Dynamic and memory efficient web page prediction model using LZ78 and LZW algorithms

Alborz moghaddam * and Ehsanollah kabir**

* Tarbiat Modares University/Department of Electrical and Computer engineering, Tehran, Iran. Email: a.moghaddam@modares.ac.ir

** Tarbiat Modares University/Department of Electrical and Computer engineering, Tehran, Iran. Email: kabir@modares.ac.ir

***Abstract*—Web access prediction has attracted significant attention in recent years. Web prefetching and some personalization systems use prediction algorithms. Most current applications that predict the next user web page have an offline component that does the data preparation task and an online section that provides personalized content to the users based on their current navigational activities. In this paper we present an online prediction model that does not have an offline component and fit in the memory with good prediction accuracy. Our algorithm is based on LZ78 and LZW algorithms that are adapted for modeling the user navigation in web. Our model decreases computational complexities which is a serious problem in developing online prediction systems. A performance evaluation is presented using real web logs. This evaluation shows that our model needs much less memory than PPM family of algorithms with good prediction accuracy.**

***Keywords*—Web Page Prediction; LZ78; LZW**

I. INTRODUCTION

Web mining has become an important research area in recent years. It can be used to improve the web cache performance[1, 2], Detecting the user interests and so recommending related pages or goods for e-commerce web sites [3], improving search engines results [4] , and personalizing the web content as the users like [5]. In web page prediction understanding the user navigation pattern and then predicting the next pages is the main problem. With a reliable prediction system we can see the next action of web users.

In most of the web usage mining techniques, sequences are either used to produce association rules or to produce tree structures or Markov chains to represent navigation patterns. Markov models are based on a well-established theory and are simple to understand. They wildly used for modeling user navigation path. Prediction by Partial Match, PPM, [2] is a commonly used technique in Prediction. Improvements on the efficiency of PPM are examined in [6] where three pruning criteria where proposed: a) support-pruning, b) confidence-pruning and c) error-pruning. In this selective PPM model, some states of the predictor were pruned in case they do not appear very frequently. In [7], the authors proposed an online PPM model. They compute each node's outgoing entropy and then normalize it. Their algorithm always chooses the nodes with low normalized entropy. In their model, prediction accuracy rate and longest match rule are also applied. The LRS PPM, [8] stores a subset of all passes that frequently accessed. It uses longest repeated sequence to predict next request.

In recent years the content of many web sites are dynamic and new pages are also added to site dynamically. So we need a model that can be online and consider the changes of web site and user behavior. The memory efficiency is an important factor for an online algorithm. Most of the current models proposed for prediction are not online [8, 9] and need an offline component for processing tasks and online models such as model that proposed in [7] may soon become too big to fit in memory.

Offline user navigation modeling methods are suitable only for applications that are stable. When user requests inserted and deleted incrementally the online models are desirable. In this paper we present efficient techniques for modeling user navigation behavior. Our model is online so changes in user request patterns will update our prediction model incrementally. We do not build per-user predictive models. Individual models require more space, and may be less accurate because they see less data than a global model [10]. Our algorithms are based on LZ78 and LZW algorithms. These algorithms are compression algorithms and also are used in sequence mining. We use these algorithms for modeling the user navigation history. The structure of the paper is as follows. In section 2, we overview some related works. An overall view of our online prediction model is proposed in section 3. Section 4 presents the user navigation modeling using LZ78 and LZW algorithms. In section 5 we provide experimental results, and in section 6 we conclude and discuss future directions.

II. RELATED WORKS

Various models have been proposed for modeling the user navigation behavior and predicting the next requests of users. According to [12], association rules, sequential pattern discovery, clustering, and classification are most popular methods for web usage mining. Collaborative filtering is another method for modeling users' behaviors. Association rules [6] were proposed to capture the co-occurrences of buying different items in a supermarket shopping. Association rules indicate groups that are related together. Methods that use association rules can be found in [6, 7]. Collaborative filtering techniques are often

based on matching the current user's profile against similar data obtained by the system over time from other users. It tries to make useful recommendations users based on discovered cluster of similar categories. But for web sites where the number of web pages is quite large it can be quite inefficient.

