Indonesian Journal of Electrical Engineering and Informatics (IJEEI)
Vol. 5, No. 4, December 2017, pp. 357~365
ISSN: 2089-3272, DOI: 10.11591/ijeei.v5i4.362

Intelligent Cooperative Adaptive Weight Ranking Policy via dynamic aging based on NB and J48 classifiers

Dua'a Mahmoud Al-Qudah*, Rashidah Funke Olanrewaju, Amelia Wong Azman
Department of Electrical and Computer Engineering, Kulliyyah of Engineering
International Islamic University Malaysia
e-mail: dm_q66@yahoo.com* , frashidah@iium.edu.my , amy@iium.edu.my

Abstract

The increased usage of World Wide Web leads to increase in network traffic and create a bottleneck over the internet performance. For most people, the accessing speed or the response time is the most critical factor when using the internet. Reducing response time was done by using web proxy cache technique that storing a copy of pages between client and server sides. If requested pages are cached in the proxy, there is no need to access the server. But, the cache size is limited, so cache replacement algorithms are used to remove pages from the cache when it is full. On the other hand, the conventional algorithms for replacement such as Least Recently Use (LRU), First in First Out (FIFO), Least Frequently Use (LFU), Randomised Policy, etc. may discard essential pages just before use. Furthermore, using conventional algorithms cannot be well optimized since it requires some decision to evict intelligently before a page is replaced. Hence, this paper proposes an integration of Adaptive Weight Ranking Policy (AWRP) with intelligent classifiers (NB-AWRP-DA and J48-AWRP-DA) via dynamic aging factor. To enhance classifiers power of prediction before integrating them with AWRP, particle swarm optimization (PSO) automated wrapper feature selection methods are used to choose the best subset of features that are relevant and influence classifiers prediction accuracy. Experimental Result shows that NB-AWRP-DA enhances the performance of web proxy cache across multi proxy datasets by 4.008%,4.087% and 14.022% over LRU, LFU, and FIFO while, J48-AWRP-DA increases HR by 0.483%, 0.563% and 10.497% over LRU, LFU, and FIFO respectively. Meanwhile, BHR of NB-AWRP-DA rises by 0.9911%,1.008% and 11.5842% over LRU, LFU, and FIFO respectively while 0.0204%, 0.0379% and 10.6136 for LRU, LFU, FIFO respectively using J48-AWRP-DA.

Keywords: *PSO,NB,J48, cache replacement algorithms, AWRP, dynamic aging*

1. Introduction

The extensive usage of internet leads to increase in network traffic and page access latency [1, 2]. By having web cache, web objects can be kept closer to the client. A web proxy cache saves web object that can be revisited soon closer to the user which leads to decrease of the number of web object requested from the server, hence, reducing the volume of data that is transmitted over the network as well as the delay while accessing the web page [3, 4]. Consequently, the overall performance of web system can be improved. These web objects are commonly placed in three locations; 1) browser caching, 2) proxy caching and; 3) server-side proxy caching [5][20]).

Three significant factors are the pivotal roles in cache performance. They are cache replacement algorithms, cache consistency and cache algorithm (passive or active). Cache replacement algorithm is the process of evicting pages from cache to make room for a newly requested page. This replacement procedure has a significant effect in the web caching[6], and they classified into five categories, namely: frequency based, recency-based, frequency/recency-based, function-based and randomized based [7]. Though the most common algorithms are Least Recently Used(LRU), Least Frequently Used(LFU), Size and Greedy Dual SIZE(GDS), know as conventional algorithms. However, they are not suitable for web caching because the algorithms consider one or two features while ignoring the others. Conventional algorithms also suffer from problems like cache pollution in which the cache saved objects that are not requested frequently. There are many algorithms that combine two or more features to enhance the performance of the replacement algorithms. However, the selection of features is not a straightforward task because an essential factor in one environment may be not crucial in another environment[3].

