Enhancement web proxy cache performance using Wrapper Feature Selection methods with NB and J48

Dua’a Mahmoud Al-Qudah, Rashidah Funke Olanrewaju, Amelia Wong Azman
Faculty of Engineering, International Islamic University Malaysia, Jalan Gombak 53100, Kuala Lumpur, Malaysia;
dm_q66@yahoo.com, frashidah@iium.edu.my, amy@iium.edu.my.

Abstract. Web proxy cache technique reduces response time by 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. Due to the limited size and excessive cost of cache compared to the other storages, cache replacement algorithm is used to determine evict page when the cache 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), Randomized Policy etc. may discard important pages just before use. Furthermore, using conventional algorithm cannot be well optimized since it requires some decision to intelligently evict a page before replacement. Hence, most researchers propose an integration among intelligent classifiers and replacement algorithm to improves replacement algorithms performance. This research proposes using automated wrapper feature selection methods to choose the best subset of features that are relevant and influence classifiers prediction accuracy. The result present that using wrapper feature selection methods namely: Best First (BFS), Incremental Wrapper subset selection(IWSS)embedded NB and particle swarm optimization(PSO)reduce number of features and have a good impact on reducing computation time. Using PSO enhance NB classifier accuracy by 1.1%, 0.43% and 0.22% over using NB with all features, using BFS and using IWSS embedded NB respectively. PSO rises J48 accuracy by 0.03%, 1.91 and 0.04% over using J48 classifier with all features, using IWSS-embedded NB and using BFS respectively. While using IWSS embedded NB fastest NB and J48 classifiers much more than BFS and PSO. However, it reduces computation time of NB by 0.1383 and reduce computation time of J48 by 2.998.

Keywords: web cache, replacement algorithms, wrapper feature selection, intelligent classifiers.

1. Introduction
As the rapid growth in network usage and significant role for network in our life, there is a need for good service, enough bandwidth and fast response. Web caching mainly depends on storing copies of web objects closer to user. In web caching, if the end user requests a page that founds in the cache, it is sent directly. Meanwhile, if it is not in the cache, fetching page from origin server is required. However, transferring of web object over the networks tends to increase network traffic. Thus, web caching helps in decreasing network traffic and reduce loads on the origin server [1, 2].

Meanwhile, due to limited cache size, appropriate cache replacement algorithms are required to manage web caching. Cache replacement algorithms are a process of evicting pages from cache to
make room for new requested page and there is no enough space for it. They have the significant effect in the web caching [3]. Although the conventional replacement algorithms such as LRU, LFU, and SIZE are not effective enough in web cache because they consider some features and ignore others which may have a good impact on web cache performance. Furthermore, those algorithms suffer from cache pollution problem where objects will be stored in the cache which may not revisited in future hence, waste of limited resources. Intelligent replacement algorithms are required to effectively utilize the limited resources (web cache) as well enhancing its performance by integrating more than one feature together [4].

extracting subset of features from space of features is called feature selection method. It has a significant impact on improving machine learning techniques performance. Since it is helped in removing noisy, irrelevant and unwanted features. In addition to fastest intelligent systems by reducing computation time [1, 2]. However, features selection method divided into filter methods, wrapper methods and embedded methods.

Chosen features for training intelligent system will strongly influence prediction process. Most strategies that used intelligent systems chose features manually while automated feature selection enhances prediction process by choosing the optimal and best subset of the features that have a significant impact on intelligent systems[1, 2]. Hence, this research study impact of using wrapper feature selection method to choose the best subset of features for training C4.5, NB intelligent classifiers, as well as prevent pollution problem. The paper is organized as follow section two discuss related works, section three provide a theoretical background overview, section four described and analysis the proposed methods and finally, the conclusion found in section five.

2. Related Works
Integrating of NB classifier with GDS, LRU and DA algorithms were proposed by [4]. The new proposed algorithms divided into two stages that are offline stage with reasonability to train NB classifiers and predict class of object. While online stage integrates NB with replacement algorithms and evaluate the proposed algorithms over conventional algorithms.[5] proposed NB-SIZE and C4.5-Hybrid algorithms which rely on integrating NB classifier and C4.5 decision tree with SIZE and Hybrid conventional algorithms, the proposed approaches enhance web proxy cache performance in terms of HR, BHR and latency. To enhance web cache performance in terms of HR and BHR [6] proposed integration between machine learning algorithms like SVM, NB and C4.5 decision tree with LFU_DA replacement algorithm. However, the feature that selected for training are chosen manually.

