Research Journal of Applied Sciences, Engineering and Technology 9(1): 40-46, 2015
DOI:10.19026/rjaset.9.1374
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: July 01, 2014 Accepted: August 26, 2014 Published: January 05, 2015

Research Article

Combining Pre-fetching and Intelligent Caching Technique (SVM) to Predict Attractive Tourist Places

[1]K.R. Baskaran and [2]C. Kalaiarasan
[1]Department of Information Technology, Kumaraguru College of Technology, Coimbatore, India
[2]Tamilnadu College of Engineering, Coimbatore, India

Abstract: Combining Web caching and Web pre-fetching techniques results in obtaining the required information almost instantaneously. It also results in improved bandwidth utilization, load reduction on the origin server and reduces access delay. Web Pre-fetching is the process of fetching some of the predicted Web pages in advance which is assumed to be used by the user in the near future and the caching is the process of storing the pre-fetched Web pages in the cache memory. In the literature many interesting works have been reported separately for Web caching and for Web pre-fetching. In this study we combine pre-fetching (using clustering) and caching (using SVM) to keep track of the tourist spots that are likely to be visited by the tourists in the near future based on the previous history of visits. With the help of real data it is demonstrated that our approach is superior than clustering based pre-fetching technique using traditional LRU based caching policy which does not use SVM.

Keywords: Classification, clustering, confidence, hit-ratio, support, Support Vector Machine (SVM)

INTRODUCTION

With rapid growth of WWW, there is rising demand for computer networking resources. With more and more of Web based applications being created and used by users, the increase in bandwidth does not address the delay problems (Podlipnig and Boszormenyi, 2003). To reduce the access delay experienced by users, it is wise to predict and pre-fetch Web objects based on users access patterns and cache them. The existing prediction algorithms often predict both relevant and irrelevant pages. In caching the pre-fetched Web pages, efficient cache replacement techniques have to be deployed to manage the cache content. Many times it is found that the traditional cache replacement techniques used does not increase the cache hit ratio to a great extent and also they lead to cache pollution. Hence there is a need for using intelligent caching techniques to improve the efficiency of Web caching methods (Ali *et al*., 2012). Information about intelligent caching methods is found in Ali *et al*. (2011).

Web caching and web pre-fetching: Enhancing the performance of Web based systems is possible by Web caching in which, the Web objects which have high probability of being accessed in the near future are kept closer to the user either in the client's machine or in the proxy server. Web caching is useful in reducing the latency perceived by the user, decreases the bandwidth utilization and reduces the load on the server.

The following factors (features) of Web objects (Tourist places) that influence Web proxy caching are considered in our work.

Frequency: Number of requests made to an object.

Rank: User preference in selecting and visiting the tourist place.

Size: Details (in KB) of the requested Web object (tourist place).

Hit Ratio (HR) is used to analyze the performance of Web caching method. HR is the percentage of number of requests that are served by the cache over the total number of requests. A high HR indicates the presence of the requested object in the cache most of the time. If caching and pre-fetching techniques are combined together, the hit ratio can be improved and the user-perceived latency can be reduced.

THE PROPOSED METHOD OF COMBINED CLUSTERING BASED PRE-FETCHING TECHNIQUE WITH MACHINE LEARNING TECHNIQUE

As shown in the Fig. 1, the choices of visits of tourist places based on the user's preference are identified from the raw input file obtained from the

Corresponding Author: K.R. Baskaran, Department of Information Technology, Kumaraguru College of Technology, Coimbatore, India

