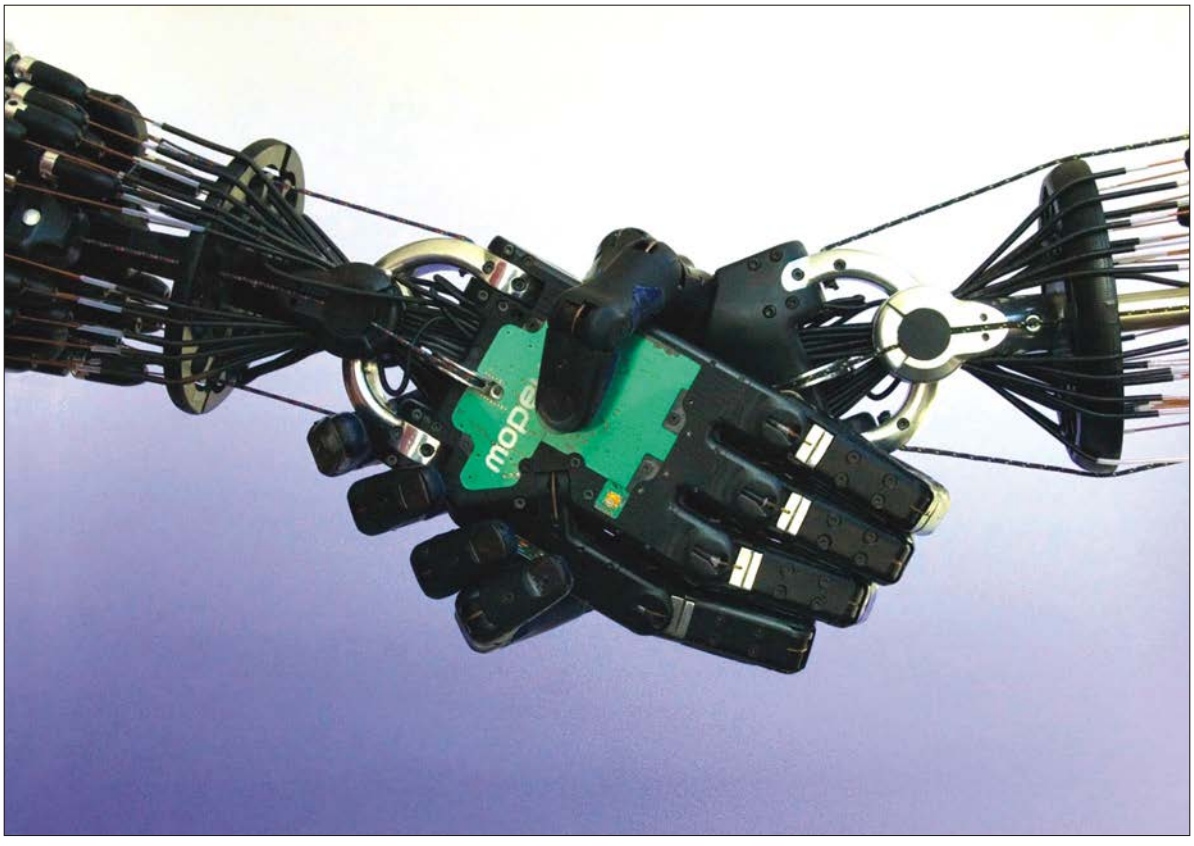

Journal of Artificial Intelligence

ISSN 1994-5450

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Artificial Intelligence 4 (1): 100-109, 2011
ISSN 1994-5450 / DOI: 10.3923/jai.2011.100-109

Web Proxy Cache Content Classification based on Support Vector Machine

W. Ali, S.M. Shamsuddin and A.S. Ismail
Soft Computing Research Group, Faculty of Computer Science and Information System, Universiti Teknologi Malaysia, 81310, Johor, Malaysia

Corresponding Author: Waleed Ali, Soft Computing Research Group, Faculty of Computer Science and Information System, Universiti Teknologi Malaysia, 81310, Johor, Malaysia

ABSTRACT

Web proxy caching plays a key role in improving the world wide web performance. However, the difficulty in determining which web objects will be re-visited in the future is still a big problem faced by existing web proxy caching techniques. In this study, we present a new approach which depends on the capability of support vector machine to learn from web proxy log data and predict the classes of objects to be re-visited. Therefore, usage of the cache can be optimized efficiently. Experimental results have revealed that the support vector machine produces similar correct classification rate compared to neuro-fuzzy system. However, the support vector machine achieves much better true positive rate and performs much faster than neuro-fuzzy system for both training and testing in several datasets.