Co-operation between J48 classifier and LRU replacement algorithm was proposed by [7] for improving LRU performance and prevent cache pollution problem. The result illustrates that using J48 classifier boost LRU performance. While [8] integrate decision tree classifier with SIZ and Hybrid algorithms to form DT-SIZE and DT-Hybrid new intelligent replacement algorithms. Moreover, Random Forest Tree classifier (RFT) is used to design a novel RFT-LRU and RFT-GDSF intelligent replacement algorithms. RFT classifier used for class prediction purpose if will be revisited again or not. After that, the classifier integrates with LRU, GDSF to improve web cache performance in terms of HR and BHR[9]. A novel web cache replacement method is proposed by [10], it depends on using SVM for classification and divided the caching probability it will revisited again or not. After that use LFU algorithms when the cache is full. The proposed algorithm is called Hybrid LFU-SVM. The evaluation proves that Hybrid LFU-SVM enhance web cache performance in terms of HR and BHR over SVMLRU, LFU and LRU algorithms However, features that used in training intelligent classifiers that are mentioned before were chosen manually.

Abdalla, Sulaiman and Ali [1], suppose using automated method to extract best subset of features to improve the prediction process in intelligent systems which used in training phase by J48 classifier and NB classifier. Automated prediction scheme caused reducing in the number of features that selected and reducing dimensionality of dataset. Moreover, reduction in computation time is achieved, and improving accuracy of NB, C4.5 using Best First wrapper method compared to manual selection method. However, [2] propose using IWSS embedded NB feature selection method to extract the best
subset feature for Naïve Bayes classifier and later integrate Naïve Bayes classifier with Adaptive Weight Ranking Policy via Dynamic Aging (AWRP-DA) to improve the performance of web systems concerning Hit Ratio (HR) and Byte Hit Ratio (BHR).

3. Theoretical Background

3.1. Feature selection
Feature selection is a process of extracting subset of features from features space. It plays a key role in improving and enhancing the performance of machine learning techniques by reducing the effect of noise and irrelevant features. Moreover, it reduces the computational time and storage required to store features. Feature selection methods are classified in three groups, namely: Filter, Wrapper and embedded methods [11]. Hence filter and embedded methods are not considered in this research.

Wrapper methods are general-purpose algorithms they depend on learning algorithms to find best subset of features. Wrapper algorithms adding (forward selection) or deleting features (backward selection) one at a time from feature space until extract the best and optimal feature subset. However, wrapper's disadvantage is the computational cost involved [12]. The main idea of wrapper processes that using introduction algorithm as black box, and then run it over dataset. Partitioning data set into internal training and holdout set are done. However, this research uses best-first method, Incremental Wrapper Subset Selection with naïve bays method and Particle swarm optimization.

3.1.1. The Best First Method (BFS): BFS Searches the space of attribute subsets by greedy hill climbing augmented with a backtracking facility. Setting the number of consecutive non-improving nodes allowed controls the level of backtracking done. Best first can work either by FSS or BSS or Bi-directional. The main goal is to choose the most hopeful node that had been generated so far and that has not already been expanded. Best-first search stop when the goal is reached [13].

3.1.2. Incremental Wrapper Feature Subset Selection with Naïve Bayes (IWSS Embedded NB): It is a wrapper algorithm. But, Naïve Bays classifier embedded with it. Thus, it combines between linear time complexity of filter algorithms and advantages of wrapper methods. IWSSS always attempt to get best ranking features and once a feature is included it maintains them until end of the search. To solve this disadvantage of IWSS method adding NB classifier to it. NB allow algorithm to add new features or interchange new feature with feature that is previously added[14].

3.1.3. Particle Swarm Optimization (PSO): The idea of PSO is derived from social behaviors. Such that a group of birds flocking at promising position for attaining an accurate goal in a multidimensional space. In PSO the problem has been optimized by repeatedly trying to improve a candidate solution. It is like other evolutionary algorithms it has a population of solutions (swarm) of individuals (particles) that characterized depending to their position and velocity and moving them in search space depending on simple mathematical formula. To reach the optimal solution, there are two factors that particles depends on it to change searching direction. That are its own best previous experience (p-best) and the best experience of all other members (g-best) social part. Update that calculate velocity for each particles, Construction which determine particle moving and finally termination that terminated iterations if the iteration reach maximum number of iteration [15, 16].