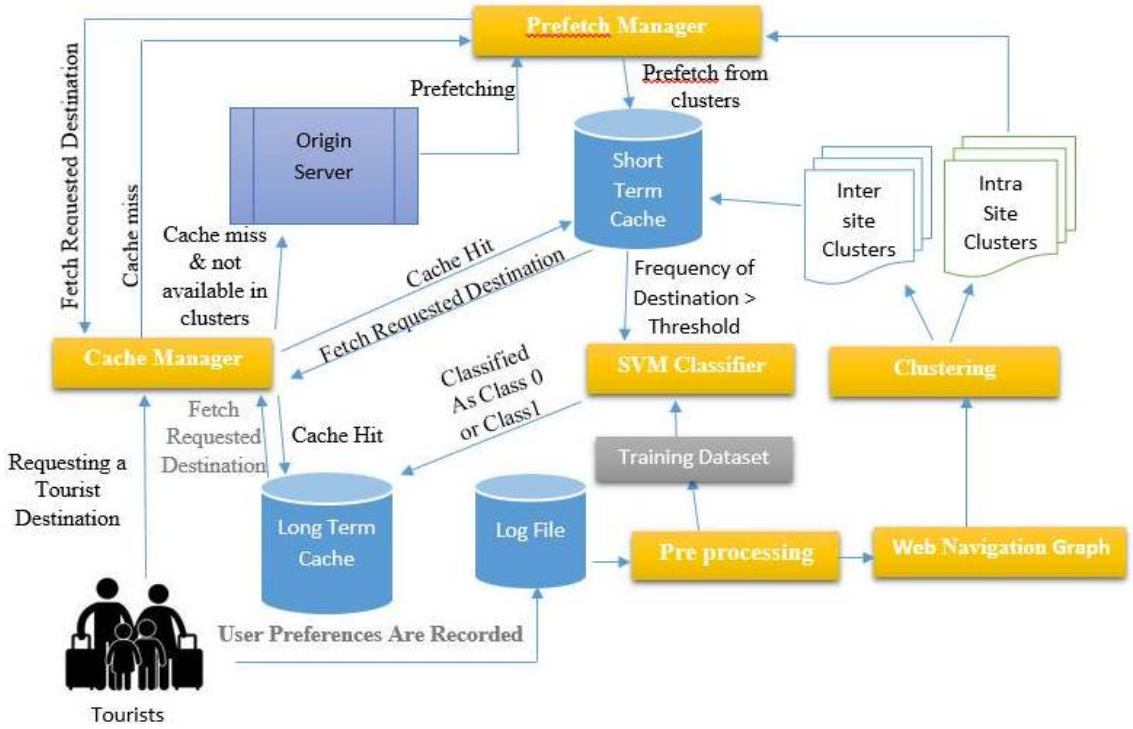

Fig. 1: Proposed system of combined web caching with pre-fetching

users. Users are requested to fill a questionnaire that asks for preferences (in order) of tourist places (cities) to be visited in a particular state in south India and various tourist spots in that city. The raw file contents are pre-processed (records that have incomplete data and having missing values are eliminated) and classified as Class 0 or 1 (Chen and Hsieh, 2006) based on the features namely: frequency, rank and size of object. The above said features are incorporated in to the SVM classifier and it is trained. Dataset is created from the raw file. Web Navigation Graph (WNG) is constructed from user's preference pattern of the various tourist destinations. WNGs show the navigations made by various users between various Web objects for inter-site clustering (between various cities in a state) and between various Web pages (various tourist spots with in a city) with in a Web object for intra-site clustering. Each node in the WNG represents a tourist place requested by the user and each edge represents user's transitions from one place to another and a weight is assigned to each edge which represents the number of transitions between those nodes. A clustering algorithm gets the contents of WNGs as inputs and two parameters namely Support and Confidence are used to keep track of frequently visited places by the user. By fixing a threshold value for these parameters, edges which have values less than this threshold can be removed (Pallis *et al*., 2008). Support is defined as the frequency of navigation between two nodes $u1$ and $u2$. The confidence is defined as $freq\ (u1, u2)\ /\ pop\ (u1)$ where $pop\ (u1)$ is the popularity of u1. Popularity of a node (place) is the number of incoming edges in to that node (place). The WNG is partitioned in to sub graphs (using Breadth-first search) by removing those edges that have low Support and Confidence values. The nodes in each connected sub-graph become a cluster.

