Journal of Intelligent Learning Systems and Applications, 2015, 7, 117-127
Published Online November 2015 in SciRes. http://www.scirp.org/journal/jilsa
http://dx.doi.org/10.4236/jilsa.2015.74011

Intelligent Dynamic Aging Approaches in Web Proxy Cache Replacement

Waleed Ali[1], Siti Mariyam Shamsuddin[2]

[1]Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University, Rabigh, Kingdom of Saudi Arabia
[2]Faculty of Computing, Universiti Teknologi Malaysia, Johor, Malaysia
Email: waleedalodini@gmail.com, mariyam@utm.my

Received 21 September 2015; accepted 10 November 2015; published 13 November 2015

Abstract

One of commonly used approach to enhance the Web performance is Web proxy caching technique. In Web proxy caching, Least-Frequently-Used-Dynamic-Aging (LFU-DA) is one of the common proxy cache replacement methods, which is widely used in Web proxy cache management. LFU-DA accomplishes a superior byte hit ratio compared to other Web proxy cache replacement algorithms. However, LFU-DA may suffer in hit ratio measure. Therefore, in this paper, LFU-DA is enhanced using popular supervised machine learning techniques such as a support vector machine (SVM), a naïve Bayes classifier (NB) and a decision tree (C4.5). SVM, NB and C4.5 are trained from Web proxy logs files and then intelligently incorporated with LFU-DA to form Intelligent Dynamic-Aging (DA) approaches. The simulation results revealed that the proposed intelligent Dynamic-Aging approaches considerably improved the performances in terms of hit and byte hit ratio of the conventional LFU-DA on a range of real datasets.

Keywords

Cache Replacement, Web Proxy Server, Dynamic-Aging Approaches, Machine Learning

1. Introduction

The most popular solution for improving Web performance is a Web caching technology. The Web caching technique is a very useful mechanism in reducing network bandwidth utilization, decreasing user-perceived delays, and reducing loads on the original servers. In Web proxy caching, Web objects that are likely to be used in the near future are kept in proxy servers, which play key roles between users and Web sites in reducing the response time of user requests and saving network bandwidth.

How to cite this paper: Ali, W. and Shamsuddin, S.M. (2015) Intelligent Dynamic Aging Approaches in Web Proxy Cache Replacement. *Journal of Intelligent Learning Systems and Applications*, **7**, 117-127. http://dx.doi.org/10.4236/jilsa.2015.74011

Due to cache space limitations, a Web proxy cache replacement is required to manage the Web proxy cache contents efficiently. In the proxy cache replacement, the proxy cache must effectively decide which objects are worth caching or replacing with other objects. The Web cache replacement is the core or heart of Web caching; hence, the design of efficient cache replacement algorithms is crucial for the success of Web caching mechanisms [1]-[9]. So, the Web cache replacement algorithms are also known as Web caching algorithms [1].

In the Web cache replacement, a few important features or factors of Web objects, such as recency, frequency, size, cost of fetching the object from its origin server and access latency of object, can influence the performance of Web proxy caching [1]-[11]. These factors can be incorporated into the replacement decision for better performance.

In Web proxy cache replacement, Least-Frequently-Used-Dynamic-Aging (LFU-DA) policy is a conventional Web cache replacement method, which attempts to remove the cache pollution in a LFU policy. LFU-DA accomplishes a superior byte hit ratio compared to other Web proxy cache replacement algorithms. However, the performance of LFU-DA in terms of hit ratio is not efficient enough since it does not consider the size or the download latency of objects.

Most of the conventional Web cache replacement approaches consider just some factors and ignore other factors that have an impact on the efficiency of the Web caching [1]-[11]. Moreover, a combination of these factors to get wise replacement decision is not a simple task. This is because one factor in a particular environment may be more important than other factors in other environments [1]-[11]. Therefore, a large portion of objects, which are stored in the cache, are never requested again or requested after a long time. This leads to cache pollution, where the cache is polluted with inactive objects. This causes a reduction of the effective cache size and negatively affects the performance of Web proxy caching. Even if we can locate a large space for the proxy cache, this will not be helpful since just the preferred objects are visited by users. Moreover, the searching for an object in a large cache needs a long response time and an extra processing overhead. Thus, the conventional Web proxy cache replacement policies are no longer efficient enough [11]. This is motivation to adopt intelligent techniques for solving the Web proxy caching problems. The second motivation behind the development of intelligent approaches in Web caching is availability of proxy logs files that can be exploited as training data. In a Web proxy server, Web proxy logs file records activities of the users and can be considered as complete and prior knowledge of future access. Therefore, the machine learning techniques can beutilized to produce an efficientand adaptive Web proxy cache replacement scheme, which can adapt to important updates and changes in the Web environment during the training phase.

