Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 46 (2015) 184 – 193

Procedia Computer Science

International Conference on Information and Communication Technologies (ICICT 2014)

Improving the Performance of a Proxy Cache Using Tree Augmented Naive Bayes Classifier

Julian Benadit.P[a,*], Sagayaraj Francis.F[a], Muruganantham.U[b]

[a]*Department of CSE, Pondicherry Engineering College, Pondicherry, 605014, INDIA*
[b]*Department of CSE, Dr.SJSPMCET, Pondicherry University, Pondicherry, 605502, INDIA*

Abstract

In this paper, we attempt to improve the performance of Web proxy cache replacement policies such as LRU and GDSF by adapting a semi naïve Bayesian learning technique. In the first part, Tree Augmented Naive Bayes classifier (TANB) to classify the web log data and predict the classes of web objects to be revisited again future or not. In the second part, a Tree Augmented Naïve Bayes classifier is incorporated with proxy caching policies to form novel approaches known as TANB-LRU and TANB-GDSF. This proposed approach improves the performances of LRU and GDSF in terms of hit and byte hit ratio respectively.

Peer-review under responsibility of organizing committee of the International Conference on Information and Communication Technologies (ICICT 2014)

Keywords: Proxy Caching;Cache replacement;Classification;Tree Augmented Naive Bayes classifier

1. Introduction

As the World Wide Web and users are explicating at a very rapid rate, the performance of World Wide Web systems becomes rapidly high. Web caching and prefetching is the one of the best methods for improving the performance of the proxy server. The basic idea in web caching is to retain the most popular web log data in a proxy cache, such that the performance of web proxy cache would have improved, when it is accessed from the user. The

* Corresponding author. Tel.:+ 919629321243.
E-mail address: benaditjulian@gmail.com

Peer-review under responsibility of organizing committee of the International Conference on Information and Communication Technologies (ICICT 2014)
doi:10.1016/j.procs.2015.02.010

main idea behind the web caching concept is a web cache replacement, algorithm and its key parameters in the algorithms.

To ameliorate the functioning of a proxy cache, a lot of research work has been done in their cache replacement policies.Table 1. Presents a summary of the existing Cache Replacement Policies(CRP)[6,7].Most of these replacement algorithms consider only certain key factors and assign a key value based on the priority for each web document which stored in the cache. But, it is difficult to have a better cache replacement policy that performs well in all situations, because each replacement policy has a different key parameter to optimize web resources. Moreover, various elements can act upon the cache replacement policy to receive a better replacement decision and it is not an easy task for because one parameter is more significant than the other one. Due to this restriction, there is a need for an active method which intelligently manages the web proxy cache by satisfying the objectives of web caching requirement. So this platform promotes, for the integration of the machine learning methods in the web caching replacement algorithms.

In our previous surveys, the intelligent techniques have been applied in web caching algorithm. By extensive use of this prediction pattern, the caching algorithms become more efficacious and adapted for the use of web cache environment, other than the classic web caching algorithms which are already in practice. Moreover, there are multiple users who are fond of the access of web cache algorithms, but it is very necessary to prove a prediction model, which are upgraded often so that web objects can be revisited in the future on a proper standard. In this paper, we proposed the Tree Augmented Naïve Bayes classifier (TANB)[7] to train the web object based on the recurrent sliding window method for their systematic classification, so that the web objects can be predicted in the future or not. We then formulated the semi- naive Bayesian learning method called TANB to classify the web objects and then it is incorporated with traditional caching algorithm called TANB-LRU and TANB-GDSF to improve its web caching performance on a better channel.

Table 1. Cache Replacement Policies.

CRP	Key Parameter	Cache Replacement Technique
LFU	Number of References.	The least frequently accessed first.
LRU	Time Since last access.	The least recently accessed first.
GDS	Document Size S_d . Document cost C_d . An Inflation Value L.	Least value first according to value $p_i = C_d / S_d + L$.
GDSF	Document Size S_d . Document cost C_d . Number of non-aged references f_d time since last access. Temporal correlation measure β . An Inflation Value L.	Least value first according to value $p_i = (C_d \times F_d / S_d)^{\beta} + L$.