Cache memory is divided into short-term cache and long-term cache. Two-thirds of the total cache space is allotted to short-term cache and the remaining one-third space is allotted to long-term cache. When the details of the user requested tourist place is neither found in the short-term cache or in the long-term cache then the details of the requested tourist place is fetched from the origin server and sent to the user and a copy of it is placed in the short-term cache. On the other hand when the details of the user requested Web object (tourist spot) is found in the short-term cache (cache hit), it is returned to the user and a search is made in that user's cluster for intra pages (local tourist spots of that city) of that Web object and other Web objects (not present in the short-term cache) present in that cluster. Details of those Web objects with intra pages if any are pre-fetched from the origin server and cached into the short-term cache by predicting that user will request for them in the near future, during the browser idle time (Pallis *et al*., 2008). The access count of the requested Web object is incremented by one in the short-term cache (if there is a cache hit). If this access count becomes greater than the threshold value chosen, that Web object is given as input to the SVM classifier for classification (Chen and Hsieh, 2006). If the Web object is classified as Class 1, then that Web object is

Chennai,Salem,Coimbatore,Trichy,NILGRIS,Trichy,
Coimbatore,NILGRIS,Trichy,NILGRIS,Trichy,Chenn
ai,Coimbatore,,Chennai,Coimbatore,NILGRIS,Trich
y,NILGRIS,Chennai,Trichy,Coimbatore,Trichy,Coim
batore,Trichy,NILGRIS,Trichy,NILGRIS,Trichy,Sale
m,Chennai,Salem,Chennai,Chennai,Trichy,Chenna
i,Trichy,Chennai,Trichy,Coimbatore,NILGRIS,Coim
batore,NILGRIS,Coimbatore,NILGRIS,Chennai,NILG
RIS,Chennai,NILGRIS,Chennai,NILGRIS,Chennai,Sal
em,Chennai,Salem,Coimbatore,Chennai,Coimbato
re,Chennai,Trichy,Salem,Trichy,Salem,Chennai,Coi
mbatore,Chennai,Coimbatore,Trichy,Chennai,Tric
hy,Chennai,NILGRIS,Coimbatore,NILGRIS,Coimbat
ore,Chennai,Trichy,Chennai,Trichy,Salem,NILGRIS,
Trichy,Chennai,Trichy,Chennai,Salem,Coimbatore,
Salem,Coimbatore,Chennai,Coimbatore,Chennai,S
alem,Chennai,Salem,Chennai,NILGRIS,Coimbatore

Fig. 2: Sample user access pattern

moved to the top of the long-term cache. If it is classified as Class 0, then it is moved to the bottom of the long-term cache. If sufficient space is not available in the long-term cache, then Web objects present in the bottom of the long-term cache are removed until sufficient space is made available for that Web object. If the details of the requested Web object is not available in the short-term cache (cache miss), a search is made in the long-term cache. If cache hit occurs in the long-term cache, it is returned to the user and the Web object is re-classified by the SVM. If classified as Class 1, it is moved to the top of the long-term cache else it is moved to the bottom. Pre-fetching of other Web objects and Web pages if any that belong to that cluster in to the short-term cache is initiated. LRU (Least Recently Used) technique is used for removal of Web objects from the short-term cache if sufficient space is not available for caching a new Web object.

For classification of a Web object as Class 0 or Class 1, the following strategy is followed.

If the frequency of visits of a Web object is < = 1, it is classified as Class 0, else if the frequency is >1, the description (size in KB) of the Web object is considered. If the description size is <600 KB, it is classified as Class 1 else its rank is considered. If the rank (preference) of that Web object is < = 2 it is classified as Class 1 else if its rank is >2, its frequency is again considered. If it is < = 40, it is classified as Class 0 else 1.

Figure 2 shows a sample user access pattern and Fig. 3 is the Wait-for graph constructed for the above user access pattern. In the Fig. 3, F stands for Chennai, Bi stands for Salem, BB stands for Nilgris, T stands for Trichy, Y stands for Coimbatore, ‘s’ stands for support value, ‘c’ stands for confidence value and Pop stands for popularity.