Received Aug 11, 2017; Revised Oct 23, 2017; Accepted Nov 11, 2017

Adaptive Weight Ranking Policy(AWRP) attempts to give weight to each page depending on three factors: recency, frequency and total number of access to be made. This overcomes the problem of traditional algorithms such as least recently used LRU, First Input First Output FIFO and Clock Adaptive Replacement CAR [8, 9]. Intelligent replacement algorithms were proposed to overcome problems of conventional replacement algorithms and to enhance the performance of web cache. The intelligent based algorithm predicts which web objects will be revisited in the future or not. Hence, machine learning algorithms such as Fuzzy, Artificial Neural Network, Adaptive Neuro-Fuzzy, Naïve Bayes, Decision Tree, and Support Vector Machine have been used for the prediction. Sucht can be combined with one or more conventional algorithms to improve the performance, mainly the hit ratio and byte hit ratio[10].

2. Related Works

Conventional cache replacement algorithms are the most used algorithms. The algorithms apply to CPU caches and virtual memory system but, they are not efficient in web caching. In [7, 10, 11], authors explained the most common algorithms which are: Least Recently Used(LRU), Least Frequently Used (LFU), SIZE, Greedy-Dual-Size (GDS) and Greedy-Dual-Size-Frequency (GDSF).

Debabala Swain [8] have used an alternative, an adaptive replacement algorithm that ranks objects in the cache based on weight as defined in (1):

$$Wi = Fi/(N-Ri) \quad \ldots\ldots \ (1)$$

Where Fi is a frequency index, Ri is recency index, and N is the total number of access to be made. It behaves like LRU and LFU algorithms. However, the performance is better than LRU, LFU, and CAR based on hit ratio, Although it assumes all blocks have equal size and it adds time and space overhead.

ElAarag [7] analyzed twenty-seven proxy cache replacement algorithms based on HR, BHR and object removal rate. The author concluded that most of the algorithms are simple and utilizes a relatively low CPU and space overhead. The PSS, CSS, M-Metric, MIX and GDSF algorithms gave system administrators greater control over their proxy servers.

Khaleel, Osman [12] proposed a new multi-agent replacement algorithm called Average Least Frequency Used Removal (ALFUR). It is consist of multi-agents namely: Reader agent, Analyzer agent, Removal agent and performance agent. All agents are incorporated together to evict the object that has the smallest frequency, size and oldest web object in the cache. ALFUR algorithm is compared to LFU, LRU, SIZE and PCCIA algorithms. The result indicated that using ALFUR enhance web cache performance regarding HR and BHR with 84.12%, 53.9% respectively when the cache size is 500MB.

Ahmed and Shamsuddin [13] developed an intelligent client-side web caching scheme (ICWCS) which depended on splitting client cache side into two caches: short-term cache which is managed using LRU algorithm, and long-term cache which is accomplished using a neuro-fuzzy system. The performance of BFR and HR of web caching was improved over LRU and LFU algorithms. Though, the performance regarding byte hit ratio was not good enough due to not taking into account the cost and size of the predicted objects in cache replacement process. Moreover, the training process required longer time and extra computational overheads.

NB-GDS, NB-LRU, and NB-DA approaches were developed by Ali, Shamsuddin [3] which combined between Greedy-Dual Size (GDS), LRU and DA conventional algorithms and Naïve Bayes classifier. These approaches were made in two stages: offline stage to train NB classifier and predict either web object will be re-visited or not, and online stage which corporate NB and conventional algorithms to evict pages when the proxy cache is full. Though algorithm performed better than BPNN and ANFIS regarding HR and BHR, however, NB adds additional computational overhead needed for target output preparation.

Kumar and Reddy [14] proposed NB-SIZE and C4.5-Hybrid; integration between conventional SIZE, Hybrid algorithms, Naïve Bayes and decision tree (C4.5) to significantly improve pure HR, BHR, and latency.

Another hybrid of LFU-DA and machine learning technique such as SVM, NB, and C4.5 proposed by Ali and Shamsuddin [15] to enhance the performance of HR and BHR. Though it works well, however, features are selected manually in training machine learning techniques.

Ibrahim, Yasin [16] proposed co-operation between J48 classifier and LRU to improve the performance of LRU and avoid cache pollution problem. The simulation results show that intelligent algorithms performance is better than LRU. Authors in [9] proposed a method using IWSS embedded NB feature selection method to extract the best subset feature for Naïve Bayes classifier. The Naïve Bayes classifier was later integrated with Adaptive Weight Ranking Policy via Dynamic Aging (AWRP-DA) to improve the performance of web Hit Ratio (HR) and Byte Hit Ratio (BHR).