The structures of this paper with the features are processed below. In section 2, we present the related work with an Existing machine learning methods towards web caching policies. The Section 3, proposes the brief introduction of Tree Augmented Naïve Bayes classifier model. In Section 4 we introduce the proposed novel web proxy caching approach integrates with the cache replacement algorithm. Experimental results and Performance Evaluations are presented in Section 5 and Section 6. The Section 7 concludes our paper with revealed results.

2. Related Work

Web Caching holds an important role in improving the performance of the web proxy cache. The basic idea of web caching is known for its Replacement Policy, which calculates the most popular web object by storing it in the proxy cache, hence that the popular web documents can be put back with the unpopular ones. Most of the common

replacement algorithm assigns new key value computed by factors such as size, frequency and cost. Using this key value, we would rate the top most web documents on their corresponding key-values.

In preceding Paper, it exploits supervised learning methods to cope with the matter[1,3,5,11]. Most of the recent surveys use Back Propagation Neural Network, Naïve Bayes, and Decision Tree in world-wide caching which is presented in Table 2. Though BPNN training might consume a wide amount of time and need further process overheads. Moreover, the Naïve Bayes result in less accuracy in classifying the large web data sets and similarly decision tree also result in less prediction accuracy in training large data sets and it consumes more memory space, So in this paper, we have attempted to increase the performance of web cache replacement, strategies by integrating semi Naïve Bayes Learning classifier[7] called TANB.

Table 2. Summary of Existing Machine Learning Methods towards web caching policies.

Intelligent methods	Key Parameter	Eviction
NB[1]	The training data are classified based on the probability of each class given in the document.	Violation of Independence assumption Zero conditional probability problem.
ID3[11]	A model based on decision trees consist of a series of simple decision rules, often presented in the form a graph.	Not good for predicting the continuous class attributes. Low prediction accuracy, high variance.
BPNN[5]	Back-propagation neural network uses a supervised learning method at each layer to minimize the error. The error generated in each functional hidden unit is an average of the evaluated error.	In the training phase the target output is not clear and it consumes more time for classification. The storing web object is more complicated.

In conclusion, we achieved a large scale evaluation compared with other classifier like Back Propagation Neural Network, Naïve Bayes, and Decision Tree in term, classification accuracy like Precision and Recall on different log data sets we collected and the proposed method has improved the performance of the proxy cache in terms of hit and byte hit ratio.

3. Tree Augmented Naive Bayes Model

The Naive Bayes model, encodes incorrect independence assumptions that, given the class label, the attributes are independent of each other. Only in the actual universe, the attributes of any organization are mostly correlated and the case as in the Naive Bayes rarely happens. Hence, if the model also takes into account the correlation between the properties, then the classification accuracy can be bettered.

The answer to this problem can be accosted by a semi-naïve Bayes learning method called Tree augmented Naive Bayesian network[7].In a TAN model, see Fig. 1. all the variables are related to class variables by means of the directed edges.

If there are 'N' Variables in a scheme, and then the corresponding tree structure will have 'N' nodes. Thus, N-1 edges should be added, to get a tree structure that connects all the lymph glands in the graph. Also, the sum of the weights of all these edges needs to be the maximum weight among all such tree structures.

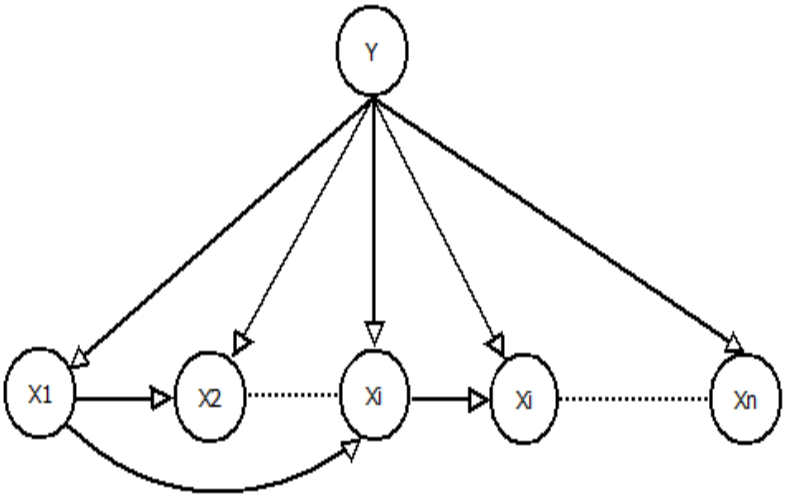

Fig. 1. Tree Augmented Naïve Bayes Model.