Web Navigation Graph (WNG): A weighted directed Web graph G (x, y) is used to represent the requests of

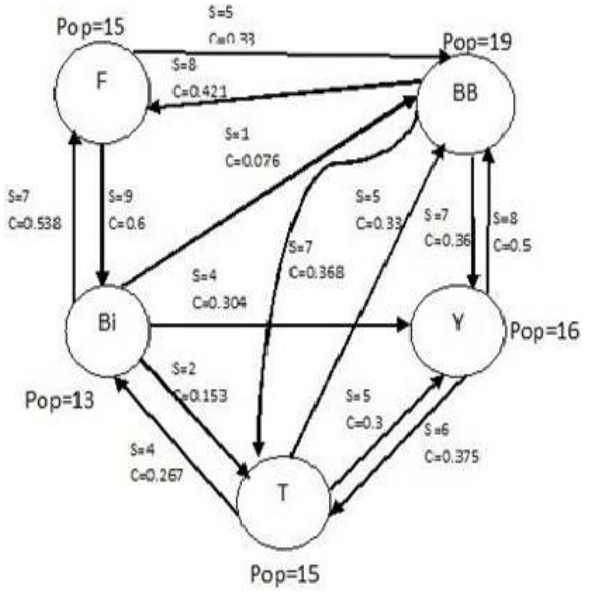

Fig. 3: Sample web navigation graph

each user, where each node in the WNG represents a tourist spot (city) and the edges in the graph indicate the number of transitions (visits) between the two tourist spots. To make the size of the WNG manageable, those edges whose connectivity between two Web objects is lower than a specified threshold are removed. Support and confidence are the two parameters that determine the connectivity between the two tourist spots. Let W : <xi, xj> be an edge from node xi to node xj.

Support of G, denoted by freq (xi, xj) is defined as the frequency of navigation steps between xi and xj. Confidence of g is defined as freq (xi, xj) /pop (xi), where pop (xi) is the popularity of xi. If the support threshold chosen is very less, too many less important user’s transitions for clustering may be included and if the chosen threshold value is high, many interesting transitions that occur at low levels of support may be missed.

Web clustering algorithm: The algorithm for clustering inter-site Web pages is described below. A weighted directed Web graph $G(x, y)$ is used to represent the access patterns of a user. The access patterns indicate a user’s interest to visit various tourist spots. This graph is partitioned into sub graphs by filtering those edges that have low support and confidence values. The Web objects (nodes) in each connected sub graph in the remaining navigational graph will form a cluster. To this clustering algorithm, Web navigational graph, support threshold and confidence threshold are given as inputs. Support and confidence value for each edge in the WNG is calculated and the popularity for each tourist spot is computed. All the edges with support or confidence value less than the corresponding threshold values are

removed. BFS (Breadth First Search) algorithm is applied to the navigational graph. BFS takes a node in the graph (called as source) and visits each node reachable from the source by traversing the edges. It outputs a sub-graph that consists of the nodes reachable from the source. This procedure is applied for all the nodes of the graph. All the nodes in each connected sub-graph forms a cluster. The time complexity of BFS is $O\ (|x| + |y|)$ where $|x|$ the number of nodes and $|y|$ is the number of edges in the graph (Pallis *et al*., 2008).

Survey of intelligent web proxy caching algorithms: Intelligent Web caching methods are more efficient than the traditional caching methods. Information about intelligent caching methods is found in Ali *et al*. (2011) and that of conventional replacement methods are found in Podlipnig and Boszormenyi (2003). In the literature details of many techniques used for cache replacement are found. Cache replacement policy based on Back-propagation neural network has been used in Neural Network Proxy Cache Replacement (NNPCR) (Cobb and ElAarag, 2008) and NNPCR-2 (Romano and ElAarag, 2011). A Web object is selected for replacement based on the rating returned by BPNN (Back-propagation neural network). However, the performance of BPNN in NNPCR or NNPCR-2 was influenced by the optimal selection of the network topology and its parameters that are based on trial and error method. BPNN learning process can also be time consuming and it did not take into account the cost and size in replacement decisions.