3. Proposed Model

The proposed model is composed of several stages like other intelligent models. It splits the dataset into training/testing sets. Preprocessing, feature extraction, feature selection and reduction are executed on dataset before training model and testing it. The last phase is evaluating intelligent techniques that are used and evaluating new replacement algorithms. This model is built based on integration of intelligent techniques Naïve Bayes (NB) and C4.5 decision tree (J48) with Adaptive Weight Ranking Policy via Dynamic aging factor (AWRP-DA) to enhance web system performance.

3.1. Raw Data Collection

Whenever the user requests a page, the proxy server stores the event into a log file. Log files have information about users' behaviors and requests. Proxy log files can be gathered from different proxies that placed inside organization or universities. Availability of log files was one of the most important motivations to incorporate machine learning with conventional replacement algorithms [7]. Proxy log files that are used in this research are provided by IRCache network. IRCache trace files are gathered from different proxy servers that are placed around the United States. That is; BO2, PA, SJ, SV, UC and SD servers located at Boulder Colorado, Palo Alto California, MAE-West San Jose California, Silicon Valley California (FIX-West), Urbana-Champaign, Illinois San Diego California, between 9 and 10 of January 2007. Standard features exist in most trace files namely: time elapsed, timestamp, object size, client address, log tag with HTTP script, URL, user identification, request method, content type and hierarchy of data and hostname.

3.2. Data Pre-Processing

Microsoft Visual C# 2017 was used to designed program that read proxy log files and does pre-process. The pre-processing steps from parsing, filtering and finalizing followed to remove unwanted features and simplifying others.

3.3. Feature Extraction

After getting pre-processed log files extracting features process apply for them to obtain new features that have a significant impact on classifies performance. The log files will contain URL_ID, Timestamp, Elapsed time, Size, Type, Frequency, SWL_frequency, IR_frequency, Recency, Time spend and Mean after the finished feature extraction process. Detailed of extraction in [17].

3.4. Feature Selection

After finalizing data pre-processing and feature extraction. Weka 3.8 is used to select the best subset of features. WrapperSubsetEval is used as attribute evaluator in all data and PSO search method is used for both classifier, NB and J48 with default Weka attributes for rest parameters. The following table 1 presents the best subset of features that are chosen by PSO method with each classifier.

Table 1. the best subset of features that are chosen by PSO method with each classifier

	BO2	PA	SJ	SV	UC	SD
NB	Freq, FIR	Id, Timestamp, Type, Freq , FIR, Recency, Mean	Id, Timestamp , FIR, Swl_Freq, Timespend , Recency, Mean	Id, Type, Freq, FIR, Swl_Freq, Recency	Id, Elapsedtime, Size,Type, Freq, FIR, Swl_Freq, Recency, Mean	Id ,Elapsedtime, Size, Type, Freq, Recency, Mean
#of features	2	7	7	6	9	7
J48	Id, Timestamp, Type, Freq , Fir, Recency, Mean	Id, Timestamp, Type, Freq , Fir, Recency, Timespend	Id, Timestamp , Type, Freq , Fir, Recency, Mean, Timespend	Id, Timestamp , Type, Freq, Fir, , Recency	Id, Timestamp, Eapsedtime, Type, Freq, Fir, Timespend , Recency, Mean	Id, Timestamp, Elapsedtime, Type, Freq, Fir, Swl_Freq, Recency
#of Features	7	7	8	6	9	8

3.2. Classifier Training

For preparing data to train each proxy datasets are divided into 70% for training and 30% for testing. Then the data are discretized by using Minimum Description Length (MDL) method that suggested by [18] with default setup in WEKA. The input features are set as features set regarding < X1,….Xn>. The output is defined as one if the page requested again within forwarding sliding window and zero if not requested again. Finally set input/output form as <X1,…..Xn, Y> which any classifier can be trained using previous input/output pattern. Naïve Bayes and J48 classifiers are training and testing using features that are extracted using Practical swarm optimization method. The trained classifiers are saved to integrate with proposed web proxy cache replacement algorithm.