The techniques that rely on sequential patterns such as Markov models and sequential association rules mining contain more precise information about users' navigation behavior. Association rules were proposed to capture the co-occurrence of buying different items in a supermarket shopping [11]. Association rules indicate groups that are presented together. In [12] study on different kinds of sequential association rules for web document prediction is proposed. It shows how to construct the association rule based prediction models for web log data.

The prediction scheme described in [1] used a dependency graph, DG to model user navigation behavior. The DG prediction algorithm constructs a dependency graph that describes the pattern of user page requests. Every page that visited by users is represented by a node in the dependency graph.

There is an arc from node A to B if and only if at some point in time a client accessed to B within w accesses after A, where w is the lookahead window size. The weight of the arc is the ratio of the number of accesses to B within a window after A to the number of accesses to A itself. A DG is effectively a first-order Markov model. In this method the consecution of requests is not considered. A first-order Markov model for a collection of user navigation sessions is proposed in [13], this method is extended in [14] to represent higher order conditional probabilities by making use of a cloning operation.

One of the famous predictors is all kth-order Prediction-by-Partial-Match, PPM predictor. All-kth-Order Markov model maintains Markov predictors of order i, for all $1 \leq i \leq k$ [2]. This model has improved prediction coverage and accuracy but the number of states in the model grows exponentially when the order of model increases. Lower order Markov models are not very accurate in predicting the user's browsing behavior, since these models do not look far into the past to correctly discriminate the different observed patterns. Higher order Markov models give better prediction precision with reduced hit rate. Improvements on the efficiency of PPM were examined in [6]. Three pruning criteria are proposed: a) support-pruning, b) confidence-pruning, c) error-pruning. The subject of [6] is mainly the efficiency. The resulting model, called Selective Markov Model has a low state complexity. But this model is not online and cannot be incrementally updated. The Longest Repeating Subsequence, LRS PPM [8] stores a subset of all passes that are frequently accessed. It uses longest repeated sequence to predict next request. In this model each path occurs more than some Threshold T, where T typically equals one. In [9] Popularity-based PPM is proposed. In this model, the tree is dynamically updated with a variable height in each set of branches where a popular URL can lead a set of long branches, and a less popular document leads a set of short ones. It includes some optimization techniques to reduce the memory complexity. The study in [7] presents an online method for predicting next user request. In this model the entropy of a node is an important factor in prediction. But in this model the memory efficiency of algorithm are not considered.

The techniques mentioned above, work well for non-dynamic web sites that do not have complex structure and not dynamically generate web pages. Large number of states in markov models, pattern in sequential pattern and association rules led to need much runtime requirements such as memory and computation power.

III. OUR PREDICTION MODEL

In our prediction model we use LZ78 and LZW algorithms to model the user navigation behaviors. With these two algorithms we reduce the model complexity which is an important factor in online systems. The efficiency of these algorithms are evaluated and presented in section 5. An online model does not need to process historical data. When a user request a new web page, then following action taken.

• A filtering process is done to ban the images and videos and script files from further process.

• The user session corresponding to new page request is found. We use the IP address to distinguish the user session of new request.

• Updates are performed in prediction tree according to specified algorithm. In this paper we use LZ78, LZW, PPM and LRS algorithms for learning the prediction tree.

Fig.1 shows how a new request affects the prediction tree. If a user is idle for more than 30 minutes, we assume that the user session is expired. LearnString Method, update the prediction tree according to selected algorithm. In next section we explain haw a new request is inserted in prediction tree.

```
PTree: Prediction tree
while (there is Request)
{
Request req = GetNextRequest();
If IsFileter(req) continue;
userSession = GetUserActiveSession();
if(userSession is Expired)
{
LearnString(PTree,userSession);
Renew(userSession);//Empty userSession
}
AddRequestToSession(userSession,req);
pSet =  NextPrediction(PTree,userSession);
if(pSet!= null)
{
if (req in pSet)// we could predict the next requests
{positive++; // we have a correct prediction
else
{negative++;}// we have a wrong prediction
}
LearnString(PTree,userSession); //learn the string regarding to specified algorithm.
}
```

Figure 1. How a new request added to tree regarding to new user request.