Given a pair of variables, this method measures how much data from one variable provides about the other making a tree for the TAN model, The conditional mutual information used for constructing TAN as given, which is defined equally in Equation 1.

$$I_p(X, Y \mid Z) = \sum_{x,y,z} P(x, y, z) \log \left(\frac{P(x, y \mid z)}{P(x \mid z) P(y \mid z)} \right) \tag{1}$$

The TAN implementation also applies the same Laplace correction discussed in the Naïve Bayes model. In this work place the source variable is always the first variable found in the dataset. The formula for the TAN classification is given by Equation 2.

$$P(C \mid X_1, ..., X_n) = P(C).P(X_{root} \mid C) \prod_i P(X_i \mid C, X_{parent}) \tag{2}$$

The key feature in the TAN model is the tree structure. In order to construct a tree, the parent of each the attribute needs to identify. Also, only the most correlated attributes must be connected to each other, then the complexity of the TAN model is polynomial, while the complexity of the Bayesian network is exponential. Thus, by restricting the level of dependency between the properties, we can have a less complex model. The algorithm to construct a tree for the TAN model is as follows:

The tree construction method consists of five steps which is represented below in Algorithm I.

Algorithm I. Existing Tree Augmented Naïve Bayes Classifier

Step 1. Calculate $I_p(X_i, X_j \mid C)$ between each pair of attributes $i \neq j$.

Step 2. Construct a complete undirected graph in which the vertices are the attributes $X_1 \ldots\ldots X_n$ and interpret the weight of an edge connecting X_i to X_j by $I_p(X_i, X_j \mid C)$ in Equation 1.

Step 3. Develop a maximum weighted spanning tree.

Step 4. Alter the undirected tree to a directed one by randomly choosing a root variable and setting the direction of all the edges outward from the root[7].

Step 5. Construct a Semi-Naïve Bayes learning method called Tree augmented naïve Bayes by adding each vertex labelled by C and adding a directional edge from C to each X_i.

4. Proposed Novel Web Proxy Caching Approach

The proposed method will present a working flow see Fig. 2. For novel web proxy caching based on the proposed machine learning technique. In our proposed work we use the semi-naïve Bayesian learning[7] method called Tree Augmented naïve Bayes for classifying the datasets that can be revisited again or not in the future. The mining steps consist of two different phases for classifying the datasets in the first phase, we pre-process to remove the irrelevant information in the proxy log data sets, a different technique is applied at the preprocessing stage such as data cleaning, data filtering and data integration. Once this task has been accomplished, the proxy sets have been trained in the second phase by the classifier TANB, which predicts whether the web objects that; can be revisited once again in the future or not. In the third phase, the predicted web object[4] has been integrated to the traditional web proxy caching algorithm like LRU and GDSF for replacement strategies.

4.1. Tree Augmented Naïve Bayes-Input/Output System Based On Recurrent Sliding Window Mechanism

The input parameters to Tree Augmented Naïve Bayes in the Table 3. are labelled as $\{R_i, F_i, RT_i, Size_i\}$ and the predicted output as Target_o. Frequency (F_i) and Recency (R_i) for the objects are estimated based on the recurrent sliding window mechanism[5]. The recurrent Sliding window method is used to obtain the Recency value based on the time before and after the request is made. Otherwise, its recency will have the maximum value among (SWL_i) and therefore the Recency of web object is calculated as follows in Equation 3.

$$\text{Recency} = \begin{cases} \max(SWL_i, \Delta T_i) & \text{if } obj_i \text{ requested before} \\ SWL_i & \text{if } obj_i \text{ request for the first time} \end{cases} \tag{3}$$

Frequency of object, Obj_i is incremented by 1 with respect to an previous frequency value, if the request for Obj_i is within the time interval, or within the boundary of backward-looking SWL, otherwise the frequency value will re-initialize to 1 see in Equation 4. Target output (Target_o) will set to 1, if the object (Obj_i) is re-visited again with the forward sliding window; else Target_o will be 0.