3.2. Decision Tree
C4.5 is the most know algorithm to build decision tree. And using it is very common in different scopes like marketing, finance, computer science etc. The idea of C4.5 algorithm depends on building a tree start with root node which is the entire given dataset, then it works recursively to divide the data into smaller subsets by testing a given attribute at each node. The process terminated when the subsets are pure [17].
C4.5 Java code implementation in WEKA data mining tool is J48 classifier. WEKA detailed available at http://www.cs.waikato.ac.nz/ml/weka/. J48 classifier is used to predict output value depending on various features values of training data. Internal node in J48 represent various features. However, branches among nodes represent possible value of features. While output value represents by leaf node. Attribute that need to predicted is called dependent variable since it depends on all other variable values which are called (independent variable) [7].

3.3. Naïve Bayes

Naïve Bayes classifier is a simple supervised machine learning algorithm which depends on the Bayes' probability theory. It can deal with any number of attributes and classes. Moreover, Naïve Bayes technique is considered the fastest machine learning techniques and it is very easy to construct. In addition, its performance is very good in a lot of applications and applied well in many domains [4, 12].

In NB, all features under class label are considered conditionally independent. That leads to ignore any correlation among features. [17]. Discretizing process is required to convert continuous values into discrete values for better performance. Minimum Description Length (MDL) method will be used that described by [18] with default setup in WEKA software to convert continuous values into discrete values. After that, the dataset is ready to train NB. NB classifier trained to classify objects to revisited objects or not within forwarding sliding window. Prior probabilities Pr= (C =cj) and conditional priorities Pr (Ai= ai | C=cj) estimation required by NB to predict object's class. Probabilities calculation is shown in equation 1 and 2. Where Pr refers to probability value, C refers to class and A refers to attribute.

\[
Pr= (C =cj) = \frac{\text{number of examples of class } c}{\text{number of total examples}} \ldots \ldots (1)
\]

\[
Pr (Ai= ai | C=cj) = \frac{\text{number of examples with } Ai= ai \text{ and class } cj}{\text{number of examples of class } cj} \ldots \ldots (2)
\]

4. Proposed Methods

4.1. Raw Data Collection

proxy log files data had been collected from six proxy servers of IRCache. That collected from BO2, PA, SJ, SV, UC and SD servers that 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. Data collected in 9-1-2007 is used in training phase. Features that exist in the log trace files are 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.

4.2. Pre-processing

Data pre-processing is a process of removing noisy, incomplete, outlier and inconsistent data which presented in the dataset. Pre-processing done following three steps that are Parsing, filtering and finalizing.

In parsing boundaries between successive records in log files are determined. In addition to, determining each unique field inside each record. While, Filtering process done by Removing requests that generated by automated programs like web robots, spiders and crawlers. Since they cause traffic to web sites, removing requests for image files associated with requests for particular pages, removing irrelevant records, records that have unsuccessful status. We consider the success status which has 200 codes., removing any request has methods except GET and POST. And removing uncatchable requests that have "?" inside URL and cgi-bin requests. And Finalizing (for simplify) done by replacing each URL with unique random integer number. In addition to, using specified integer numbers to define object type like HTML = 1, Image = 2, Audio = 3, Video = 4, Application =
5 and Other types = zero. Finally, log files will contain the main fields that have impact from which the useful information can be extracted: URL Id, time stamp, elapsed time, object size, type of Web object.

4.3. Feature Extraction
In this state, the features are obtained from log files that play a significant role in web cache performance. The most common features that are extracted and used by researchers are recency, frequency, cost and size since they have an obvious influence in web cache [4, 19-23]. In table 1 features description and extracting procedure are described [1].