Several research works have developed intelligent approaches that are smart and adaptive to the Web caching environment. A multilayer perceptron network (MLP) [1], back-propagation neural network (BBNN) [2] [3], logistic regression (LR) [4], multinomial logistic regression (MLR) [5], adaptive neuro-fuzzy inference system (ANFIS) [6], and others [7]-[9] have been utilized in Web caching. In most of these studies, an intelligent supervised machine learning technique was employed in Web caching individually or integrated just with Least-recently used (LRU) algorithm. In addition, they utilize an artificial neural network (ANN) in Web proxy caching although ANN training may consume more time and require extra computational overhead. Besides, ANN is influenced by the optimal selection of the network topology and its parameters. More significantly, integration of an intelligent technique in Web cache replacement is still a popular research subject [7] [8].

In this paper, alternative supervised machine learning techniques are proposed to improve the performance of Web proxy caching. Support vector machine (SVM), Naïve Bayes (NB) and decision tree (C4.5) are three popular supervised learning algorithms, which identified as three of the most influential algorithms in data mining [10]. SVM, C4.5 and NB have many advantages over ANN and other machine learning algorithms. They are very efficient, easy to construct with few parameters. In addition, C4.5 and NB are simple to understand and interpret. In addition, SVM, C4.5 and NB perform classifications more accurately and faster than other algorithms in a wide range of applications such as text classification, Web page classification and bioinformatics applications, medical filed, military applications, forecasting, finance and marketing [10]. Hence, SVM, C4.5 and NB classifiers can be utilized to produce promising solutions for Web proxy caching.

The proposed intelligent Web proxy replacement approaches are entirely different from the existing intelligent works. Our intelligent dynamic aging approaches is based on SVM, C4.5 and NB classifiers to combine the most significant factors for predicting probability that Web objects would be re-visited later. Then, unlike the existing intelligent works, the proposed intelligent dynamic aging approaches integrate classification information in proxy cache replacement decision as an alternative to the frequency factor in LFU-DA.

The remaining parts of this paper are organized as follows. Background and related works are presented in Sections 2 and 3. Web proxy cache replacement is presented in Sections 2, while Section 3 describes briefly machine learning classifiers used in this study. In Section 4, a methodology of Intelligent Dynamic-Aging (DA) approaches used for improving Web proxy replacement is explained. Section 5 presents and discusses results of intelligent Dynamic-Aging (DA) approaches compared with conventional Web proxy cache replacement technique. Finally, Section 6 summarizes and concludes the works presented in this paper.

2. Web Proxy Cache Replacement

Web caching becomes one of the most successful solutions for improving the performance of Web-based system. In Web caching, the popular Web objects that likely to be used soon are stored in positions closer to user like client machine or proxy server. Significantly, the Web caching has several attractive advantages:

- Reducing user perceived latency.
- Reducing network traffic and therefore reducing network costs for both content provider and consumers.
- Reducing loads on the origin servers.
- Increasing reliability and availability of Web and application servers.
- Reducing hardware and support costs.

Web caching technique can be implemented in various locations throughout the Web. The Web proxy caching is widely utilized by computer network administrators, technology providers, and businesses to reduce user delays and to reduce Internet congestion [1] [3].

The Web proxy cache replacement plays an extremely important role in Web proxy caching. Hence, the design of efficient cache replacement algorithms is required to achieve highly sophisticated caching mechanism. The effective cache replacement algorithm is vital and has a profound impact on Web proxy cache management [6]-[8] [11]. Therefore, this study pays attention to improvement of Web proxy cache replacement approaches. In general, cache replacement algorithms are also called Web caching algorithms [1].

As cache size is limited, a cache replacement policy is needed to handle the cache content. If the cache is full when an object needs to be stored, the replacement policy will determine which object is to be evicted to allow space for the new object. The optimal replacement policy aims to make the best use of available cache space, improve cache hit rates, and reduce loads on the origin server.

One of the simplest and most common cache replacement approaches is Least-Recently-Used (LRU) algorithm, which removes the least recently accessed objects until there is sufficient space for the new objects. LRU is easy to be implemented and proficient for caching uniform size objects such as the CPU cache. However, it does not perform well in Web caching since it does not consider the size or the download latency of objects [1].

Least-Frequently-Used (LFU) is another common Web caching policy that replaces the object with the least number of accesses. LFU keeps popular Web objects and evicts rarely used ones. However, LFU suffers from the cache pollution in objects with the large reference accounts, which are never replaced even if these objects are not re-accessed again [1] [12] [13]. As attempt to reduce cache pollution caused by LFU, Least-Frequently-Used-Dynamic-Aging (LFU-DA) is introduced by Arlitt *et al.* [14]. LFU-DA added dynamic aging to LFU policy. LFU-DA computes the key value $K(g)$ of object g using Equation (1).

$$K(g) = L + F(g) \tag{1}$$

where $F(g)$ is the frequency of the visits of object g, while L is a dynamic aging factor. L is initialized to zero, and then updated to the key value of the last removed object.