$$\text{Frequency} = \begin{cases} F_i + 1 & \text{if } \Delta T_i \le SWL_i \\ \text{Max}\left[\dfrac{\text{frequency}}{\dfrac{\Delta T_i}{SWL_i}}, 1\right] & \text{if } obj_i \text{ beyond } SWL_i \end{cases} \quad (4)$$

In, Table 3. Presented below uses the various parameters for training the log data sets using the Recurrent sliding window method. This method trains the data sets using sliding window length, Frequency and Recency of the web object. After the training data sets are prepared, these datasets are then integrated with TANB classifier. The TANB classifier takes the input parameters and classifies the data sets. After the data sets are classified, it has been used for web proxy caching algorithm for replacement.

Table 3. Input Parameters for TANB and Recurrent Sliding Window Method.

TANB Parameters	Meaning
R_i	Recency of web object.
F_i	Frequency of web object.
RT_i	Retrieval Time of web object.
$Size_i$	Size of web object.
Recurrent Sliding window Parameters	**Meaning**
Obj_i	Requested web object.
ΔT_i	Time Since Obj_i was last requested.
F_i	Frequency of Obj_i within sliding window.
SWL_i	Sliding Window Length.
$Target_o$	Target output.

4.2. Tree Augmented Naïve Bayes-Greedy Dual Size Frequency (TANB-GDSF)

Tree Augmented Naïve Bayes learning method integrates with Greedy Dual Size Frequency[2,6] to improve the performance of Byte Hit Ratio. In this method, the TANB uses as inputs the recency and frequency of the web object based on the recurrent sliding window mechanism, and as well as Retrieval time, the size of the Web object, and the classifier produces a target output in order to indicate whether the web object can visited in future or not see in Equation 5. Later, this re-visited data can be classified as cacheable data in order to increase the performance of a replacement algorithm in terms of the cache hit ratio.

$$K_i = F_d \times C_d / S_d + L + Target_o \quad (5)$$

4.3. Tree Augmented Naïve Bayes-Least Recently Used (TANB-LRU)

Least Recently Used is the most common algorithm among all the caching algorithms[4,6]. But, this algorithm suffers from cache Contamination, i.e. the unpopular data will remain in the proxy cache for a longer period and it suffers from cache pollution. For reducing cache contamination in LRU, a TANB classifier is integrated with LRU to form a new approach Called TANB-LRU in order to increase the performance of the Byte hit ratio.