| Table 1. Features description and extracting procedure[1]. |
|-----------------------------|-----------------------------|-----------------------------|
| No. | Feature Name | Description | Calculation |
|-----|--------------|--------------|-------------|
| 1   | URL_ID       | Identification number for each URL. | Randomly generate unique integer number for each URL. |
| 2   | Timestamp    | Time when the client socket is closed in millisecond. | Given |
| 3   | Elapsed time | Time between accept () and close () of the client socket in millisecond. | Given |
| 4   | Size         | Number of bytes written to the client in bytes. | Given |
| 5   | Object type  | Web type object. | Given |
| 6   | Frequency    | Web object's frequency | Number of frequent occurs of web object. |
| 7   | SWL-frequency| Web object's frequency based on sliding window. | Swl-freq = \(\frac{swl \times freq + 1}{\Delta t \leq swl 1 otherwise}\) |
| 8   | IR-Frequency | Web objects frequency plus internal request frequency. | IR_Freq = freq + IR |
| 9   | Recency      | Web object's recency based on sliding window. | Recency= \(\max(sw, \Delta t), if object requested before swl, otherwise\) |
| 10  | Time Spend   | The time that spent in each page in millisecond. | The different timestamp in millisecond between current page and next one which immediately followed. |
| 11  | Mean         | The average of time spent in each page in millisecond. | The sum of time spent in each page for all its frequents divide by its number of frequents. |

Sliding window (SWL) was defined as a time before and after the request was made in millisecond. SWL should be around the mean time of object. They use it in calculate object's frequency and recency as shown in table 2. They trained BPNNs with different SWL values between 30 minute and two hours. The results show that when small SWL value uses the training performance is decreased. But, using large training dataset with small SWL value can increase training performance. They conclude that 30 minutes are good enough time periods for large dataset. Sarhan, Elmogy and Ali [21] define internal request as request of another web page originating from root web page. Counting object's frequency with internal request was done by increment frequency of main page with each subpage request. Thus, frequency of root page will be equal to its frequency plus subpages frequency which is called Internal Request Frequency (IR_Frequency).

4.4. 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 three different methods used in search method namely: BSF, IWSS-embedded-NB and PSO for both classifier NB and J48 with default Weka attributes for rest parameters. The following tables 3 and 4 present the best subset
of features that are chosen by each method. In general, IWSS-embedded NB choose the smallest subset of features while PSO choose larger number of features.

Table 2. The selected best subset for NB classifier in different datasets by different feature selection methods.

| NB       | BO2          | PA           | SJ          | SV           | UC          | SD          |
|----------|--------------|--------------|-------------|--------------|-------------|-------------|
| Best First | Id, Elapsed Time, Type, Freq, FIR, Swl_Freq, Recency, Time Spend, Mean | Id, Type, Freq, FIR, Swl_Freq, Recency | Id, Timestamp, Freq, FIR, Recency, Timespend, Mean | Id, Freq, FIR | Id, Type, Freq, Recency, Mean |
| Number of features | 9 | 6 | 7 | 7 | 2 | 5 |
| IWSS Embedded NB | Freq, FIR | Freq, FIR | ID, Timestamp, Type, Recency | FIR, ID, Type, Recency | Freq, Elapsedtime, Type, Freq, Mean, Recency, Timespend, Size | Freq, FIR, ID, Type, Mean, Timespend, Elapsedtime |
| Number of features | 2 | 2 | 7 | 4 | 9 | 7 |
| PSO | Freq, FIR | Id, Timestamp, Type, Freq, FIR, Swl_Freq, Recency, Mean | Id, Timestamp, Freq, FIR, Swl_Freq, Timespend, Recency, Mean | Id, Type, Freq, FIR, Swl_Freq, Recency | Id, Elapsedtime, Size, Type, Freq, Swl_Freq, Recency, Mean | Id, Elapsedtime, Size, Type, Freq, Recency, Mean |
| Number of features | 2 | 7 | 7 | 6 | 9 | 7 |

Table 3. The selected best subset for J48 classifier in different datasets by different feature selection methods.

| J48       | Bo2          | Pa           | Sj          | Sv           | Uc          | Sd          |
|----------|--------------|--------------|-------------|--------------|-------------|-------------|
| Best First | Id, Timestamp, Elapsedtime, Type, Freq, Fir, Recency, Timespend, Mean | Id, Timestamp, Freq, Fir, Type, Recency | Id, Timestamp, Freq, Fir, Type, Recency | Id, Timestamp, Freq, Fir, Recency | Id, Timestamp, Type, Freq, Fir, Mean, Recency, Timespend, Mean | Id, Timestamp, Type, Freq, Fir, Mean, Timespend, Elapsedtime |
| Number of Features | 9 | 6 | 9 | 6 | 6 | 9 |
| IWSS embedded NB | Freq, FIR | Freq, FIR | Id, Timestamp, Type, Freq, Fir, Recency | FIR, Id, Type, Recency | Id, Elapsedtime, Type, Freq, Fir, Mean, Recency, Timespend | Freq, Fir, Id, Type, Mean, Timespend, Elapsedtime |
| Number of Features | 9 | 6 | 9 | 6 | 6 | 9 |
4.5. Training