IV. USER NAVIGATION MODELING USING LZ78 AND LZW ALGORITHMS

In this section we propose two algorithms that are based on LZ78 and LZW. LZ78 algorithm is proposed by Jacob Ziv and Abraham Lempel in 1977 [15]. LZW proposed by Terry Welch in 1984 [16]. We show that they can be used for implementing recommendation systems in online environments. An online prediction method needs not rely on time-consuming preprocessing of the available historical data in order to build a prediction model. The preprocessing is done when we have a new request. LZW and LZ78 basically are lossless data compression algorithms with good functionality. The most important part of these algorithms is the dictionary construction algorithm that we use it for creating the prediction model.

A. User Navigation modeling using LZ78

As we mentioned in previous section, LZ78 is a lossless compression algorithm. Fig.2 shows that how the dictionary constructed from sequences using LZ78. In web environment we use user web page requests sequence as input sequence of LZ78 algorithm. Fig.3 shows how prediction tree is constructed. In Fig.2 and 3 variable w is sequence that is saved in each user session. This algorithm can insert sequences with long length, but generally total number of sequences that inserted in tree is less than PPM algorithm. We explain this algorithm with an example. Suppose the user requests the pages ABABCBC sequentially. If we use the LZ78 algorithm, then the A, B, AB, C and BC should be inserted in the tree. In Table 1 the first row shows the user requests. The second row shows the sequences inserted in the tree and the third row shows the sequences that maintained in active user session. When a sequence is inserted in the tree the weights of edges that represent the pass from the root to the last request of sequence is incremented. Now assume that user B requests the sequence of pages ABCABCD. Table 2 shows the results. If user A requests ABABCBC and user B requests ABCABCD, the prediction tree is like Fig.4.

```
w := NIL;
while (there is input)
{
R := next symbol from input;
if (wR exists in the dictionary)
{
w := wR;
} else {
output (index(w), R);
add wR to the dictionary;
w := NULL;
}
}
```

Figure 2. The LZ78 dictionary construction algorithm

```
w := NIL;
while (there is Request)
{
R := next request
if (wR exists in the prediction tree)
{
w := wR;
} else {
Add wR to tree;
Update weights of path that represent wR;
w := NULL;
}
}
```

Figure 3. Inserting a new user sequence in the prediction tree using LZ78

TABLE 1: Sequences inserted in tree and sequences maintained for a user session

Request	A	B	A	B	C	B	C
Sequence that inserted in tree	A	B	-	AB	C	-	BC
Sequence that maintained for a user	-	-	A	-	-	B	

TABLE 2 : Sequences inserted in tree and sequences maintained for a user session after user B requests ABCABCD

Request	A	B	C	A	B	C	D
Sequence that inserted in tree	-	-	ABC	-	-	-	ABCD
Sequence that maintained for a user	A	AB	-	A	AB	ABC	-