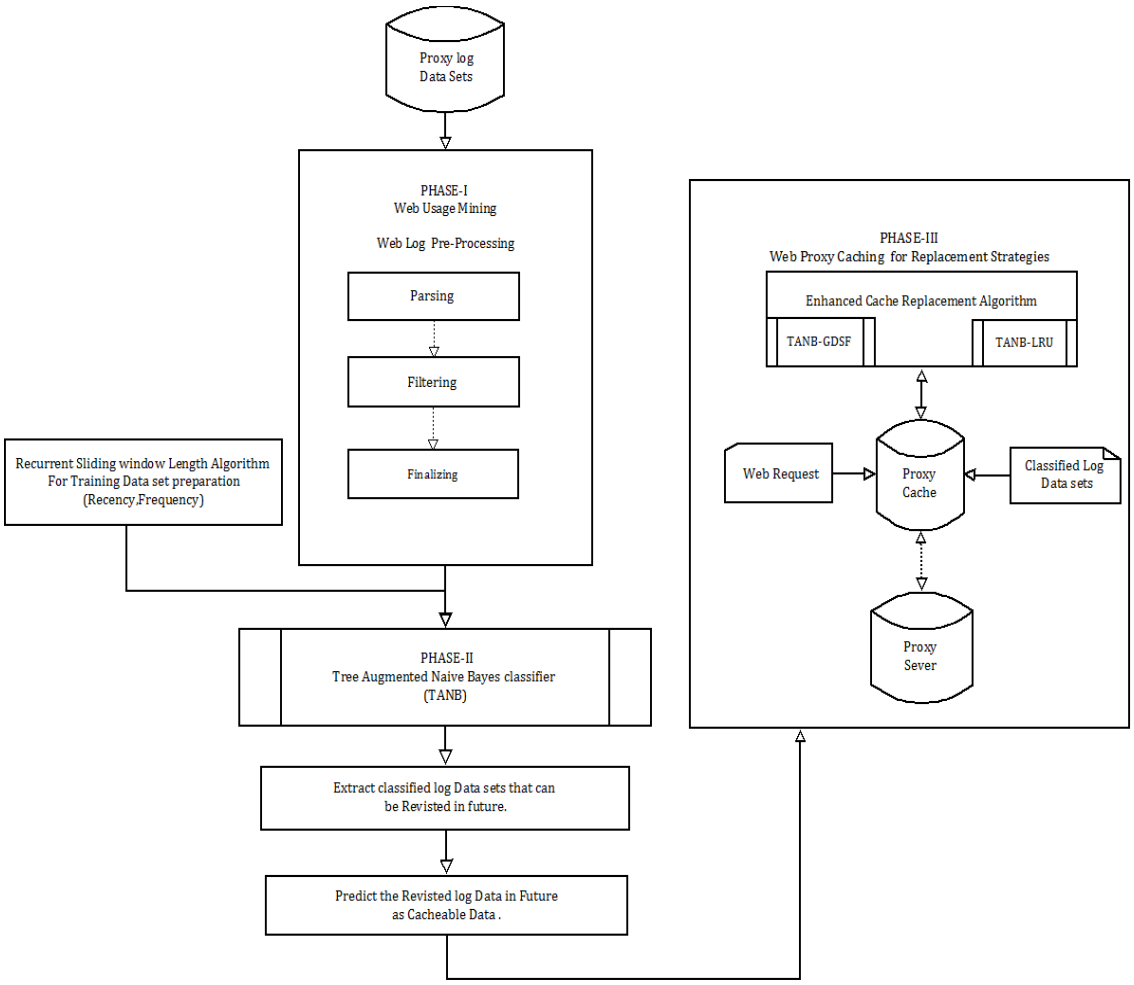

Fig. 2. Working Flow of Web Proxy Caching Approach based On Tree Augmented Naïve Bayes Classifier.

5. Experimental Results

5.1. Web Log Pre-processing

The Data sets[14] to be used for simulation undergo some pre-processing technique to remove irrelevant requests, extract useful information. The steps involved in web log pre-processing[9] are Data cleaning, Parsing and Filtering which removes irrelevant request and to identify the boundaries between the successive records stored. Finally an output format generated after the pre-processing[1] steps consist of the following fields shown in Table 4.

5.2. Training Phase

In this phase, the Pre-processed log file was trained by the recurrent sliding window method SWL[2,5] in order to obtain the Frequency and Recency of the web object as shown in Table 3. In this paper, we have used the Recurrent Sliding window length of 15 minutes, i.e. (900sec) as the time period to train the pre-processed data sets for the recency, frequency, value as the object, usually stays in the cache. This generated recency, frequency obtained by Recurrent sliding window and its timestamp, the size of the object used as the input for the semi naïve Bayes learning method called TANB which classify the web object to could be re-visited again or not. In this phase,

the trained log file has been cross-validated as testing and training data. Thus the concluded data set is arranged and normalized according to the series [0,1] i.e. Cacheable or un-cacheable data. Here we have used the open source software Keel[13] for the TANB classifier, finally the classified data set has been integrated into the web proxy caching algorithms LRU, GDSF for Replacement strategies.

Table 4. Sample Training Data Based on Recurrent Sliding window Mechanism.

URL-ID	Timestamp	Recency	Frequency	Retrieval Time	Size	Predicted Output
1	1082348905.73	900	1	53	43907	1 (cacheable)
2	1082348907.41	900	1	703	14179	1 (cacheable)
3	1082348908.47	900	1	284	1276	0 (un-cacheable)
4	1082349578.75	900	1	263	25812	0 (un-cacheable)
1	1082349661.61	900	2	71	43097	0 (un-cacheable)
5	1082349675.35	900	1	203	8592	0 (un-cacheable)
3	1082349688.90	900	2	231	24196	1 (cacheable)
4	1082349653.72	900	2	875	25812	1 (cacheable)

6. Performance Evaluation

6.1. Classifier Evaluation

Precision and Recall are the performance parameters suited in many machine learning applications for measuring how accurately the data sets are classified. These two parameters Precision and Recall can be measured based on the result obtained from the confusion matrix[10], which is generated by the TANB classifier. In Table 5. The performance metrics are defined as follows.