LFU-DA accomplishes a superior byte hit ratio compared to other Web proxy cache replacement algorithms. However, LFU-DA may suffer in hit ratio measure due to its retaining of the most popular objects; regardless of object size [2] [13] [14].

SIZE policy is one of the common Web caching policies. It replaces the largest object(s) from cache when space is needed for a new object. Thus, a cache can be polluted with small objects which will not be accessed again. To alleviate cache pollution in SIZE policy, Cao and Irani [15] suggested Greedy-Dual-Size (GDS) policy as an extension of SIZE policy. GDS policy integrated several factors and assigned a key value for each Web object stored in the cache. When cache space becomes occupied and a new object is required to be stored in the cache, the object with the lowest key value is removed. When the user requests the object g, the GDS algorithm assigns key value $K(g)$ of object g as shown in Equation (2).

$$K(g) = L + \frac{C(g)}{S(g)} \quad (2)$$

where $C(g)$ is the cost of fetching object g from the server into the cache; $S(g)$ is the size of object g; and L is an aging factor. L in GDS starts at zero and is updated to the key value of the last replaced object. The key value $K(g)$ of object g is updated using the new L value since object g is accessed again. Thus, larger key values are assigned to objects that have been visited recently. If $C(g)$ is set to 1, it becomes GDS (1), and when the cost is set to the number of TCP packets needed to transmit object g, *i.e.*, $C(g) = 2 + S(g)/536$, it becomes GDS (packets).

Cao and Irani [15] proved that the GDS algorithm achieved better performance compared with some traditional caching algorithms. However, the GDS algorithm ignores the frequency of the Web object. Cherkasova [13] enhanced the GDS algorithm by integrating the frequency factor into the key value $K(g)$ as shown in Equation (3). The policy is known as Greedy-Dual-Size-Frequency (GDSF).

$$K(g) = L + F(g) * \frac{C(g)}{S(g)} \quad (3)$$

where $F(g)$ is the frequency of the visits of object g. Initially, when g is requested, $F(g)$ is initialized to 1. If g is in the cache, its frequency is increased by one. Similar to GDS, we then have GDSF (1) and GDSF (packets). Although GDSF achieves the best hit ratio, it yields a modest byte hit ratio [13].

Most Web proxy servers are still based on theses conventional replacement policies mentioned earlier for Web proxy cache management. These conventional Web caching methods form the basis of other caching algorithms [3] [11]. However, these conventional approaches still suffer from some limitations as shown in **Table 1**.

Table 1. Conventional Web cache replacement policies.

Policy	Brief description	Advantages	Disadvantages
LRU	The least recently used objects are removed first.	Simple and efficient with uniform size objects, such as the memory cache.	Ignores download latency and the size of Web objects
LFU	The least frequently used objects are removed first.	Simplicity	Ignores download latency and size of objects and may store obsolete Web objects indefinitely.
LFU-DA	Dynamic aging factor (L) is incorporated into LFU.	▪ Reduces cache pollution caused by LFU. ▪ High byte hit ratio	May suffer from hit ratio
SIZE	Big objects are removed first	Prefers keeping small Web objects in the cache, causing high cachet hit ratio.	▪ Stores small Web objects even if these object are never accessed again. ▪ Low byte hit ratio.
GDS	It assigns a key value to each cached object g as equation below. The object with the lowest key value is replaced first. $K(g) = L + \frac{C(g)}{S(g)}$ where $C(g)$ is the cost of fetching g from the server; $S(g)$ is the size of g; and L is an aging factor.	▪ Overcomes the weakness of SIZE policy by removing objects which are no longer requested by users. ▪ High hit ratio	▪ Does not take into account the previous frequency of Web objects. ▪ Low byte hit ratio.
GDSF	It extends GDS by integrating the frequency factor into the key value	▪ Takes into account the previous frequency of Web objects. ▪ Very high hit ratio	▪ Does not take into account the predicted accesses ▪ Low byte hit ratio.

3. Supervised Machine Learning

In supervised learning, the data (observations) are labeled with pre-defined classes. It is like that a teacher gives the classes. Support vector machine, naïve Bayes and decision tree are three popular supervised learning algorithms that perform classifications more accurately and faster than other algorithms in a wide range of applications [10].

3.1. Support Vector Machine

The support vector machine (SVM) is one of the most robust and accurate methods in all well-known machine learning algorithms. SVM was used successfully in a wide range of applications such as text classification, Web page classification and bioinformatics applications [16] [17].