3.3. Intelligent AWRP-DA Algorithm

Intelligent replacement algorithm is integration of intelligent system with Adaptive Weight Ranking Policy (AWRP). AWRP ranking web objects by calculate weight for each of them depending on three factors as shown in equation 2.

$$Wi = \frac{Fi}{(N - Ri)} \dots\dots (2)$$

Where Ri represent recency counter of object i, Fi represents frequency counter of object i, and N is a total number of access to be made. If new object k is stored in the buffer then object's parameters must be set as follow: Rk = clock which means that object k recently used, Fk = 1 which means that object k is used once and Wk = 0. Now, if object j requested and it finds inside buffer a hit occur. Thus, object parameters must be updated in following way:

$$Fi = Fi + 1 \quad \textit{for every i.}$$

Ri = corresponding clock access value.

If requested object i is not inside buffer a miss occur. Fetching object to cache is required. If cache is full then object with smallest weighting value will be evicted until sufficient space of new object. Thus, evaluation for each object will be done as follow:

a. Wi evaluated for each block I for N ≠ Ri.
b. Removing page i with lowest weight index.
c. Weighting value for each object inside buffer updated only when misses occur that reduces overhead in the system.

Intelligent systems (NB, J48) integrated with AWRP as follow:
For NB:

a. Pri = Apply NB_classifier (common features).
b. $\mathrm{Wi} = \frac{\mathrm{Fi}}{(\mathrm{N-Ri})}$
c. $\mathrm{Gi} = (\mathrm{Wi} * \mathrm{Pri}) + \mathrm{L}$

L is a dynamic aging factor. Initially L is set to 0, but upon the removal of an object i, L is set to Gi (weight of last removed object). Pri represent probability of object to revisited or not and Gi is final weight value for object i. Figure 1 illustrates intelligent NB-AWRP-DA pseudo code.

```
For each object g requested by user
Initialize L =0
Begin
        If g in cache
                Begin
                      Cache hit occur
                      Update g parameters
                      $F_i = F_i+1$
                     $R_i$= corresponding clock access value
                     $Pr_i$ = apply_NB_classifier (common features).
                End
    Else
               Begin
            Cache miss occur
                     For each object in cache
                     $W_i = F_i / (N-R_i)$
                     $G_i = W_i * Pr_i + L$
          Fetch object from original server
           While no enough space in cache
                     Begin
                     L = min (G (q)) for each q in cache
                     Evict q such that G(q) = L
                     End
             End
End
```

Figure 1. Codefragment 1 Intelligent NB-AWRP-DA pseudo code

For J48:

- Ci = Apply J48_classifier (common features).
- $Wi = \frac{Fi}{(N-Ri)}$
- $Gi = (Wi) + L$

L is a dynamic aging factor. Initially L is set to 0, but upon the removal of an object i, L is set to Gi (weight of last removed object). Ci represent class of object to revisited or not and Gi is final weight value for object i. Figure 2 illustrates intelligent J48-AWRP-DA pseudo code.

To evaluate the proposed Intelligent AWRP-DA, the trace-driven simulator is used by building one with Microsoft visual C# 2017 to simulate and evaluate proposed approach and compare with traditional web cache algorithms.

3.4. Web Proxy Cache Simulation

A trace-driven simulator build using visual C# 2017 in help with Windows Web Proxy Caching Simulation (WWPCS) simulator designed by [19]. It is designed to evaluate proposed intelligent algorithms NB-AWRP-DA and J48-IAWRP-DA and evaluate most common traditional replacement algorithms LRU, LFU and FIFO.