Key words: Web caching, proxy server, logs file, classification, support vector machine

INTRODUCTION

The world wide web (Web) is the most common and most important service on the Internet. The web contributes greatly to our life. The web has become a useful tool in many fields such as education, entertainment, medicine, remote shopping and downloading of any software. These indications illustrate the rapid growth of Internet users that results in an explosive increase in traffic or bottleneck over the Internet performance (Tian *et al*., 2002). In other words, the web has become a victim of its own success. In the end, Internet users experience slow response time, especially for popular web sites.

The most popular software based solution is web caching technique. The web caching is introduced at three levels: client level, proxy level and original server level (Chen, 2007). Successfully, proxy servers play the key roles between users and web sites, which could reduce the response time of user requests and save network bandwidth. Therefore, an efficient caching approach should be built in a proxy server for achieving better response time.

Due to cache space limitation, an intelligent way is required to efficiently manage the web cache content. The conventional caching policies are not efficient since they consider one factor and ignore other factors that have impact on the efficiency of the web caching (Koskela *et al*., 2003). Therefore, many web caching policies have been proposed for getting better performance. However, combination of these factors to predict re-visiting of web objects is not an easy task because one factor in a particular environment is more important than other factors in another environment (Wong, 2006).

Few researchers have proposed incorporating intelligent solutions to cope with the above problem. According to Tian *et al.* (2002), the intelligent approaches are more efficient and more adaptive to Web caching environment compared to others approaches. Availability of Web access log files, i.e., history of accesses, which represent complete and prior knowledge of future access is the main motivation for adopting an intelligent approach in Web caching. The Web access log files can be utilized as training data to effectively classify the next set of web objects.

Support Vector Machine (SVM) is one type of supervised learning method, which has many desirable qualities that make it one of most popular algorithms. It performs classification more accurate and faster than most other algorithms in many applications such as text classification, Web page classification and bioinformatics applications (Liu, 2007; Chen and Hsih, 2006).

In this study, SVM is proposed to predict the Web objects that can be re-accessed later. This can help in storing the ideal web objects for more effective usage of proxy cache space.

In our earlier study ICWCS (Ali and Shamsudin, 2009), we have shown that neuro-fuzzy system can be used for classifying web objects into either cacheable or uncacheable objects. In ICWCS, the neuro-fuzzy system has been employed to predict web objects that can be re-accessed later. Sulaiman *et al.* (2008) have suggested using Particle Swarm Optimization (PSO) for improving neural network performance. Then, the improved neural network is used in classifying web objects. ANN has also been used by Koskela *et al.* (2003) depending on syntactic features from HTML structure of the document and the HTTP responses of the server as inputs. However, this method ignored frequency factor. On other hand, it hinged on some factors that do not affect on performance of the web caching.

In NNPCR (Cobb and El-Aarag, 2008) and NNPCR-2 (El-Aarag and Romano, 2009), ANN has been used for making cache replacement decision. An object is selected for replacement based on the rating returned by ANN. However; the factor of object type is ignored. Foong *et al.* (1999) proposed a Logistic Regression (LR) model to predict the future request. The web objects are classified into two categories; the object will be re-accessed in a given forward looking window (Y = 1), or not (Y = 0). In similar way, Tian *et al.* (2002) presented design of an intelligent predictor that uses back-propagation neural network instead of the LR model to predict the future access of web objects. However, he ignored the most important factor in Web caching, recency factor.

This study takes into consideration the most significant parameters such as frequency, recency, size and type of web object. Moreover, our study utilizes one of the most powerful classifiers, SVM, in predicting web objects that can be re-accessed later.

SUPPORT VECTOR MACHINE

Support Vector Machine (SVM) is the most popular machine learning model that invented by Vapnik (1995). SVM is one type of supervised learning method, which has many desirable qualities that make it one of most popular algorithms. It performs classification more accurate and faster than most other algorithms in many applications such as text classification, Web page classification and bioinformatics applications (Liu, 2007; Chen and Hsieh, 2006).

The primary idea of SVM is using a high dimension space to find a hyper plane to do binary division with two classes, positive and negative samples, where the achieved error rate is minimum. In training phase, the SVM is trained to find several support vectors that represent training data. These support vectors will be formed into a model by the SVM, representing a category. Consequently, the SVM will classify a given unknown data set depending on this model.