Table 5. Performance Metrics Used in Machine Learning Classifier.

Metric	Description	Formula
Precision (p)	Precision is the ratio of True Positive class data divided the True positive and false positive class.	$\text{Precision} = \frac{TP}{TP + FP}$
Recall (r)	Recall is the ratio of True Positive class data divided the True positive and false Negative class.	$\text{Recall} = \frac{TP}{TP + FN}$
Confusion Matrix Notation:	TP-True Positive. FP-False Positive. FN-False Negative.	

From the below given graph in Fig. 3. It is known that the classification accuracy of Precision and Recall of TANB is higher than other classifier like NB and it is far better than ID3 and BPNN. In summation to that the computational time for training TANB and NB is faster than ID3 and BPNN for all Data sets in given below in Table 6. So we can conclude that performance of accuracy of TANB in classifying the proxy log datasets is more valuable and efficient with other supervised machine learning algorithms[11].

Table 6. Training Time for classifying the log data sets.

Datasets	TANB	NB	ID3	BPNN
Bo2	0.13	0.17	0.12	0.39
NY	0.31	0.38	2.00	0.85
UC	0.43	0.61	0.56	1.01
SV	0.49	0.55	0.21	0.69
SD	0.55	1.12	0.65	2.90
AVG	0.566	0.542	0.708	1.168

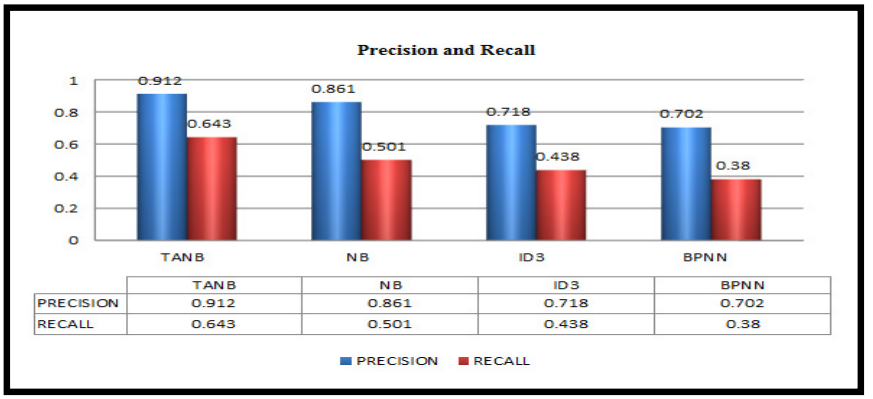

Fig. 3. Comparison of Recall and Precision.

6.2. Evaluation of Web Proxy Caching

6.2.1 Implementation

The Implementation of web proxy log file can be generated by WebTraff[12] simulator which is used to evaluate the performance metrics of the cache replacement algorithm. This tool is written in C++ and TCL scripts for the Generating the Web proxy workload which comes under with two process called ProwGen trace and Web proxy simulation.

6.2.2 Performance Measures

The overall performance of the cache replacement algorithm is measured in terms of cache hit ratio and byte hit ratio given in Table 7.

Table 7. Examples of Performance metrics used in Cache Replacement Policies.

Metric	Description	Formula
Cache Hit Ratio (HR)[8]	Cache Hit Ratio the number of requests satisfied from the proxy cache as a percentage of the total Request.	$\frac{\sum_{d \in D} cr_d}{\sum_{d \in D} r_d}$
Byte Hit Ratio (BHR) [8]	Byte Hit Ratio (weighted ratio) the amount of byte transfer from the proxy cache as a percentage of total number of bytes for the entire request	$\frac{\sum_{d \in D} s_d \cdot cr_d}{\sum_{d \in D} s_d \cdot r_d}$

From, the given graph below in Fig. 4. Specify that integrated TANB-GDSF increases GDSF performance in terms of cache Hit Ratio and TANB-LRU increases over LRU in terms of Byte Hit Ratio.