Support vector machine (SVM) is a discriminative model that essentially works on two assumptions. First, SVM utilizes a small part of the training dataset called support vectors, which are close to the separating hyperplane and provide the most useful information for classification. Second, data is transformed into a high-dimension to use linear discriminate functions.

Consider a set of training data vectors $X = \{x_1, \cdots, x_L\}$, $x_i \in R^n$ and a set of corresponding labels $Y = \{y_1, \cdots, y_L\}$, $y_i \in \{1, -1\}$.

In SVM training, several kernel functions such as polynomial, sigmoid and RBF can be used to solve different classifications problems. However, RBF kernel function is the most often used kernel function and can achieve a better performance in many applications compared to other kernel functions.

In this study, RBF kernel function, expressed in Equation (4), is used in SVM training.

$$k\left(x_i, x_j\right) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right), \gamma > 0 \tag{4}$$

where the parameter γ represents the width of the RBF.

After training, the model can be saved to be used for later prediction. In future classification, the obtained SVM uses Equation (5) to predict the class of a Web object to each input vector x.

$$y\left(x\right) = sgn\left(\sum_{j=1}^{M} \alpha_j y_j K\left(x, x_j\right) + b\right) \tag{5}$$

where *M* is the number of support vectors, α_j is Lagrangian multipliers and $b \in R$.

3.2. Naïve Bayes

Although the Naïve Bayes (NB) classifier makes the restrictive assumption of conditional independence, several research works have shown that its classification accuracy is surprisingly strong. Moreover, the NB classifier has other interesting advantages. It is very efficient, easy to construct without parameters, and simple to be understood and interpreted [17] [18].

NB is very simple Bayesian network which has constrained structure of the graph [19]. In NB, all the attributes are assumed to be conditionally independent given the class label. The structure of the NB is illustrated in **Figure 1**.

In most of the data sets, the performance of the NB classifier is surprisingly good even if the independence assumption between attributes is unrealistic [19] [20]. Independence between the features ignores any correlation among them.

Let $A_1, A_2, \cdots, A_{/A/}$ be the set of attributes with discrete values in the data set D. Let *C* be the class attribute with |*C*| values, $c_1, c_2, \cdots, c_{/C/}$. Given a test example $d = \left\langle A_1 = a_1, \cdots, A_{/A/} = a_{/A/} \right\rangle$, where a_i is a possible value of A_i. In the training phase of NB classifier, it is only required to estimate the prior probabilities $\Pr\left(C = c_j\right)$ and the conditional probabilities $\Pr\left(A_i = a_i \middle| C = c_j\right)$ from the training dataset. Then, the NB classifier can predict class of a Web object using the Formula (6).

$$c = \arg\max_{c_j} \Pr\left(C = c_j\right) \prod_{i=1}^{/A/} \Pr\left(A_i = a_i \middle| C = c_j\right) \tag{6}$$

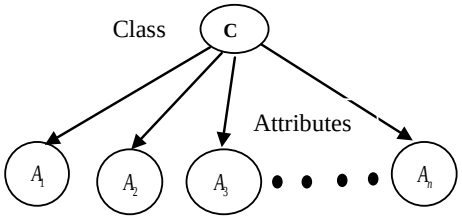

Figure 1. Structure of a naïve Bayes network.

3.3. Decision Tree

The decision tree is one of the most widely used and practical methods for inductive inference among others. The most well-know algorithm in the literature for building decision trees is the C4.5 decision tree algorithm, which was proposed by Quinlan (1993) [21]. In C4.5, the learned classifier is represented by a decision tree and can also be represented as sets of if-then rules to improve human readability. Therefore, the decision tree is simple to be understood and interpreted. Besides, it can handle nominal and categorical data and perform well with large dataset in a short time [22].

In the process of constructing the decision tree, the root node is first selected by evaluating each attribute on the basis of an impurity function to determine how well it alone classifies the training examples. The best attribute is selected and used to test at the root node of the tree. A descendant of the root node is created for each possible value of this selected attribute, and the training examples are sorted to the appropriate descendant node. The process is then repeated using the training examples associated with each descendant node to select the best attribute to test at that point in the tree.

4. Intelligent Dynamic-Aging Approaches

In Intelligent Dynamic-Aging Approaches, the supervised machine learning techniques are initially trained form Web proxy logs file. Then, the trained machine learning classifiers contribute to improvement of Web proxy caching in three stages: extraction of object's attributes, computation of object priority based on the trained classifiers and removal of objects with lowest priority.

The proxy logs files are considered a complete and prior knowledge of the users' interests and can be utilized as training data to effectively predict the next Web objects. The training dataset preparation step requires extracting the desired information from the logs proxy files, and then selecting the input/output dataset.