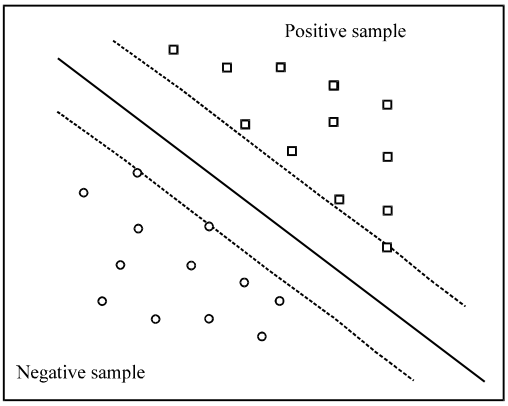

Fig. 1: The hyper plane of SVM

Let, the set of training examples D be $\{(x_1, y_1), (x_2, y_2),...,(x_n, y_n)\}$ and $x_i = (x_{i1}, x_{i2},...,x_{ir})$ is a r-dimensional input vector in a real-valued space $X \subseteq R^r$, y_i is its class label (output value) and $y_i \in \{1,-1\}$. 1 denotes the positive class and -1 denotes the negative class. In a linear problem, a high dimension space divided into two categories by a hyper plane as shown in Fig. 1. To build a classifier, SVM finds the hyper plane formula of the form Eq. 1.

$$f(x) = < w.x > + b \tag{1}$$

So, that an input vector x_i is assigned to the positive class if $f(x_i) \geq 0$ and the negative class otherwise, i.e.,

$$y_i = \begin{cases} 1 & \text{if } < w.x_i > + b \geq 0 \\ -1 & \text{if } < w.x_i > + b < 0 \end{cases} \tag{2}$$

However, for many real-life problems, it is not easy to find a hyper plane to classify the data such as nonlinearly separable data. To deal with nonlinearly separable data, the same formulation and solution techniques as for the linear case are still used. The input data is only transform from its original space into another space (usually of a much higher dimensional space) so that a linear decision boundary can separate positive and negative examples in the transformed space, which is called the feature space. The original data space is called the input space. Thus, the basic idea is to map the data in the input space X to a feature space F via a nonlinear mapping, i.e.:

$$\varnothing : X \to F, x \to \varnothing(x)$$

The same linear SVM solution method is then applied to F. In the transformed feature space (Fig. 2), they can be separated linearly.

As mentioned earlier, nonlinear decision boundaries are found via a transformation of the original data to a much higher dimensional feature space. However, this transformation is never explicitly done. Instead, kernel functions are used to compute dot products required in learning without the need to even know the transformation function. The SVM has several kernel functions

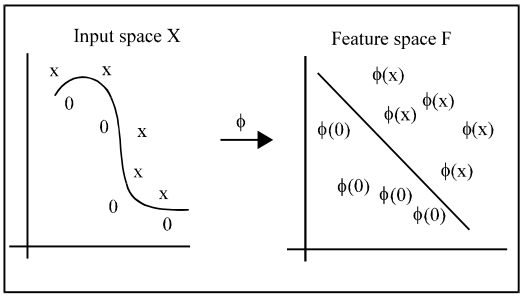

Fig. 2: Transformation from input space to feature space

Table 1: The most common kernel functions

Kernel	Kernel function
Linear	$k(x_i, x_j) = x_i^T x_j$
Polynomial	$k(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$
Sigmoid	$k(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$
Radial basis function (RBF)	$k(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \gamma > 0$

that users can apply to solve different problems. Selecting the appropriate kernel function can solve the problem of linear inseparability. Table 1 lists four kernel functions that are often used. The different kernel functions are suited to different problem types.

THE PROPOSED CLASSIFICATION APPROACH BASED ON SUPPORT VECTOR MACHINE

Here, we present a framework of the proposed approach for classification Web proxy cache content based on support vector machine (Fig. 3). As shown in Fig. 3, the proposed approach is achieved through several steps: data collection, preprocessing and training phase. Consequently, the web cache can be managed based on the trained SVM classifier.

Data collection: The web proxy log files provide information about activities performed by the users during the moment the user logs in to servers. The web proxy log files can be obtained from proxy servers located in some organizations or universities (Wessels, 2001). The web proxy logs files are considered complete and prior knowledge and can be utilized as training data to predict effectively of next web objects. An access proxy log entry usually consists of the ten following fields:
Timestamp, elapsed time, client address, Log tag and HTTP code, size, request method, URL, user identification, hierarchy data and hostname and content type.