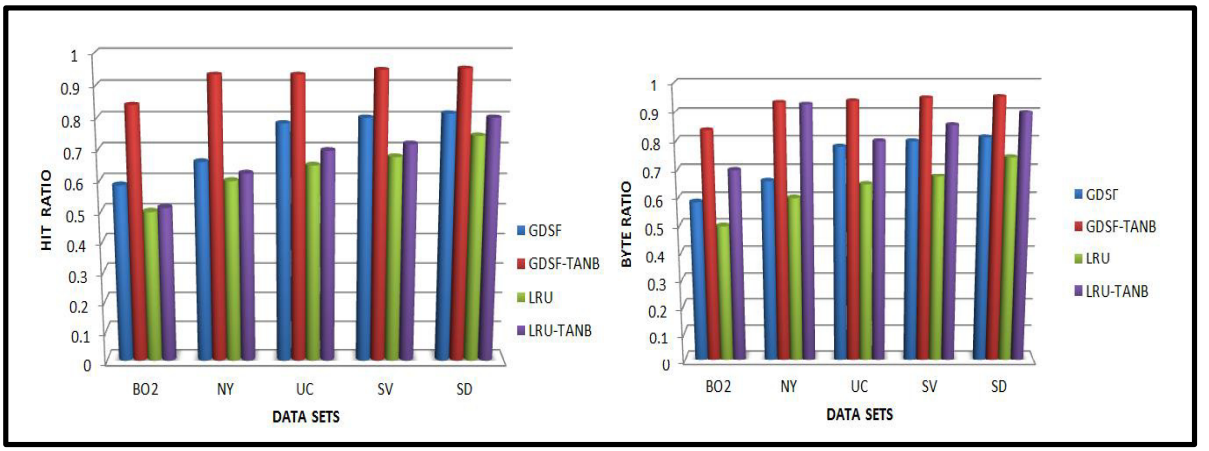

Fig. 4. Cache Hit Ratio and Byte Hit ratio for Different Data Set.

7. Conclusion

This work proposes by implementing a new approach, namely TANB-LRU and TANB-GDSF for improving the overall performance of the cache replacement of algorithms. Experimental results have revealed that TANB achieves much better Precision compared with other classifiers. In addition, in future we can consider incorporating the classification technique in the surrogate server for improving the performance of the Content Distribution network.

References

1. Waleed Ali, Siti Mariyam Shamsuddin, Abdul Samad Ismail. Intelligent Naïve Bayes-based approaches for Web proxy Caching. *Knowledge-Based System* 2012;**31**:162-175.
2. Romano S, ElAarag H. Neural network proxy Cache replacement strategy and its implementation in the squid proxy server. *Neural computing and Applications* 2011;**20**:59-78.
3. Ali Ahmed W, Shamsuddin S. Neuro-Fuzzy System in Partitioned client side web cache. *Expert System with Application* 2011;**38**:14715-14725.
4. Kumar C, Norris J.B. A New approach for a proxy-level web caching mechanism. *Decision Support System* 2008;**46**:52-60.
5. Jake Cobb , Hala ElAarag. Web Proxy Cache replacement scheme based on back-propagation neural network. *The Journal of systems and software* 2008;**81**:1539-1558.
6. Podlipnig S, Boszormenyi L. A survey of web cache replacement strategies. *ACM Computing Surveys* 2003;**35**:374-398.
7. Friedman N, Geiger D, Goldszmidt M. Bayesian Network classifier. *Machine Learning* 1997;**29**:131-163.
8. Kin-Yeung W. Web Cache replacement policies a pragmatic approach. *IEEE Network* 2006;**20**:28-34.
9. Liu B. *Web Data Mining*. 2nd ed. Springer Heidelberg Dordrecht London New York: Springer; 2011.
10. Han J, Kamber M. *Data Mining concepts and Techniques*. 3rd ed. Morgan Kaufmann; 1979.
11. Mitchell T. *Machine Learning*. 1st ed. McGraw-Hill; 1997.
12. Markatchev N, Williamson C. Web Traff a GUI for Web Proxy Cache workload Modeling and analysis. Inc:Proc.10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication systems, *IEEE Computer Society*; 20002. p. 356-363.
13. KEEL Data-Mining Software Tool available in; *http://www.Keel.es*.
14. NLANR, National Lab of Applied Network Research and Sanitized Access Logs*; http://www.ircache.net/2010*.