Combined BPNN as caching decision policy and LRU as replacement policy was proposed by Farhan, 2007. However recency factor which is considered as an important factor was ignored in the above technique. Farhan's approach was enhanced using particle swarm optimization by Sulaiman *et al*. (2008). This approach however did not incorporate superior classifier in Web caching decision. Koskela *et al*. (2003) used Multilayer Perceptron network (MLP) classifier in Web caching. HTML structure of the document and HTTP responses of the server were used as inputs to MLP to predict the class of Web objects. This class value was integrated with LRU, called LRU-C to optimize the Web cache. The frequency factor was however ignored in the replacement of cache contents. A logistic regression model to predict future requests was proposed by Foong *et al*. (1999). In his work objects with lowest re-access probability value were replaced first regardless of cost and size of the predicted object. From the above studies it is observed that intelligent caching techniques can be employed either individually or can be combined with LRU technique. Both of these approaches predict Web objects that will be re-accessed in the near future without considering the cost and size of the predicted objects for replacement. Extra computational overheads and longer duration are required for training process. In our work SVM machine learning technique is used to classify Web objects and make more accurate predictions.

Algorithm for the combined intelligent caching (SVM) and pre-fetching:

```
Begin
For each tourist place K requested by the tourists
If K is available in the short-term cache
Begin
    Cache hit has occurred
    Fetch the requested tourist place
    K from the short-term cache
    Update the information of K.
    increment the frequency of K
If the frequency of K> threshold
    limit
     Begin
    While no space available in the
    long-term cache for K
        Begin
        Expel f from long-term cache such that f is in
bottom of the long-term cache
        End
    Class of K = apply-svm (Common features)
        If class of K = 1
        Begin
Move K to top of the long-term
cache
        Else
Move K to the bottom of the long-
term cache
        End
      End
    End
Else if K is available in long-term
cache
    Begin
Cache hit has occurred
Fetch the requested tourist
place K from the long-term
cache
Update the information of k
and increment the frequency
of K
Recalculate the class of K
    If class of K = 1
    Begin
Move K to the top of the long-
term cache
    Else
  Move K to the bottom of the
  long-term cache
    End
End
    Else
     Begin
```

```
    Cache miss occurs
    Fetch K from original
    server
Cluster c = prefetch the inter site Cluster (K)
    If c is not NULL
      Begin
    While no space in short-
    term cache for tourist place
    in c
        Begin
    Expel object using LRU
from short-term cache
    End
Load all the clusters into the
    short-term cache
   End
  End
End
Procedure prefetch The intersite Cluster (m)
    Begin
    For each cluster p for the user
    If m is in cluster p p = p + prefetch The Intrasite
    clusters (p);
        return p
    If m is not in any of the cluster
        return NULL
        End
Procedure prefetch The intrasite Cluster (m)
        Begin
        For each cluster p
If intracluster object k is in cluster p
    k = k + get The Intrasite clusters (k);
    return k
If k is not in any of the cluster
    return NULL
   End
```

Clustering algorithm:

```
Begin
    For all Inter Site and Intra Site
    dataset available
    Begin
Construct Web navigation graph
G (U, V)
    End
For each G (U, V)
    Begin
    Calculate Support, Confidence and Popularity of
the node U
    If Confidence < Threshold limit of Confidence
        Begin
            Remove U
        End
    If Support < threshold limit of
        Support
        Begin
            Remove V
        End
Cluster C = Apply BFS for
G (U, V)
    End
End
```

Breadth first search:

```
Begin
Input: Graph G (U, V)
Choose some starting node u1
Mark u1 as visited
Initialize list x1 with u1
Initialize sub-graph T with u1
While x1 is not empty
    Begin
    Choose node x2 from front of the
    list
        Visit x2
    End
For each unmarked neighbor y
    Begin
   Mark y
   Add y to the end of the list x1
   Add x2->y to sub graph T
   End
Find all neighbors of the node u1
Visit each neighbor and mark the
visited node
End
```

Performance evaluation:

Dataset: Dataset details: The technique explained in this study is tested with a dataset. The dataset is obtained from an online spreadsheet data https://docs.google.com/spreadsheet/ccc?key=0As2P-0sENZe3dGg1bWs0R19mbldtdzRQNkJEbG1CZVE#gid=form. The raw file for the dataset contains the details of 300 user's preferences to visit various tourist destinations:

Total Number of Items: 1184
Total Size of Bytes for Dataset: ~ 122000
Total Number of Items used for Training: 795
Total Size of Bytes Used for Training: ~85400
Total Number of Items used for Testing: 224
Total Size of Bytes Used for Testing: ~ 36600

Seventy percent of all the requests ordered by time have been used for the user's access pattern analysis, creating training dataset and testing. The remaining 30% of the requests were used for testing the scheme.