Data preprocessing: The data should be undergone a certain pre-processing to become suitable for training phase. In this study, the data preprocessing involves the following two steps: trace preparation and dataset preparation.

In the trace preparation, irrelevant requests are removed from the log files since some the log entries are not valid or irrelevant entries. The trace preparation is carried out as follows:

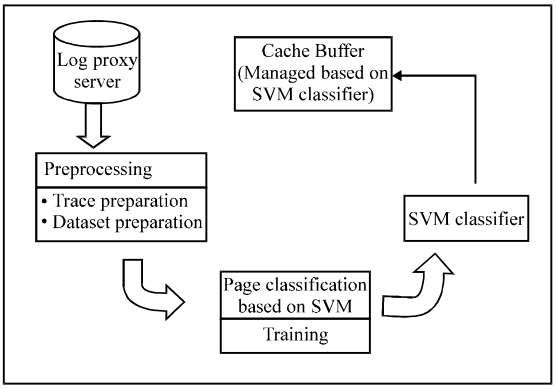

Fig. 3: A proposed framework for classification Web proxy cache content based on support vector machine

- **Parsing:** This involves identifying the boundaries between successive records in logs file, as well as the distinct fields within each record.
- **Filtering:** This includes elimination of irrelevant entries like: The uncacheable requests (i.e., queries with ? in the URLs and cgi-bin requests) and entries with unsuccessful HTTP status codes. We only consider successful entries with 200 as status codes
- **Finalizing:** This involves removing unnecessary fields. Thus, the data are finalized into the final format that is amenable for experiment

In order to prepare the dataset of web objects, the desired features of web objects are extracted from trace and logs files. The important features of web objects that indicate user interest are selected as training dataset. The input features consist of timestamp, frequency, size and type of object. Subsequently, these features are converted to the input/output dataset or training patterns required for training of SVM. The training pattern takes the format: $< x_1, x_2, x_3, x_4, y >$. x_1 is recency of web object based on sliding window, x_2 is frequency of Web object based sliding window, x_3 is size of object, x_4 is type of object and y is target output of object request.

x_1 and x_2 are extracted based on sliding window as suggested by El-Aarag and Romano (2009). The sliding window of a request is the time before and after when the request was made. In other words, the sliding window should be around the mean time that an object generally stays in a cache.

In similar way to Foong *et al.* (1999), the type of the object, x_4, is classified into 5 categories: HTML with value 1, image with value 2, audio with value 3, video with value 4, application with value 5 and others with value 0. The value of y is assigned to 1 if the object is re-requested again in the forward-looking sliding window. Otherwise, the target output is assigned to 0. Subsequently, the dataset is normalized accordingly into range [0, 1] by applying Eq. 3.

$$\bar{v} = \frac{v - \min_V}{\max_V - \min_V} \quad (3)$$

where, v is the original value of attribute, $\bar{v}$ is the normalized value of attribute and $\max_V$ and $\min_V$ is the maximum and minimum values of attribute V.

Training phase: The Support Vector Machine (SVM) has been widely used to solve the problems of classification. With the help of kernel learning, SVMs have recently achieved many successful applications to various domains.

In this study, SVM is trained depending on the prepared dataset of object objects to classify the web objects into objects will be re-accessed or not. As mentioned in Table 1, several kernel functions like polynomial, sigmoid and RBF can be used in SVM. However, in this study, RBF kernel is used since it can achieve a better performance compared to others kernel functions (Hsu *et al*., 2009). Depending on recommendations of Hsu *et al*. (2009), SVM is trained as follows:

- Prepare and normalize the dataset
- Consider the RBF kernel
- Use cross-validation to find the best parameters
- Use the best parameters to train the whole training dataset
- Test

Subsequently, SVM classifier can be utilized in improving of performance of web caching fields such as web cache admission, replacement or pre-fetching.

IMPLEMENTATION AND EXPERIMENTAL RESULTS

We have obtained data of the proxy traces of web objects requested in several proxy servers located around the United States of the IRCache network. In this study, five traces or datasets are used to test and evaluate the proposed approach as shown in Table 2. Each dataset is obtained just during 4 h on 9 January 2007. Subsequently, the dataset is divided randomly into training data (70%) and testing data (30%). As we have explained earlier, the raw dataset are prepared properly into a suitable form as shown in Table 3. Subsequently, the dataset is normalized accordingly into range [0, 1] by applying Eq. 3. In this study, a 30 min is used as the Sliding Window Length (SWL).