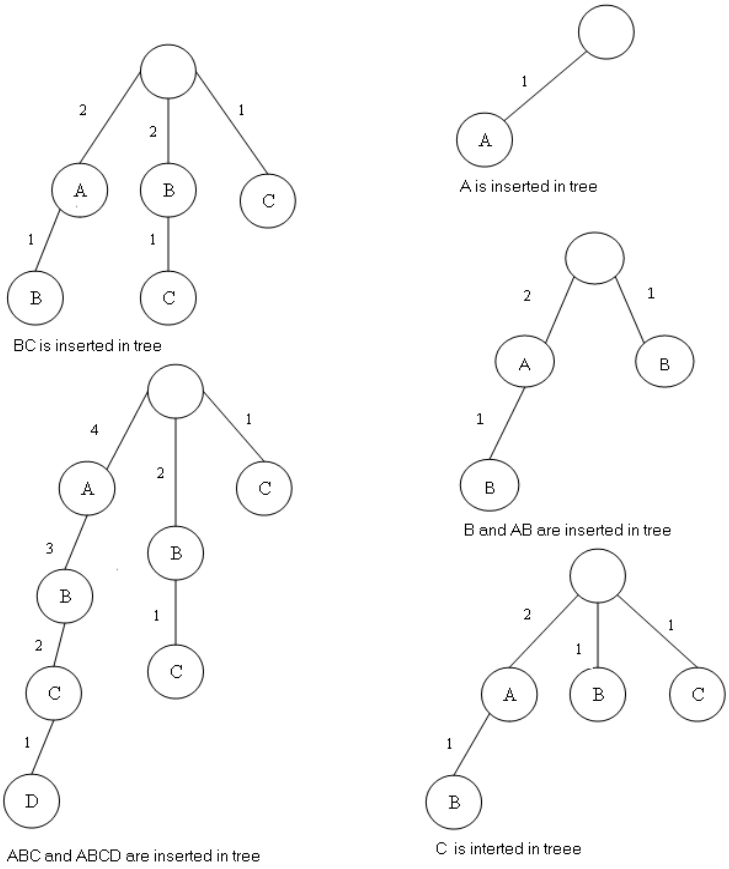

Figure 4. Snapshots of prediction tree that created by LZ78.

As you can see in Table 1 and Table 2, a small sequence is maintained for user session. The experimental result shows that the node counts of prediction tree in LZ78 are less than LZW and PPM algorithms. On the other hand some sequences are lost and do not inserted in tree. For Example pages A and C come after B, but the tree does not show these pages. Because of this, the hit rate of LZ78 is less than others but its memory efficiency is better

B. User navigation modeling using LZW

LZW is an improved version of LZ78 that is used in data compression. We will see that LZW has better performance in web prediction too. In the basic LZW algorithm, the dictionary contains all alphabets that may be seen in the sequence at the beginning. But in modeling web users we do not care it. Fig.5 shows that how the dictionary constructed from sequences using LZW.

```
w := NIL;
While (there is input) {
R := next symbol from input;
If (wR exists in the dictionary) {
w := wR;
} else {
Output (index(w), R);
Add wR to the dictionary;
w := R;
}
```

Fig 5: The LZW dictionary construction algorithm

As you have seen in Fig.5, in LZW, the last input is remained and the sequence is not cleared. Fig.6 shows how a new request from a user is inserted in the prediction tree using LZW algorithm.

If we use LZW algorithm, then for ABABCBC, the sequences, AB, BA, ABC, CB, and BC are inserted in tree and for ABCABCD, the sequences ABCA and ABCD are inserted in tree. Table 3 and Table 4 show the sequences that are inserted in prediction tree and sequences that are kept in user session. As you can see in these tables, LZW keeps more sequences than LZ78. Fig.7 shows the resulting LZW prediction tree.

```
w := NIL;
while (there is Request)
{
R := next request
if (wR exists in the prediction tree)
{
w := wR;
} else {
Add wR to tree;
Update weights of path that represent wR;
w := R;
}
}
```

Fig 6: Inserting a new user sequence in the prediction using LZW.

TABLE 3: Sequences inserted in tree and sequences maintained for a user session

Request	A	B	A	B	C	B	C
Sequence that inserted in tree	-	AB	BA	-	ABC	CB	-
Sequence that maintained for a user	A	B	A	AB	C	B	BC

TABLE 4: Sequences inserted in tree and sequences maintained for a user session

Request	A	B	C	A	B	C	D
Sequence that inserted in tree	-	-	-	ABCA	-	-	ABCD
Sequence that maintained for a user	A	AB	ABC	A	AB	ABC	D

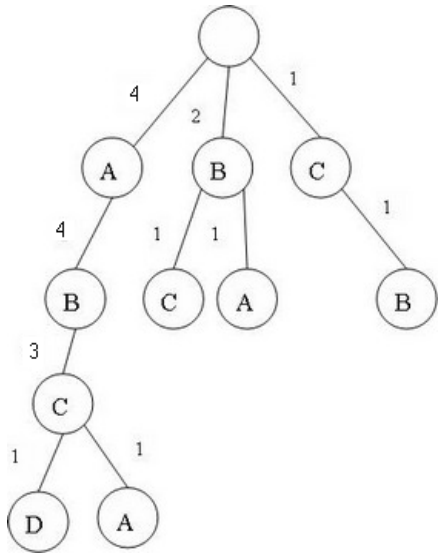

Figure 7. The prediction tree that created by LZW algorithm. AB, BA, ABC, CB and BC for sequence ABABCBC and ABCA, ABCD For sequence ABCABCD are inserted in tree.