The simulator uses processed proxy dataset as input and create two output files contain HR and BHR for each algorithm. To run the simulator infinite cache size in GB must be set and determining web cache algorithm that will be evaluated. Cache size starts from 1 MB and rising according to a base 2 logarithmic scale until infinite cache size determined by user is reached.

```
For each object g requested by user
Initialize L =0
Begin
    If g in cache
        Begin
            Cache hit occur
            Update g parameters
            $F_i = F_i + 1$
            $R_i$= corresponding clock access value
            $c_i$ = apply_J48_classifier (common features).
        End
    Else
        Begin
        Cache miss occur
            For each object in cache
            $W_i = F_i / (N - R_i)$
            $G_i = W_i + L$
        Fetch object from original server
        While no enough space in cache
            Begin
            L = min (G (q)) for each q in cache
            Evict q such that G(q) = L and Ci != 1
            End
        End
End
```

Figure 2. Intelligent J48-AWRP-DA pseudo code

4. Evaluation

4.1. Web Proxy Cache Algorithms Evaluation

In this study, two measures are used that are Hit Rate (HR) and Byte Hit Rate (BHR), this are the most effective and popular measures. HR is defined as the ratio of a total number of hit pages over all requested pages by clients. Where BHR is defined as the ratio of bytes served by the cache over the total number of bytes requested by the clients. BHR can be significantly different from HR in a case where only a few, but large files are being served by the cache. HR and BHR defined in (3) and (4) respectively where N is a total number of requests, δi =1 if request i in the cache else δi=0, and bi is the size of the ith request. However, HR and BHR work oppositely [14].

$$HR = \frac{\sum \delta i}{N} \ldots\ldots \quad (3)$$

$$BHR = \frac{\sum (\delta i * bi)}{\sum bi} \ldots\ldots (4)$$

4.1.1. Performance Measures of Infinite Cache

An infinite cache; it is a cache that can hold all pages requested without needing to remove any page. Thus, there is no need to take in consider any replacement algorithms used because of replacement algorithms required when the cache is full. The maximum value of HR and BHR are reached in an infinite cache. In fact, the infinite cache cannot be designed. Thus, replacement algorithms required when the cache becomes full. In this research, infinite cache size is considered 32GB.

4.1.2. Impact of Cache Size on Performance Measures

The proposed algorithms NB-IAWRP_DA and J48-IAWRP_DA are compared with LRU, LFU and FIFO the most popular traditional algorithms. All policies are evaluated by simulating the cache size start from 1MB and scaling up to 32GB. Figure 3 present impact of cache size on HR for BO2, PA, SJ, SV and UC proxy datasets respectively by comparing HR values of each proxy datasets with varying cache sizes. While Figure 4 compared among replacement algorithms in term of BHR over BO2, PA, SJ, SV and UC proxy datasets respectively. It is obvious from HR and BHR graphs that increasing cache size cause rising in HR and BHR for all proxy datasets. However, increasing percentage becomes smaller when cache size increases and HR and BHR values tend to be stable and adjacent to their maximum value. On contrast, replacement algorithms are required repeatedly when cache size is small. Since the cache being full rapidly. Thus, real performance of replacement algorithms evidence clearly when cache size not very small and not so big. The best range that indicates realistic performance is between 32 MB and 32 GB.

Figure 3 shows HR values for BO2, PA, SJ, SV and UC indicate that proposed NB-AWRP-DA and J48-AWRP-DA algorithms achieve the highest HR value over traditional algorithms (LRU, LFU and FIFO). This is expected since the proposed algorithms consider recency, frequency features and dynamic aging factors in contrast to traditional algorithms which take in account one feature at a time. In addition, integrating NB and J48 trained classifiers with proposed algorithms enhance the performance of them efficiently.

NB-AWRP-DA performance is better than J48-AWRP-DA since NB-AWRP-DA algorithm depends on the probability of requested the page again or not. While J48-AWRP-DA depends on the class of page if is expected to request again will take priority and another chance to stay in the cache. Since priority is much precise than expecting class of page the NB-AWRP-DA give higher HR than J48-AWRP-DA algorithm.

FIFO classifier achieve the lowest HR value among all which is expected since it does not take in account any features to select evicted page. Moreover, LFU performs better than LRU in most proxy files. Proxy files HR range varies between high in some files like SJ and lower in others like PA. This happens because variety in total number of requested pages and how much the page frequently requested.