Training classifiers plays a significant role in replacement algorithm performance. Thus, each proxy datasets are divided into 70% for training and 30% for testing. After that, the data is prepared for training by discretizing it by using Minimum Description Length (MDL) method that suggested by [18] with default setup in WEKA 3.8.

The best subset of features which is chosen by BFS, IWSS embedded NB and PSO are used with the dataset to train and evaluate learning algorithms. Training data is prepared as follow:

1. Defining input features as \(<X_1\ldots X_n>\) features set.
2. Setting target output value \(Y\) to 1 if object is requested again within forward sliding window. Otherwise, it is setting to zero.
3. Setting input and output in pattern format as \(<X_1, X_2, \ldots X_n, Y>\).

When training pattern prepared in this form any classifier can be trained using this training pattern to predict if object revisited in future or not. Naïve Bayes and J48 classifiers are training and testing four times. Firstly, with all feature extracted which are URL_ID, timestamp, elapsed time, size, type, frequency, IR_frequency, SWL_frequency, recency, time spend and mean. Then using features extracted by Best First method, IWSS embedded NB method and Particle Swarm Optimization method, which are mentioned in detailed in table 3 for NB classifier and in table 4 for J48 classifier. The trained classifiers can be saved for later use.

5. Classifiers Evaluation and Discussion

NB and J48 classifiers are evaluated using Correct Classification Rate (CCR) or accuracy defined in (3), which is the most common performance measures are used. But, the datasets used in this research are imbalance. Thus, CCR or accuracy measures is not enough to assess classifier performance. In imbalance data one class has the majority and another class has minority instances in binary classification. So that sensitivity or true positive rate (TPR) defined in (4), specificity or true negative rate (TNR) defined in (5) and geometric mean (Gmean) defined in 6) are used to assessment classifier performance in addition to CCR [4]. Confusion matrix is presented in table 4.

\[
CCR = \frac{(TP+TN)}{(TP+FP+FN+TN)} \% \quad ... \quad (3)
\]

\[
TPR = \frac{TP}{(TP + FN)} \% \quad ... \quad (4)
\]

\[
TNR = \frac{TP}{(TN + FP)} \% \quad ... \quad (5)
\]
\[ Gmean = \sqrt{TPR \times TNR} \% \quad \ldots \quad (6) \]

**Table 4.** Confusion Matrix

|                  | Predicted positive | Predicted negative |
|------------------|--------------------|--------------------|
| Actual positive  | True Positive (TP) | False negative (FN) |
| Actual negative  | False Positive (FP)| True negative (TN)  |

In this study, web object that requested again within forward-looking window SWL which is 30 minutes belongs to the positive class otherwise it belongs to negative class. In this study, minority class was the positive class which is the most important one and the majority represent negative class.

Comparison among six different proxy datasets that are BO2, PA, SJ, SV, UC and SD are presented in evaluation tables for training. Each table presents comparison between NB and J48 classifiers result. In addition, it presents compare between using NB and J48 classifiers with all features and with using features that had been extracted using automated feature selection methods.

**Table 5:** CCR of different proxy datasets Training phase

| Datasets | NB Training | J48 Training |
|----------|-------------|--------------|
|          | WFS | Best First | IWSS embedded NB | PSO | WFS | Best First | IWSS embedded NB | PSO |
| BO2      | 91.57% | 91.93% | 91.93% | 91.93% | 91.93% | 91.93% | 91.93% | 91.93% |
| PA       | 85.53% | 85.44% | 85.79% | 85.44% | 87.74% | 87.67% | 85.80% | 87.57% |
| SJ       | 77.50% | 78.67% | 79.38% | 81.22% | 86.49% | 86.49% | 86.46% | 86.59% |
| SV       | 82.38% | 82.47% | 82.70% | 82.47% | 88.67% | 88.67% | 83.19% | 88.67% |
| UC       | 89.00% | 90.20% | 90.19% | 90.20% | 91.91% | 91.91% | 90.35% | 91.91% |
| SD       | 92.88% | 94.22% | 94.16% | 94.22% | 96.51% | 96.51% | 94.21% | 96.61% |
| AVG      | 86.48% | 87.15% | 87.36% | 87.58% | 90.54% | 90.53% | 88.66% | 90.57% |