V. EXPERIMENTAL RESULTS

The web log we tested was NASA log from the NASA Kennedy Space Center server in Florida available from the site http://ita.ee.lbl.gov/html/traces.html. It contains 1,569,898 requests and 72,198 IPs and 4,737 pages. This log contains 1,654,882 requests and 85,137 IP's from August 28th, 1995 to September 3rd, 1995. In our Evaluations, if a user is idle for more than 30 minutes, we assume that the next request from the user starts a new session. A longest matching method is used in all of tested models. For standard PPM model we set its maximum context length 5. For all algorithms we used a global model for prediction and proposed 10 pages as a prediction set. We used the hit-rate metric and precision metric to evaluate our methods. If any one of the ten recommended pages is the next request of the user, our prediction is correct.

Precision: The ratio of correct prediction divided by the number of requests that the model has prediction for them.

Hit rate: The ratio of correct predictions divided by total number of requests.

All Experiments are performed on a Pentium 4, Core 2 Duo 2.33 GHz with a 1G main memory running Microsoft Windows XP 2002.

Fig.8 compares the precision of four models. The precision of the LZW is higher than others.

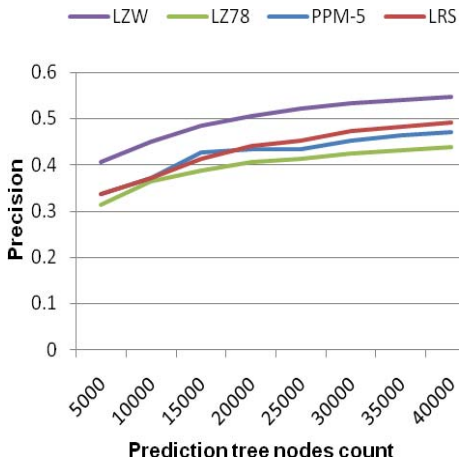

Figure 8. Prediction precision vs prediction tree nodes count

Fig.9 compares the hit rate of four models. Hit rate of LZW is higher than other tested models. PPM maintains all sequences that their lengths are less than specified context length. In spite of that hit rate of PPM is not higher than LZW and LRS. LZ78 has lowest hit rate.

In an online system the memory requirement of a model is an important factor. Fig.10 shows that LZ78 and LZW consume much less memory than PPM and LRS models and the difference is not negligible.

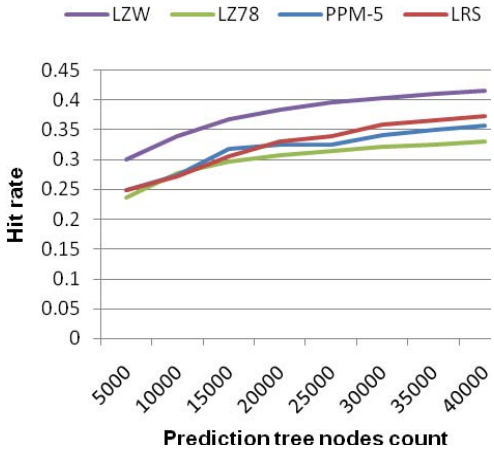

Figure 9. Prediction hit rate vs prediction tree nodes count

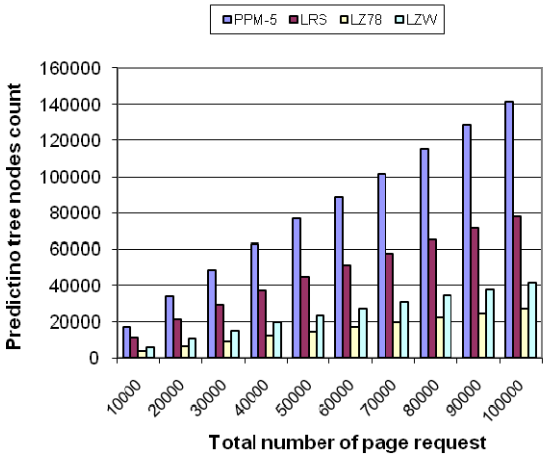

Figure 10. Prediction tree nodes count vs page requests count.

PPM-5 is also more time consuming than other models, because PPM-5 model is larger than others. Fig.11 shows the time spent for processing 100000 page requests. The time that spent for processing user request is dependent on model size. LZ78 and LZW spend less time to process page requests because of their smaller model.