A training pattern takes the format $\langle x_1, x_2, x_3, x_4, x_5, x_6, y \rangle$. $x_1, \cdots, x_6$ represent the input features and y represents the target output of the requested object. **Table 2** shows the inputs and their meanings for each training pattern.

The value of y will be assigned to 1 if the object is re-requested again within the forward-looking sliding window. Otherwise, the target output will be assigned to 0. The idea is to use information about a Web object requested in the past to predict revisiting of such Web object within the forward-looking sliding window. In this study, sliding window is set to 30 minutes during the training datasets preparation.

The frequency factor is an important indicator for predicting the revisiting of Web objects in the future; however, other factors can also contribute in predicting the revisiting of object. In Intelligent Dynamic-Aging Approaches, the key value of object g is determined not just by its occurrence frequency in the past, but also by six factors: recency, frequency, sliding window-based frequency, retrieval time, size, and type of object. Therefore, the frequency is replaced by the probability of revisiting of object in the future as a proposed enhancement of the LFU-DA algorithm.

In Intelligent Dynamic-Aging Approaches, either the trained SVM, NB or C4.5 is combined with DA to produce novel intelligent Web proxy caching approaches for making cache replacement decisions. The proposed intelligent cache replacement approaches are known as SVM-DA, NB-DA and C4.5-DA.

In SVM-DA, NB-DA and C4.5-DA, SVM, NB and C4.5 are trained using Web proxy logs files to predict the class of objects to be re-visited or not. Then, the trained SVM, NB and C4.5 can combine the most significant factors, such as the recency, frequency, SWL-based frequency, retrieval time, size, and type of object, for predicting the probability that Web objects would be re-visited.

Table 2. The meaning of inputs in the training pattern.

Input	Meaning
x_1	Recency of Web object access based on backward-looking sliding window
x_2	Frequency of Web object accesses
x_3	Frequency of Web object accesses based on backward-looking sliding window
x_4	Retrieval time of Web object in milliseconds
x_5	Size of Web object in bytes
x_6	Type of Web object

In other words, when a user visits Web object *g*, the SVM, NB or C4.5 classifier can predict the probability *Pr*(*g*) of belonging object *g* to the class of objects that would be revisited. Then, the probabilities of *g* are accumulated as scores, $W(g) = \sum Pr(g)$, and used in cache replacement decision as show in Equation (7).

$$K(g) = L + W(g) \tag{7}$$

where *L* is a dynamic aging factor, which is initialized to zero, and then updated to the key value of the last removed object.

The object with the lowest scores is a better choice for replacement. In the proposed SVM-DA, NB-DA and C4.5-DA approaches, *L* is useful to prevent the cache pollution and improve the performance in implementation for longer periods of time. **Figure 2** explains the algorithm of intelligent DA approaches.

5. Results and Discussion

In this study, the proxy datasets or the proxy logs files were obtained from five proxy servers located in the United States from the IRCache network and covered fifteen days [23]. The proxy logs files of 21st August, 2010 were used in the training phase, while the proxy logs files of the following two weeks were used in the simulation.

The machine learning techniques were trained using WEKA. The SVM model was trained using the libsvm library [24]. The generalization capability of SVM is controlled through a few parameters such as the term *C*(*C* is a positive regularization constant, which controls allowable errors in the trained solution) and the kernel parameter like RBF width γ. To decide which values to choose for parameter *C* and γ, a grid search algorithm was implemented using MATLAB as suggested by Hsu *et al*. [25]. The parameters that obtain the best accuracy using a 10-fold cross validation on the training dataset were retained. Next, a SVM model was trained using WEKA depending on the optimal parameters to predict and classify the Web objects whether the objects would be re-visited or not. Regarding C4.5 training, a J48 learning algorithm was used, which is a Java re-implementation of C4.5 and provided with WEKA tool. The default values of parameters and settings were used as determined in WEKA. In NB training, the datasets were discretized using a MDL method suggested by Fayyad and Irani [26] with default setup in WEKA. Once the training dataset was prepared and discretized, NB was trained using WEKA as well.

In order to evaluate the proposed Intelligent Dynamic-Aging Approaches, the trace-driven simulator WebTraff [27] was modified to meet the proposed caching approaches. The trained classifiers were integrated with WebTraff to simulate the proposed intelligent Web proxy cache replacement approaches. The WebTraff simulator receives the prepared log proxy file as input and generates files containing performance measures as outputs. In addition, the maximum cache size should be determined in the simulator. The simulator starts automatically with a cache size of 1 MB, and scales it up by a factor of two for each run until the maximum desired cache size is reached. The output reported by the simulator shows cache size (in MB), hit ratio, and byte hit ratio for each cache size simulated.