Table 5 CCR training table presents that both of NB and J48 give a satisfactory performance. But in the most cases J48 gives a higher accuracy than NB. In average, both classifiers give higher accuracy value using features selected by feature selection methods than without using them. NB training average CCR without using feature selection methods, using Best First, using IWSS embedded NB and PSO are 86.48%, 87.15%, 87.36% and 87.58% respectively. whereas, in training J48 the result was as follow 87.58%, 90.54%, 90.53%, 88.66% and 90.57%. Accuracy results for NB and J48 classifiers indicate that PSO feature selection method boost classifier performance in average more than other feature selection methods.

Table 6 illustrate True Positive Rate (TPR) or sensitivity result for NB and J48 classifiers in training and testing. The result show that TPR for J48 classifier is better than NB classifier. However, both classifier do not have a very high TPR. This is referring to data imbalance since positive class represent the minority. Moreover, SD and UC datasets give the higher sensitivity than other proxy
datasets since they have larger number of instances than others. That indicate when the data is large then it will give better indication and better performance for classifiers.

**Table 6. TPR of different proxy datasets Training phase**

| Datasets | NB Training | J48 Training |
|----------|-------------|--------------|
|          | WFS         | Best First   | IWSS embedded NB | PSO | WFS          | Best First | IWSS embedded NB | PSO |
| BO2      | 53.2        | 51.1         | 51.1             | 51   | 51.1         | 51         | 51.6             |
| PA       | 25.7        | 27.3         | 23.2             | 27.3 | 39.2         | 38.6       | 23              | 38.6 |
| SJ       | 24.3        | 25           | 33.5             | 41.1 | 78.9         | 78.9       | 78.8            | 79   |
| SV       | 30.5        | 29.8         | 29.6             | 29.8 | 67.7         | 67.7       | 35.8            | 67.7 |
| UC       | 49.4        | 60.6         | 60.6             | 60.6 | 70.3         | 70.3       | 61.1            | 70.3 |
| SD       | 58.1        | 79.3         | 70.4             | 78.3 | 82.1         | 82.1       | 78.3            | 82.2 |
| AVG      | 40.2        | 45.5166      | 44.733333        | 48.0333 | 64.866 | 64.783 | 54.666666 | 64.9 |

In contrast to TPR result the NB classifier give higher TNR than J48. And the TNR result is very high compared to TPR as presented in table 7. This is because the negative class represent by majority of instances. Moreover, using PSO feature selection method promote the classifier performance in general.

**Table 7. TNR of different proxy datasets Training phase**

| Datasets | NB Training | J48 Training |
|----------|-------------|--------------|
|          | WFS         | Best First   | IWSS embedded NB | PSO | WFS          | Best First | IWSS embedded NB | PSO |
| BO2      | 99          | 99.9         | 99.9             | 99.9 | 99.9         | 99.9       | 99.9            |
| PA       | 98.9        | 98.5         | 99.8             | 98.5 | 98.6         | 98.7       | 99.9            | 98.7 |
| SJ       | 97.7        | 99           | 96.8             | 96.5 | 89.4         | 89.4       | 89.4            | 89.4 |
| SV       | 97.7        | 98.1         | 98.2             | 98.1 | 94.9         | 94.9       | 97              | 94.9 |
| UC       | 96.9        | 96.1         | 96.1             | 96.1 | 96.2         | 96.2       | 96.2            | 96.2 |
| SD       | 96.1        | 96           | 96.9             | 96   | 98.1         | 98.1       | 96              | 98.1 |
| AVG      | 97.716      | 97.933       | 97.95            | 97.5166 | 96.18 | 96.2 | 96.4            | 96.2 |

Good classifier the one which achieve high TPR and high TNR, but due to data imbalance Gmean measure is needed. Since it gives mean value between TPR and TNR. Thus, it gives better indication
for both together. Gmean measure provides the balanced performance of learning algorithm between two classes positive and negative. Thus, Gmean is used when both classes positive and negative are considered. Table 8 present Gmean result for training NB and J48 classifiers. It can be seen obviously that J48 has higher Gmean than NB. In addition, PSO is the best selection method used to improve classifiers performance. In general, using automated feature selection methods boost classifiers performance.