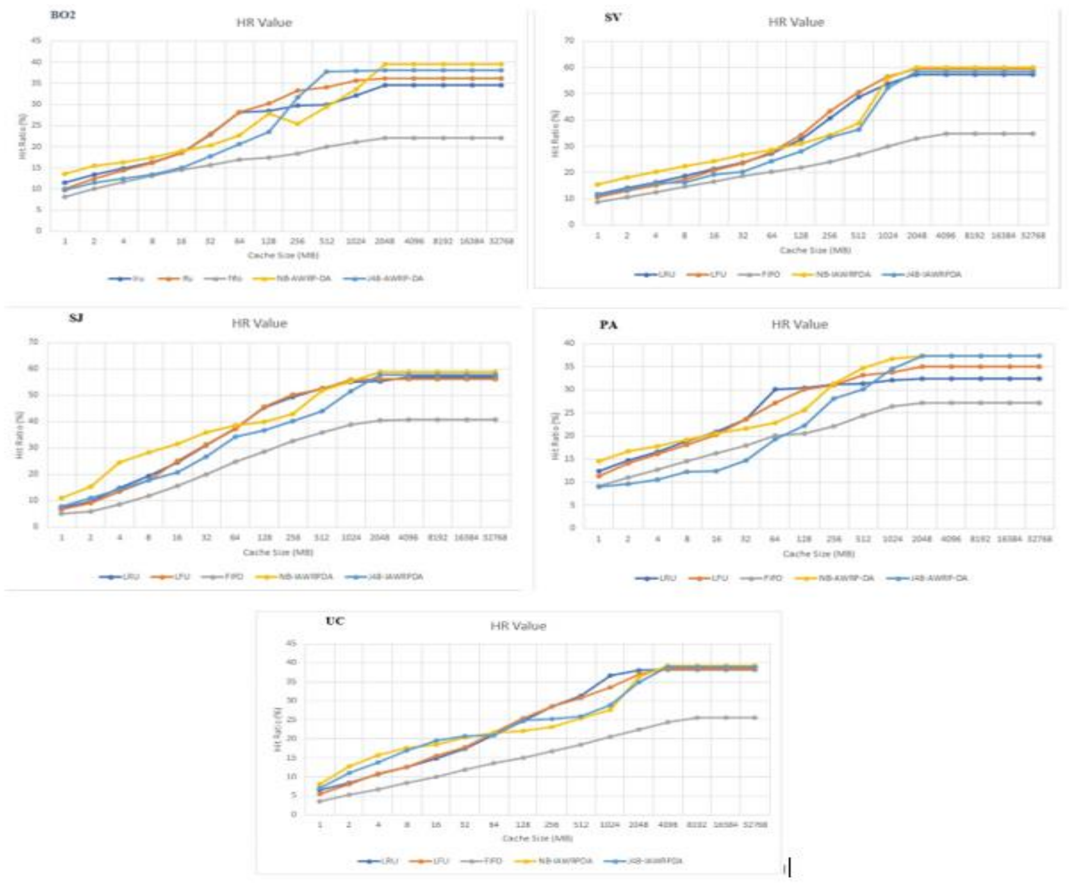

Figure 3. HR values for BO2, PA, SJ, SV and UC

Figure 4 shows BHR value for BO2, PA, SJ, SV and UC proxy datasets. The figure compares between proposed intelligent algorithms and traditional algorithms. It is obvious that the proposed NB-AWRP-DA achieve higher BHR value than LRU, LFU and FIFO algorithms by

0.9911%, 0.563% and 10.497% respectively. While J48-AWRP-DA rises BHR value by 0.0204% over LRU, by 0.0379% over LFU and by 10.61362% over FIFO algorithms. However, enhancing percentage of proposed algorithms over FIFO is much bigger than in LRU and LFU. This is because BHR value depends on the size object more than other features Thus, the BHR result is almost same when the cache size over 2048 MB. However, FIFO has the worst BHR value since it does not give property for any object depending on any features.