As you have seen in this section, LZW has the best prediction and hit rate. But LZ78 consume less memory and spent less time to create its model. This is a tradeoff between the memory consumption and hit rate. If the memory and speed are critical, LZ78 can be used. In such systems the decrease in required memory is more important than low hit rate. But LZW has suitable memory efficiency for online environments too.

VI. CONCLUSION AND OPEN ISSUES

In this paper we explore predictive modeling techniques that attempt to reduce model complexity while retaining predictive accuracy. Modeling and predicting user surfing paths involves tradeoffs between model complexity and predictive accuracy. While most of the previous works in user navigation modeling and prediction were offline, we propose an online model based on LZ78 and LZW algorithms for modeling user navigation path. Our models need no training or preprocessing of the historical data. Our experimental results show that our methods consume much less memory than others. LZ78 consume less memory than LZW, but has less hit rate and precision. In LZW last input is remained and so it can contain more sequences than LZ78. LZW has the best precision and hit rate and also needs less memory and computational power than other models. It seems that LZW algorithm is the best if we consider the precision, memory efficiency and running speed together. The online manner of our methods makes them suitable for dynamic web sites. For further works, using the features of text contents and interface can be considered. Using this features make web page prediction algorithms more precise and useful.

ACKNOWLEDGMENT

This work was supported in part by the Iran Telecommunication Research Centre, ITRC, under Contract T-500-20593, TMU-87-01-08.

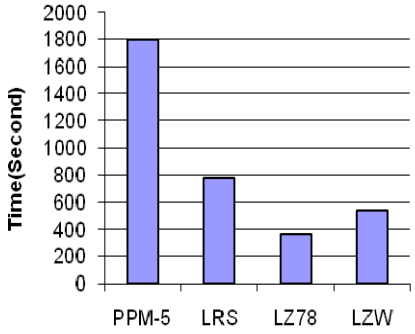

Figure 11. Time spent to create prediction tree after processing 100000 page requests.

REFERENCES

[1] V. N. Padmanabhan and J. C. Mogul, "Using predictive prefetching to improve World Wide Web latency." vol. 26, 1996, pp. 22-36.

[2] T. Palpanas, Web Prefetching Using Partial Match Prediction: National Library of Canada= Bibliothèque nationale du Canada, 2000.

[3] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," ACM Press New York, NY, USA, 2000, pp. 158-167.

[4] D. Zhang and Y. Dong, "A novel Web usage mining approach for search engines." vol. 39: Elsevier, 2002, pp. 303-310.

[5] S. S. Anand, P. Kearney, and M. Shapcott, "Generating semantically enriched user profiles for Web personalization," ACM Press New York, NY, USA, 2007.

[6] M. Deshpande and G. Karypis, "Selective Markov models for predicting Web page accesses." vol. 4: ACM Press New York, NY, USA, 2004, pp. 163-184.

[7] Z. Ban, Z. Gu, and Y. Jin, "An online PPM prediction model for web prefetching," ACM New York, NY, USA, 2007, pp. 89-96.

[8] J. Pitkow and P. Pirolli, "Mining longest repeating subsequences to predict world wide web surfing," USENIX Association Berkeley, CA, USA, 1999, pp. 13-13.

[9] X. Chen and X. Zhang, "A Popularity-Based Prediction Model for Web Prefetching," IEEE Computer Society, 2003.

[10] B. D. Davison, "Learning Web Request Patterns," Springer, 2004.

[11] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining—a general survey and comparison." vol. 2: ACM Press New York, NY, USA, 2000, pp. 58-64.

[12] Q. Yang, T. Li, and K. Wang, "Building Association-Rule Based Sequential Classifiers for Web-Document Prediction." vol. 8: Springer, 2004, pp. 253-273.

[13] J. Borges and M. Levene, "Data Mining of User Navigation Patterns," Springer-Verlag London, UK, 1999, pp. 92-111.

[14] J. Borges and M. Levene, "Generating dynamic higher-order Markov models in web usage mining." vol. 3721: Springer, 2005, pp. 34-45.

[15] Jacob Ziv and Abraham Lempel; A Universal Algorithm for Sequential Data Compression, IEEE Transactions on Information Theory, 23(3), pp.337-343, May 1977.

[16] Welch, T. A. 1984. A Technique for High-Performance Data Compression. Computer 17, 6 (Jun. 1984), 8-19