**Table 8. Gmean of different proxy datasets Training phase**

|Datasets| BO2| PA| SJ| SV| UC| SD| AVG|
|---|---|---|---|---|---|---|---|
|NB Training| WFS| Best First| IWSS embedded NB| PSO| WFS| Best First| IWSS embedded NB| PSO|
|J48 Training| WFS| Best First| IWSS embedded NB| PSO| WFS| Best First| IWSS embedded NB| PSO|
|---|---|---|---|---|---|---|---|---|
|---|72.5727|71.4485|71.448512|71.4485|71.378|71.448|71.378568|71.797|
|---|50.4155|51.8560|48.11817|51.8560|62.170|61.723|47.934330|61.723|
|---|48.7248|49.7493|56.945588|62.9773|83.986|83.986|83.932830|84.039|
|---|54.5880|54.0682|53.914005|54.0682|80.154|80.154|58.928770|80.154|
|---|69.1871|76.3129|76.312908|76.3129|82.236|82.236|76.666942|82.236|
|---|74.7222|87.2513|82.593946|86.6994|89.744|89.744|86.699480|89.798|
|---|62.6754|66.7652|66.193881|68.4401|78.987|78.944|72.593847|79.015|

A comparison among computation training time for NB and J48 classifiers without using feature selection methods and with using them are illustrated in table 9. It is obvious that NB classifier is faster than. In the average using IWSS embedded NB feature selection method with both classifier fastest classifier training process than using other feature selection methods. Using automated selecting methods decrease the number of features that used by classifier in addition to exclude irrelevant features. Thus, the classifiers run faster. In average, computation time for NB classifier are 0.216666667, 0.086666667, 0.086666667 and 0.096666667 sec for WFS, using Best First, IWSS embedded NB and PSO respectively. For J48 computation time finished with 4.268333333, 2.201666667, 1.27 and 2.545 sec using WFS, using Best First and PSO respectively.

At all, the results for each classifier using different feature selection methods was little bit different because the features chosen by them were almost similar differed in one or two features or sometimes they were same. Moreover, some web proxy files provide better result and good indication than other because they have larger number of instances inside them than other. This obvious in SD proxy which gives in general the best result at all since it is the largest log file used. Imbalanced data was the reason behind large diversity between TPR and TNR for each classifier. Clearly, that both classifier give satisfactory performance and using feature selecting methods boost the performance of NB classifier more than j48. In addition, they minimize the number of feature used and fastest the training classifiers operations.

**Table 9. Computation Time (sec) of different proxy datasets Training phase**
Datasets | NB Training | J48 Training |
|---------|-------------|-------------|
|         | WFS | Best First | IWSS embedded NB | PSO | WFS | Best First | IWSS embedded NB | PSO |
| BO2     | 0.18 | 0.04 | 0.04 | 0.94 | 0.89 | 0.31 | 0.62 |
| PA      | 0.06 | 0.03 | 0.02 | 1.29 | 0.63 | 0.12 | 1.19 |
| SJ      | 0.09 | 0.07 | 0.07 | 2.66 | 1.86 | 1.06 | 2.14 |
| SV      | 0.12 | 0.11 | 0.03 | 2.44 | 2.85 | 0.34 | 2.77 |
| UC      | 0.34 | 0.07 | 0.09 | 2.37 | 2.31 | 1.75 | 3.2  |
| SD      | 0.51 | 0.2  | 0.22 | 15.91 | 4.67 | 4.04 | 5.35 |
| AVG     | 0.21666 | 0.08666 | 0.078333 | 0.09666 | 4.2683 | 2.2016 | 1.27 | 2.545 |

6. Conclusion
The training result of NB and J48 classifiers show that using feature selection methods to choose best subset of features enhance classifier performance in terms of accuracy (CCR), sensitivity (TPR) and specificity (TNR). In addition, PSO result show that it is the best selection method used to enhance performance of NB and J48. Meanwhile, computation time for both classifier is decreased using IWSS embedded NB more than BFS and PSO.

