,0,0,2	A	3
2,1,2,3,3,0	A	4
1,2,2,3,3,3	B	3

4.1 Data format

Since the raw data are unstructured, unorganisied and individual listings aren't always complete as far as the fields listed above are concerned that leads to some serious problems .Thus to overcome these issues, the log is prepared in a table format named CSV format of dataset in given below. In the dataset transactions shows the page action takes place during user session. Class field shows the class of transaction according to the transaction pattern.

K-nearest neighbor algorithm (k-NN) is a method used for classifying objects based on closest training examples in the feature space. K-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification.

The k-NN algorithm can also be adapted for use in estimating continuous variables. One such implementation uses an inverse distance weighted average of the k-nearest multivariate neighbors. This algorithm functions as follows:

a) Compute Euclidean or Mahalanobis distance from target plot to those that were sampled.
b) Order samples taking for account calculated distances.
c) Choose heuristically optimal k nearest neighbor based on RMSE done by cross validation technique.
d) Calculate an inverse distance weighted average with the k-nearest multivariate neighbors.

Using a weighted k-NN also significantly improves the results: the class (or value, in regression problems) of each of the k nearest points is multiplied by a weight proportional to the inverse of the distance between that point and the point for which the class is to be predicted.

After classification of algorithm we apply this data over Markov model the data being input is in class wise manner. suppose all data is classify in three classes A, B, and C. First of all we provide then first of all the input for class A is provided and then for class B and class C.

The pseudo code for this model is given below

```
for each set of sequences associated
with a user session
{ while (there is sub-sequence s that
has not been considered starting from
first request)
        for i from 0 to minimum (|s|,
Markov model order)
                {
        let ss be  the subsequence
containing last i items  from s
        let p be a pointer for
pointing a selected item
                if |ss|==0
        increment p.selfcount  else{
for   j from first(ss) to last(ss)
{
increment p.count
if not-exist (p,j)
increment p.No
add a new node for j to the list of
p's
let p point to j
if j=last(ss) increment p.selfcount
} }
}
}
```

5. RESULTS

The efficiency of proposed model is simulated over build time, search time, memory used and error rate. The results of simulation study are presented in the form of table and graphs. For better accuracy experiments are repeated over different size of dataset.

Table 2. shows memory used

Dataset size	Markov Model	Markov Model + KNN
1043	46241	43432
924	43304	42273
784	43782	43450
566	39923	41263
442	38234	39245

Graph 1 shows memory used by system

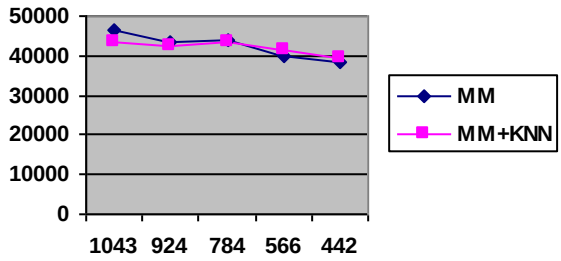


Table 2 and Graph shows that the memory used by the proposed system are comparatively more less than memory required by simple Markov Model for the different sizes of dataset.

Table 3 and graph 2 shows build time (built time is defined as time consumed to train the algorithm). Here it is clear that the time consumed to build proposed model is less than the time consumed for simple Markov Model.

Table 3. shows model build time

Dataset size	Markov Model	Markov Model + KNN
1043	743	678
924	732	1023
784	623	566
566	765	498
442	702	542

Graph 2 shows model build time

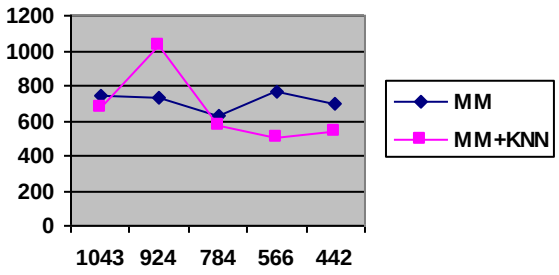
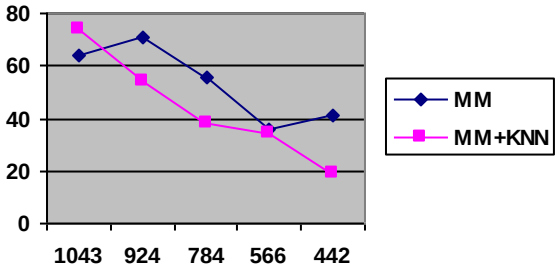