In WebTraff simulation, an infinite cache is required to determine the maximum cache size in the simulator, which represents the stop point during the simulation. The infinite cache is a cache with enough space to store all requested objects without the need to replace any object. The infinite cache size is defined as the total size of

```
Begin

Initialize L = 0;
 For each Web object g requested by user
        Begin
         If g in cache
Begin
                    Cache hit occurs
                    Update information of g
                  // compute probability Pr(g) that object g will be revisited
                   // in the future depending on SVM, NB or C4.5 classifier
                  Pr(g) = classifier.compute_ probability( common features)

                  // update priority of g based on SVM, NB or C4.5
                  W(g) = W_old(g) + Pr(g)  such that initial W(g) = 0
                  K(g) = L + W(g)
           End

            Else
           Begin
                Cache miss occurs
                While no enough space in cache buffer for g
            Begin
                L = min(K(q)), for each q in cache
                                Evict q such that K(q) = L
              End
                Fetch g into cache from origin server.
           End
End

End
```

Figure 2. The algorithm of intelligent DA approaches.

all unique requests. Therefore, it is unnecessary to consider any replacement policy for an infinite cache. In an infinite cache, the hit ratio (HR) and byte hit ratio (BHR) reach their maximum values.

HR and BHR are the most widely used metrics in evaluating the performance of Web caching [1]-[8]. HR is defined as the percentage of requests that can be satisfied by the cache. Note that HR only indicates how many requests hit and does not indicate how much bandwidth can be saved. BHR is the number of bytes satisfied from the cache as a fraction of the total bytes requested by the user. BHR represents the saved network bandwidth achieved by caching approach.

In order to clarify the benefits of the proposed Intelligent Dynamic-Aging Approaches, the improvement ratio (*IR*) of the performance in terms of HR and BHR for each particular cache size can be calculated using Equation (8).

$$IR = \frac{(PM - CM)}{CM} \times 100\,(\%) \tag{8}$$

where *IR* is the percent of improvement in terms of HR or BHR achieved by the proposed method (*PM*), Intelligent Dynamic-Aging method, over the conventional method (CM) which is LFU-DA.

As can be observed from **Table 3**, as the cache size increased, the performance improvement of the proposed intelligent SVM-DA, NB-DA and C4.5-DA over LFU-DA increased for the five proxy datasets. However, the performance of the proposed intelligent SVM-DA, NB-DA and C4.5-DA were roughly the same as the conventional LFU-DA at high levels of the proxy cache capacity. The intuition behind this is that when the cache size was close to the size of the infinite cache, the performance became stable and close to its maximum level. On the contrary, when the cache size was small, the objects replacement was frequently required. Hence, the effect of the performance of replacement policy was clearly noted.

Table 3 presents the average *IRs* achieved by SVM-DA, NB-DA and C4.5-DA over the conventional LFU-DA algorithm for the five proxy datasets in each particular cache. The simulation results in **Table 3** show that the proposed SVM-DA, NB-DA and C4.5-DA significantly improved the performance of the conventional LFU-DA approach. The average *IRs* of HR achieved by SVM-DA, NB-DA and C4.5-DA over LFU-DA in-

Table 3. The average IRs (%) achieved by intelligent DA approaches over conventional LFU-DA.

Cache size (MB)	Average IR of SVM-DA over LFU-DA (%)		Average IR of NB-DA over LFU-DA (%)		Average IR of C4.5-DA over LFU-DA (%)	
	HR	BHR	HR	BHR	HR	BHR
1	13.47	10.72	35.458	19.536	16.088	12.81
2	13.344	8.266	25.776	41.184	15.388	10.694
4	11.234	30.37	19.948	69.074	15.88	26.312
8	14.85	5.61	16.9	21.932	14.916	8.232
16	12.192	4.666	12.512	9.262	12.098	5.768
32	11.328	4.514	9.104	4.558	10.946	4.912
64	10.036	7.854	9.806	11.814	10.276	9.056
128	8.328	6.02	4.358	−1.934	7.712	2.316
256	9.49	0.674	3.848	−1.276	8.172	0.766
512	8.422	0.732	2.036	−1.158	7.348	0.45
1024	6.058	0.308	−0.422	−0.436	5.018	0.102
2048	5.232	−0.034	−0.964	−1.084	3.876	−0.552
4096	2.91	−0.108	−3.04	−0.894	1.5	−0.19
8192	1.858	−0.474	−2.828	−0.498	0.838	−0.286
16,384	0.894	−0.484	−1.658	−0.146	0.788	0.232
32,768	0.706	0.028	−1.148	0.038	0.61	0.118

creased by 14.85%, 35.458% and 16.088% respectively. In terms of BHR, the superior BHR of the conventional LFU-DA was maintained using the proposed SVM-DA, NB-DA and C4.5-DA approaches in the worst case. In the best case, the average IRs achieved by SVM-DA, NB-DA and C4.5-DA over LFU-DA increased by 30.37%, 69.074% and 26.312%, respectively. This was due to the capability of the SVM, NB or C4.5 classifier used with the proposed approaches to accurately predict the probability of objects belonging to the class of objects that would be revisited. On the other hand, LFU-DA worked to the advantage of some large objects under BHR. In other words, LFU-DA gave some larger objects, which recently requested, the same chance to stay in the cache as their smaller counterparts.