The experimental result shows that using feature selection methods enhance classifiers performance to predict class of object accurately. However, this is because it is reduced the number of feature that the classifiers deal with and remove irrelevant feature. In addition, to reduce computation learning time for the classifiers. Moreover, PSO selection method enhance classifiers performance more than BFS and IWSS embedded NB selection methods.

The experimental results show that J48 classifier with PSO achieve higher accuracy (CCR), TPR and Gmean than NB classifier by 2.99%, 16.866 and 10.5749% respectively. However, in term of computation time NB classifier with IWSS embedded NB reduces computation time of NB classifier training by 0.1383 sec. However, IWSS embedded NB speed up J48 classifier by reducing computation time with 2.998 sec. Moreover, NB classifier is faster than J48 classifier. In the future, more feature selection methods would be used with more numbers of classifiers. and integration those classifiers with web cache replacement algorithms should be done.

7. References
[1] Abdalla A, Sulaiman S and Ali W 2015 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 7 146-64
[2] Olanrewaju R F, Alqudah D, Azman A W and Yaacob M 2016 Intelligent web proxy cache replacement algorithm based on adaptive weight ranking policy via dynamic aging Indian Journal of Science and Technology 9
[3] Muralidhar K and Geethanjali D N 2012 Improving the performance of the browsers using fuzzy logic International Journal of Engineering Research and Technology 3
[4] Ali W, Shamsuddin S M and Ismail A S 2012 Intelligent Naïve Bayes-based approaches for Web proxy caching Knowledge-Based Systems 31 162-75
[5] Kumar P V and Reddy V R 2014 Novel Web Proxy Cache Replacement Algorithms using Machine Learning Techniques for Performance Enhancement INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY 3 339-46

[6] 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

[7] Ibrahim H, Yasin W, Hamid N A W A and Udzir N I 2014 Intelligent Cooperative Web Caching Policies for Media Objects based on Decision Tree Supervised Machine Learning Algorithm

[8] Kumar P V and Reddy V R 2014 Web proxy cache replacement policies using decision tree (DT) machine learning technique for enhanced performance of web proxy International Journal on Recent and Innovation Trends in Computing and Communication 2 302-9

[9] Benadit J P, Francis S F and Nadhiya M 2014 Enhancement of Web Proxy Caching Using Random Forest Machine Learning Technique International Journal of Computer Science Issues (IJCSI) 11 83

[10] Aimtongkhram P, So-In C and Sanguanpong S 2016 A novel web caching scheme using hybrid least frequently used and support vector machine. In: Computer Science and Software Engineering (JCSEE), 2016 13th International Joint Conference on: IEEE) pp 1-6

[11] Chandrashekar G and Sahin F 2014 A survey on feature selection methods Computers & Electrical Engineering 40 16-28

[12] Ratnamahatana C A and Gunopulos D 2002 Scaling up the naive Bayesian classifier: Using decision trees for feature selection

[13] Kohavi R and John G H 1997 Wrappers for feature subset selection Artificial intelligence 97 273-324

[14] Bermejo P, Gámez J A and Puerta J M 2014 Speeding up incremental wrapper feature subset selection with Naive Bayes classifier Knowledge-Based Systems 55 140-7

[15] Lin S-W, Ying K-C, Chen S-C and Lee Z-J 2008 Particle swarm optimization for parameter determination and feature selection of support vector machines Expert systems with applications 35 1817-24

[16] Moraglio A, Di Chio C and Poli R 2007 Genetic Programming: Springer) pp 125-36

[17] 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

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

[19] Ali W, Shamsuddin S M and Ismail A S 2012 Intelligent Web proxy caching approaches based on machine learning techniques Decision Support Systems 53 565-79

[20] Romano S and ElAarag H 2011 A neural network proxy cache replacement strategy and its implementation in the Squid proxy server Neural computing and Applications 20 59-78

[21] Sarhan A, Elmogy A M and Ali S M 2014 New Web cache replacement approaches based on internal requests factor. In: 9th International Conference on Computer Engineering & Systems (ICCES): IEEE) pp 383-9

[22] Sathiyaamoorthi V and Bhaskaran V M 2012 Web caching through modified cache replacement algorithm. In: International Conference on Recent Trends In Information Technology (ICRTIT). (Chennai, Tamil Nadu: IEEE) pp 483-7

[23] 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. In: Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services: ACM) pp 262-9