EXPERIMENTAL RESULTS

Hit ratio analysis: In the above graph (Fig. 4) SVM-LRU means SVM-LRU caching with pre-fetching and LRU means LRU caching with pre-fetching.

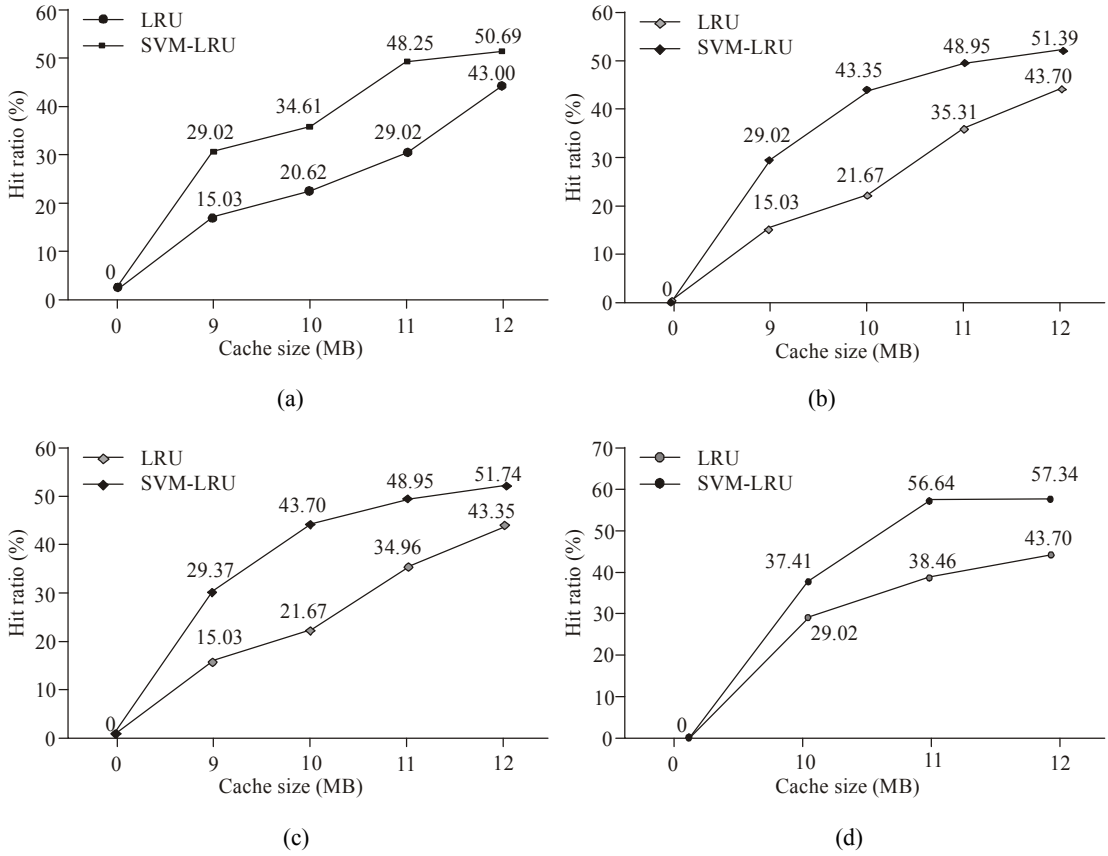

Fig. 4: Analysis of HR using SVM and LRU pre-fetching on different values of support and confidence, (a) support: 2 and confidence: 0.30, (b) support: 3 and confidence: 0.30, (c) support: 9 and confidence: 0.30, (d) support: 4 and confidence: 0.25

HR is calculated for different values of Support and Confidence using SVM-LRU caching with pre-fetching and LRU caching with pre-fetching. The performance of both the pre-fetching methods for various cache sizes are plotted in the above graphs. Results inferred from the above graphs are stated below:

- If the Confidence and Support value is increased, the number of clusters created will increase and the number of Web objects in the created clusters will decrease.
- For increase in Support and Confidence values, there will be a slight increase in HR for increasing cache sizes.
- Compared to LRU caching with pre-fetching, in SVM-LRU caching with pre-fetching there is significant increase in the Hit Ratio (HR) independent of the cache size, support and confidence values.
- Regarding Hit Ratio (HR), on an average 53% of the total size of the requested information is found fetched from Cache (cache hit) using SVM-LRU caching with pre-fetching and only 43% of the total size of the requested information is found fetched from cache using LRU caching with pre-fetching and remaining information are fetched from the original server. The above information shows the superiority of SVM-LRU caching with pre-fetching over LRU caching with pre-fetching.
- There will be a decrease in the access latency experienced by the user while using SVM-LRU caching with pre-fetching due to increase in cache hit percentage and thus the load on the original server will decrease.
- SVM-LRU caching with pre-fetching leads to decrease in the bandwidth utilization as a result of more cache hits and an increase in the efficiency of cache replacement.

CONCLUSION AND RECOMMENDATIONS

In this study, a clustering algorithm is used to cluster the tourist places represented in the Web navigation graph. Frequently used tourist places are kept tracked by the Confidence and Support values. If the details of the requested tourist places of the user are present in the short-term cache then the details of all the

other tourist places in that cluster along with the intra sites of that place are pre-fetched into the short-term cache during the browser idle time. If a tourist place and its associated tourist spots in the short-term cache are accessed number of times than a fixed threshold value then it is moved to long-term cache after classifying them using SVM algorithm. If the details of the requested tourist places are present in the long-term cache (miss in the short-term cache) it is returned to the user and the Web object is re-classified by the SVM. If classified as Class 1, it is moved to the top of the long-term cache else it is moved to the bottom. Pre-fetching of other Web objects and Web pages if any that belong to that cluster in to the short-term cache is initiated. If there is cache miss in both the caches, then the details of that tourist place are fetched from the origin server and it is given to the user. A copy of it is also placed in to the short-term cache. The efficiency of SVM pre-fetching is compared with that of LRU pre-fetching using real data set and it is demonstrated that SVM pre-fetching has high HR for various values of Support, Confidence and cache sizes. Extension of this work is possible by comparing the efficiency of other intelligent caching techniques with that of SVM technique.

REFERENCES

Ali, W., S.M. Shamsuddin and A.S. Ismail, 2011. A survey of web caching and prefetching. Int. J. Adv. Soft Comput. Appl., 3(1), ISSN: 2074-8523.

Ali, W., S.M. Shamsuddin and A.S. Ismail, 2012. Intelligent web proxy caching approaches based on machine learning techniques. Decis. Support Syst., 53: 565-579.

Chen, C. and C. Hsieh, 2006. Web page classification based on a support vector machine using a weighted vote schema. Expert Syst. Appl., 31: 427-435.

Cobb, J. and H. ElAarag, 2008. Web proxy cache replacement scheme based on back-propagation neural network. J. Syst. Software, 81: 1539-1558.

Farhan, 2007. Intelligent web caching architecture. M.A. Thesis, Faculty of Computer Science and Information System, UTM University, Johor, Malaysia.

Foong, A.P., H. Yu-Hen and D.M. Heisey, 1999. Logistic regression in an adaptive web cache. IEEE Internet Comput., 3(5): 27-36.

Koskela, T., J. Heikkonen and K. Kaski, 2003. Web cache optimization with nonlinear model using object features. Comput. Netw., 43: 805-817.

Pallis, G., A. Vakali and J. Pokorny, 2008. A clustering-based pre-fetching scheme on a web cache environment. Comput. Electr. Eng., 34: 309-323.

Podlipnig, S. and L. Boszormenyi, 2003. A survey of Web cache replacement strategies. ACM Comput. Surv., 35(4): 374-398.

Romano, S. and H. ElAarag, 2011. A neural network proxy cache replacement strategy and its implementation in the Squid proxy server. Neural Comput. Appl., 20: 59-78.

Sulaiman, S., S.M. Shamsuddin, F. Forkan and A. Abraham, 2008. Intelligent Web caching using Neurocomputing and particle swarm optimization algorithm. Proceeding of the 2nd Asia International Conference on Modeling and Simulation (AICMS, 08), pp: 642-647.