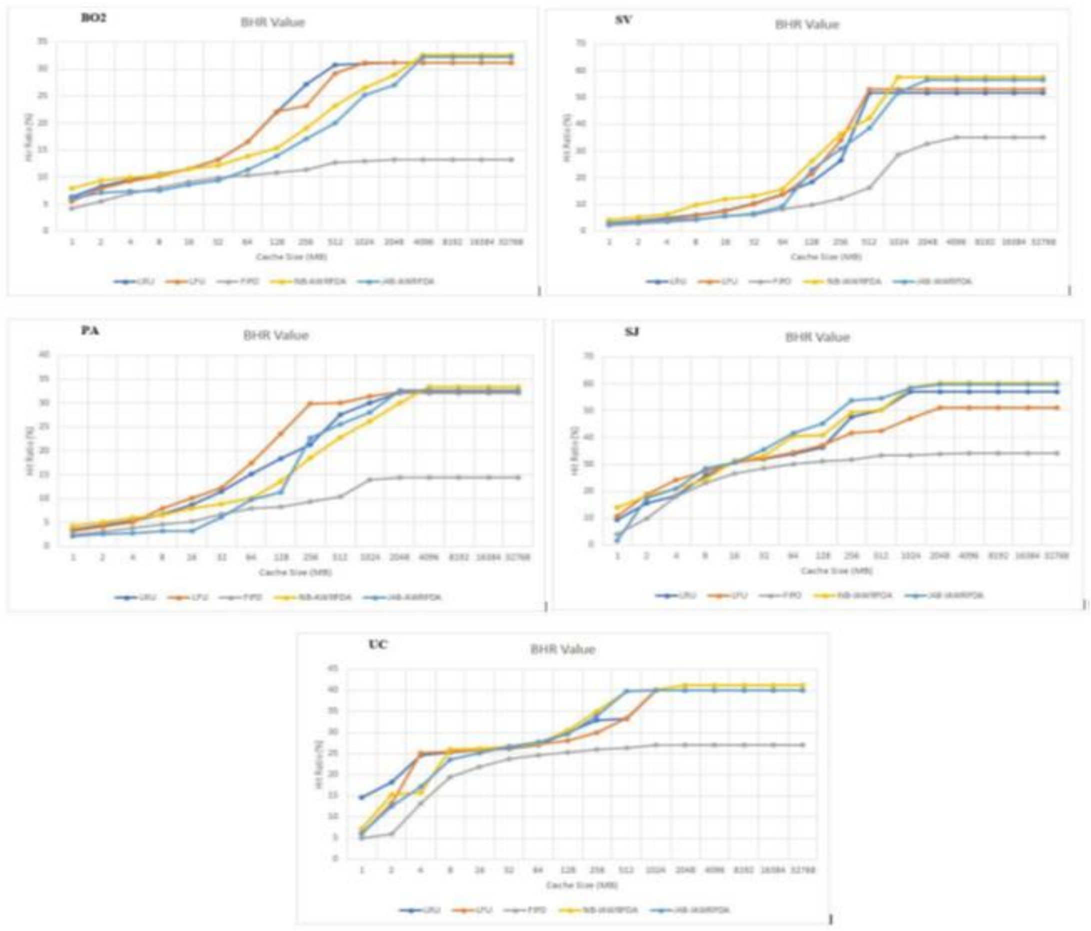

Figure 4. BHR value for BO2, PA, SJ, SV and UC proxy datasets

5. Conclusion

This research has proposed two intelligent web proxy cache replacement algorithms, namely Naïve Bayes Adaptive Weight Ranking Policy via dynamic aging (NB-AWRP-DA) and J48 Adaptive Weight Ranking Policy via dynamic aging (J48-AWRP-DA). To improve AWRP replacement algorithm performance over conventional replacement algorithms. Initially, NB and J48 classifiers are learned from proxy log files to predict the class of object which will be revisited again or not. Then integrate classifier with AWRP is done either by using probability in NB or predicting object class in J48. However, adding dynamic aging factor to the algorithms enhance their performance by preventing cache pollution problem.

The trained classifiers are integrated effectively with AWRP. Integrating NB depends on the probability of requested object while J48 depend on the predicted class of object. The result show integrating NB enhance web proxy cache performance more than J48 in terms on HR and BHR. Since probability gives precise expectation than using zero or one to represent class. Hence, the experimental result show that NB-AWRP-DA the average improvement ratio in terms of HR are 3.5244%, 4.008, 4.087 and 14.022% over J48-AWRP-DA, LRU, LFU and FIFO algorithms respectively. While average improvement ratio by NB-AWRP-DA in terms of BHR was 0.9706%, 0.9911%, 1.008% and 11.5842% over J48-AWRP-DA, LRU, LFU and FIFO algorithms respectively. Average enhancing ratio using NB-AWRP-DA and J48-AWRP-DA in terms of HR is larger than BHR. Hence the newely developed algorithm is suitable for database servers, media and other applications.