Table 4 shows model Search time

Dataset size	Markov Model	Markov Model + KNN
1043	64	74
924	71	54
784	56	38
566	36	34
442	41	19

Graph 3 shows model Search time



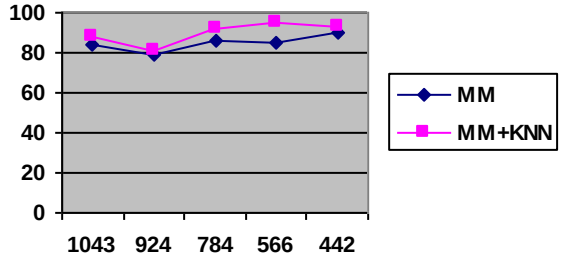
Search time is defined as time taken to predict and evaluate model. Graph 3 and table 4 much difference in search time in both cases.

Graph 4 and table 5 simulates the result of both algorithms. In both cases, the accuracy of proposed system is higher enough at all the cases.

Table 5 shows model Error Rate

Dataset size	Markov Model	Markov Model + KNN
1043	83.53	88.27
924	78.72	81.224
784	86.25	91.86
566	84.89	94.63
442	90.22	93.41

Graph 4 shows model Error Rate



6. CONCLUSION AND FUTURE WORK

The proposed model provides more accurate prediction with efficient utilization of memory. The time required to access a web page is also comparatively very less. Hence it is suggested to implement this model for web page prediction to personalize any web site. Although the suggested model gives improved performance over most of available traditional model. But the achieved performance is not satisfactory enough, thus future enhancements are required to design a model to achieve better performance..

7. REFERENCES

[1] Deshpande, M. and Karypis, G. 2001. Selective Markov Models for Predicting Web-Page Accesses. First SIAM International Conference on Data Mining, 1-15.

[2] Dongshan, X. and Xi'an, S. J. 2002. A new markov model for web access prediction. Computing in science and engineering, vol 4, issue 6, 34-39.

[3] Sherchan, W. 2011. A Trust Prediction Model for Service Web, International Joint Conference of IEEE TrustCom-11, 258-265.

[4] Khanchana, R and Punithavalli, M. 2011. An Efficient Web Page Prediction Based on Access Time-Length and Frequency, 3rd International Conference on Electronics Computer Technology (ICECT), Kanyakumari.

[5] Nigam, B. and Jain, S. 2010. Analysis of markov model on different web prefetching and caching schemes. IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 78-83, Coimbatore, India.

[6] Borges, J. and Levene, M. 2007. Evaluating Variable-Length Markov Chain Models for Analysis of User Web Navigation Sessions. IEEE Transactions on knowledge and data engineering, vol. 19, no. 4 .

[7] Nigam, B. 2010. Generating a New Model for Predicting the Next Accessed Web Page in Web Usage Mining, Third International Conference on Emerging Trends in Engineering and Technology, 485-490.

[8] Popa, R. and Levendovszky, T. 2008. Marcov Models for Web Access Prediction. 8th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics, 539-550.

[9] Fan, W. I., Chiu, I. and Lin, J. 2005. Prediction of the Intention of Purchase of the User Surfing on the Web Using Hidden Markov Model. International Conference on Services Systems and Services Management, Vol. 1.

[10] Awad, M.A. and Khalil, I. 2012. Prediction of User's Web-Browsing Behavior: Application of Markov Model. IEEE Transactions on systems, man, and cybernetics—part b: cybernetics, Vol 42, No. 4.

[11] Ahmed, N. S., Zafar, S. Asghar, S. 2010. Sequential Pattern Finding: A Survey. International Conference on Information and Emerging Technologies (ICIET), 244-249.

[12] Sarukkai, R. 2000. Link prediction and path analysis using markov chains. Ninth International World Wide Web Conference, Volume 33 Issue 1-6.

[13] Poonam Kaushal, Hybrid Markov model for better prediction of web page International Journal of Scientific and Research Publications , Volume 2, Issue 8, August 2012, ISSN 225-3153.