When the proposed intelligent DA approaches were compared with each other, the HRs of the proposed SVM-DA, NB-DA and C4.5-DA approaches were comparable when the cache size was small, while HR of SVM-DA was better compared to the HRs of NB-DA and C4.5-DA with large cache sizes. On the other hand, the BHR of NB-DA was better than BHR of SVM-DA and C4.5-DA with small caches, while the BHRs of SVM-DA and C4.5-DA was better compared to BHR of NB-DA when the cache size was equal or greater than 128 MB. Thus, the performance of SVM-DA was more stable and provided a better balance between HR and BHR when compared to other algorithms across the different cache sizes.

6. Conclusion

LFU-DA policy is a conventional Web caching method, which attempts to remove the cache pollution in a LFU policy. However, the performance of LFU-DA in terms of hit ratio is not efficient enough. In this paper, novel intelligent dynamic aging approaches based on SVM, NB and C4.5, which are known as SVM-DA, NB-DA and C4.5-DA, were suggested for making cache replacement decisions with better hit ratio. As an alternative to the frequency factor in LFU-DA, the proposed SVM-DA, NB-DA and C4.5-DA approaches combined the most significant factors depending on intelligent classifier for predicting probability that Web objects would be re-visited later. The simulation results revealed that the performance of the conventional LFU-DA was significantly

enhanced by using the proposed intelligent SVM-DA, NB-DA and C4.5-DA approaches. This was due to the capability of the SVM-DA, NB-DA and C4.5-DA approaches to accurately predict the probability of objects belonging to the class of objects that would be revisited. The average *IRs* of HR achieved by SVM-DA, NB-DA and C4.5-DA over LFU-DA increased by 14.85%, 35.458% and 16.088%, respectively, while the average IRs of BHR increased by 30.37%, 69.074% and 26.31%, respectively. In the worst case, the proposed SVM-DA, NB-DA and C4.5-DA approaches retained the superior BHR of the conventional LFU-DA. Lastly, it also was observed from the simulation results that SVM-DA achieved the best BHR among all algorithms.

References

[1] Koskela, T., Heikkonen, J. and Kaski, K. (2003) Web Cache Optimization with Nonlinear Model Using Object Features. *Computer Networks*, **43**, 805-817. http://dx.doi.org/10.1016/S1389-1286(03)00334-7

[2] Romano, S. and ElAarag, H. (2011) A Neural Network Proxy Cache Replacement Strategy and Its Implementation in the Squid Proxy Server. *Neural Computing & Applications*, **20**, 59-78. http://dx.doi.org/10.1007/s00521-010-0442-0

[3] Cobb, J. and ElAarag, H. (2008) Web Proxy Cache Replacement Scheme Based on Back-Propagation Neural Network. *Journal of Systems and Software*, **81**, 1539-1558. http://dx.doi.org/10.1016/j.jss.2007.10.024

[4] Foong, A.P., Yu-Hen, H. and Heisey, D.M. (1999) Logistic Regression in an Adaptive Web Cache. *Internet Computing*, **3**, 27-36. http://dx.doi.org/10.1109/4236.793455

[5] Sajeev, G. and Sebastian, M. (2011) A Novel Content Classification Scheme for Web Caches. *Evolving Systems*, **2**, 101-118. http://dx.doi.org/10.1007/s12530-010-9026-6

[6] Ali Ahmed, W. and Shamsuddin, S.M. (2011) Neuro-Fuzzy System in Partitioned Client-Side Web Cache. *Expert Systems with Applications*, **38**, 14715-14725. http://dx.doi.org/10.1016/j.eswa.2011.05.009

[7] Ali, W., Sulaiman, S. and Ahmad, N. (2014) Performance Improvement of Least-Recently-Used Policy in Web Proxy Cache Replacement Using Supervised Machine Learning. *International Journal of Advances in Soft Computing & Its Applications*, **6**, 1-38.