References

[1] Ali W, S Sulaiman, N Ahmad. 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*. 2014; 6(1).

[2] Sarhan A, AM Elmogy, SM Ali. *New Web cache replacement approaches based on internal requests factor*. in 9th International Conference on Computer Engineering & Systems (ICCES). 2014.

[3] Ali W, SM Shamsuddin, AS Ismail. Intelligent Naïve Bayes-based approaches for Web proxy caching. *Knowledge-Based Systems*. 2012; 31 :162-175.

[4] Sathiyamoorthi V, VM Bhaskaran. *Web caching through modified cache replacement algorithm*. in International Conference on Recent Trends In Information Technology (ICRTIT). Chennai, Tamil Nadu: IEEE. 2012.

[5] Kumar Saha, A., et al. An Optimization Technique of Web Caching using Fuzzy Inference System. *International Journal of Computer Applications.* 2012; 43(17): 20-23.

[6] Muralidhar K, DN Geethanjali. Improving the performance of the browsers using fuzzy logic. *International Journal of Engineering Research and Technology.* 2012; 3(1).

[7] ElAarag H. A quantitative study of web cache replacement strategies using simulation. *in Web Proxy Cache Replacement Strategies*. 2013: 17-60.

[8] Debabala Swain, BPaDS. AWRP: Adaptive Weight Ranking Policy for Improving Cache Performance. *Journal of Computing*. 2011; 3(2): 209-2014.

[9] Olanrewaju RF, AW Azman, M Yaacob. Intelligent web proxy cache replacement algorithm based on adaptive weight ranking policy via dynamic aging. *Indian Journal of Science and Technology*. 2016; 9(36).

[10] Ali W, SM Shamsuddin, AS Ismail. A survey of Web caching and prefetching. *Int. J. Advance. Soft Comput. Appl.* 2011; 3(1): 18-44.

[11] El Aarag H, S Romano. *Comparison of function based web proxy cache replacement strategies*. in International Symposium on Performance Evaluation of Computer & Telecommunication Systems. SPECTS. 2009.

[12] Khaleel MSA, SEF Osman, HAN Sirour. *Proposed ALFUR using intelegent agent comparing with LFU, LRU, SIZE and PCCIA cache replacement techniques*. in International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE). 2017.

[13] Ahmed WA, SM Shamsuddin. Neuro-fuzzy system in partitioned client-side Web cache. *Expert Systems with Applications*. 2011; 38(12): 14715-14725.

[14] Kumar PV, VR Reddy. Novel Web Proxy Cache Replacement Algorithms using Machine Learning Techniques for Performance Enhancement. *International Journal of Engineering Sciences & Research Technology*. 2014; 3(1): 339-346.

[15] Ali W, SM Shamsuddin. Intelligent Dynamic Aging Approaches in Web Proxy Cache Replacement. *Journal of Intelligent Learning Systems and Applications*. 2015; 7(4): 117.

[16] Ibrahim H., et al., Intelligent Cooperative Web Caching Policies for Media Objects based on Decision Tree Supervised Machine Learning Algorithm. 2014.

[17] Abdalla A, S Sulaiman, W Ali. Intelligent Web Objects Prediction Approach in Web Proxy Cache Using Supervised Machine Learning and Feature Selection. *International Journal of Advances in Soft Computing & Its Applications*. 2015; 7(3): 146-164.

[18] Irani KB. Multi-interval discretization of continuous-valued attributes for classification learning. 1993.

[19] Yasin W, et al. Windows web proxy caching simulation: A tool for simulating web proxy caching under windows operating systems. *Journal of Computer Science*. 2014; 10(8): 1380-1388.

[20] Liu X, Zhu J, Mao J, Shao X, Lu L. 2012. Design of real-time communication adapter for different protocol sensors in sensor Web. *Indonesian Journal of Electrical Engineering and Computer Science*; 10(5): 1101-1105.