[8] Yasin, W., Ibrahim, H., Udzir, N.I. and Hamid, N.A.W.A. (2014) Intelligent Cooperative Least Recently Used Web Caching Policy based on J48 Classifier. 16*th International Conference on Information Integration and Web-Based Applications & Services*, 262-269. http://dx.doi.org/10.1145/2684200.2684299

[9] Ali, W., Shamsuddin, S.M. and Ismail, A.S. (2011) A Survey of Web Caching and Prefetching. *International Journal of Advances in Soft Computing & Its Applications*, **3**, 18-44.

[10] Wu, X., Kumar, V. and Ross Quinlan, J. (2008) Top 10 Algorithms in Data Mining. *Knowledge and Information Systems*, **14**, 1-37. http://dx.doi.org/10.1007/s10115-007-0114-2

[11] ElAarag, H. (2013) A Quantitative Study of Web Cache Replacement Strategies Using Simulation. In: ElAarag, H., Ed., *Web Proxy Cache Replacement Strategies*, Springer, London, 17-60. http://dx.doi.org/10.1007/978-1-4471-4893-7_4

[12] Arlitt, M., Friedrich, R. and Jin, T. (2000) Performance Evaluation of Web Proxy Cache Replacement Policies. *Performance Evaluation*, **39**, 149-164. http://dx.doi.org/10.1016/S0166-5316(99)00062-0

[13] Cherkasova, L. (1998) Improving WWW Proxies Performance with Greedy-Dual-Size-Frequency Caching Policy. HP Technical Report, Palo Alto.

[14] Arlitt, M., Cherkasova, L., Dilley, J., Friedrich, R. and Jin, T. (2000) Evaluating Content Management Techniques for Web Proxy Caches. *ACM SIGMETRICS Performance Evaluation Review*, **27**, 3-11. http://dx.doi.org/10.1145/346000.346003

[15] Cao, P. and Irani, S. (1997) Cost-Aware WWW Proxy Caching Algorithms. *Proceedings of the* 1997 *Usenix Symposium on Internet Technology and Systems*, Monterey, 8-11 December 1997.

[16] Chen, R.-C. and Hsieh, C.-H. (2006)Web Page Classification Based on a Support Vector Machine Using a Weighted Vote Schema. *Expert Systems with Applications*, **31**, 427-435. http://dx.doi.org/10.1016/j.eswa.2005.09.079

[17] Lu, S.H., Chiang, D.A., Keh, H.C. and Huang, H.H. (2010) Chinese Text Classification by the Naïve Bayes Classifier and the Associative Classifier with Multiple Confidence Threshold Values. *Knowledge-Based Systems*, **23**, 598-604. http://dx.doi.org/10.1016/j.knosys.2010.04.004

[18] Hall, M. (2007) A Decision Tree-Based Attribute Weighting Filter for Naive Bayes. *Knowledge-Based Systems*, **20**, 120-126. http://dx.doi.org/10.1016/j.knosys.2006.11.008

[19] Friedman, N., Geiger, D. and Goldszmidt, M. (1997) Bayesian Network Classifiers. *Machine Learning*, **29**, 131-163. http://dx.doi.org/10.1023/A:1007465528199

[20] Pernkopf, F. (2005) Bayesian Network Classifiers versus Selective K-NN Classifier. *Pattern Recognition*, **38**, 1-10.

http://dx.doi.org/10.1016/j.patcog.2004.05.012

[21] Quinlan, J.R. (1993) C4.5: Programs for Machine Learning. Morgan Kaufmann, Burlington.

[22] Huang, C.-J., Wang, Y.-W., Huang, T.-H., Lin, C.-F., Li, C.-Y. and Chen, H.-M. (2011) Applications of Machine Learning Techniques to a Sensor-Network-Based Prosthesis Training System. *Applied Soft Computing*, **11**, 3229-3237. http://dx.doi.org/10.1016/j.asoc.2010.12.025

[23] NLANR (2010) National Lab of Applied Network Research (NLANR). Sanitized Access Logs. http://www.ircache.net/

[24] Chang, C.C. and Lin, C.J. (2001) LIBSVM: A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm

[25] Hsu, C.W., Chang, C.C. and Lin, C.J. (2009) A Practical Guide to Support Vector Classification. http://www.csie.ntu.edu.tw/~cjlin/libsvm

[26] Fayyad, U.M. and Irani, K.B. (1993) Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *Proceedings of the* 13*th International Joint Conference on Artificial Intelligence* (*IJCAI*-93), Chambèry, 28 August-3 September 1993, 1022-1027.

[27] Markatchev, N. and Williamson, C. (2002) WebTraff: A GUI for Web Proxy Cache Workload Modeling and Analysis. In: *Proceedings of the* 10*th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*, IEEE Computer Society, 356-363. http://dx.doi.org/10.1109/MASCOT.2002.1167096