Once the dataset is prepared and normalized, we train SVM model using the libsvm library (Chang and Lin, 2001). The generalization capability of SVM can be controlled through a few parameters like the term C and the kernel parameter like RBF width γ. To decide which values to choose for parameter C and γ, a grid search algorithm is implemented as suggested by Hsu *et al*. (2009). We keep the parameters that obtain the best accuracy using a 10 fold cross validation on the training set. Finally when the optimal parameters are found, a SVM model is trained depending the optimal parameters to predict and classify the web objects into objects will be re-accessed or not.

In order to benchmark SVM performance, SVM are compared to neuro-fuzzy system (ANFIS) as described in our previous work ICWCS (Ali and Shamsuddin, 2009). Table 4 summarizes the parameters setting for ANFIS training.

Table 2: Trace files and the name of proxy that log files came from

Trace file name	Proxy server name	Location
uc.sanitized-access.20070109,	uc.us.ircache.net	Urbana-Champaign, Illinois
bo2.sanitized-access.20070109	bo.us.ircache.net	Boulder, Colorado
sv.sanitized-access.20070109	sv.us.ircache.net	Silicon Valley, California (FIX-West)
sd.sanitized-access.20070109	sd.us.ircache.net	San Diego, California
sj.sanitized-access.20070109	sj.us.ircache.net	MAE-West San Jose, California

Table 3: Example of dataset of web objects

	Inputs			
Target	Type	Size	Frequency	Recency
0	1	2884	1	1800
1	1	25792	2	1800
1	2	2034	1	1800
0	5	2492	1	1800
1	1	25792	3	1800
0	2	756	1	1800
1	1	25792	4	1800
0	2	9015	1	1800
1	1	25792	5	1800
0	2	2228	1	1800
0	2	3834	1	1800
1	1	513	2	1800
0	1	23261	1	1800
0	2	5275	5	1800
0	2	4323	1	2212
0	2	1362	1	2848
1	2	4349	1	3813
0	2	2338	3	1800
0	2	2981	1	4694
0	2	8021	1	1800
0	2	2413	1	8717
0	1	933	1	1800

Table 4: Parameters setting for ANFIS training

ANFIS parameter	Value
Type of input Member function(MF)	Bell function
No. of MFs	2 for each input
Type of output MF	Linear
Training method	Hybrid
No. of linear parameters	80
No. of nonlinear parameters	24
Total No. of parameters	104
No. of fuzzy rules	16
Error tolerance	0.005

PERFORMANCE EVALUATION AND DISCUSSION

Correct Classification Rate (CCR) is a measure for classification accuracy. However, CCR alone is insufficient for measuring the classification accuracy, especially if the data is imbalanced. Since most web objects are visited just one time by user, the negative class represents the majority class. On the contrary, the positive class represents the minority class, which includes the objects that will be re-accessed in the near future and the more important class in web caching. Therefore, True Positive Rate (TPR) or sensitivity, True Negative Rate (TNR) or specificity and geometric mean (Gmean) are also used in this study for evaluating SVM performance as given in Table 5 and 6.

Table 7 shows comparison of CCR of SVM and ANFIS for different dataset in both training and testing data. As can be seen, both of SVM and ANFIS produce good performance.

Table 5: Confusion matrix

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

Table 6: The most common measures

Measure name	Formula
Correct classification rate	$CCR = \frac{\#correctely\ classified\ examples}{\#totale\ examples}$
True positive rate	$TPR = \frac{TP}{TP + FN}$
True negative rate	$TNR = \frac{TN}{TN + FP}$
Geometric mean	$G_{mean} = \sqrt{TPR * TNR}$

Table 7: The correct classification rate for different datasets

	CCR of training dataset		CCR of testing dataset	
Dataset	SVM	ANFIS	SVM	ANFIS
1uc	0.9046	0.9238	0.8986	0.9186
1sj	0.8645	0.8590	0.8574	0.8594
1sv	0.8728	0.8760	0.86	0.8746
1sd	0.9490	0.9262	0.94	0.9279
1bo2	0.9388	0.9570	0.9304	0.9587
Average	0.9059	0.9084	0.8973	0.9078

Table 8: TPR,TNR and Gmean of training data

	TPR		TNR		Gmean	
Dataset	SVM	ANFIS	SVM	ANFIS	SVM	ANFIS
1uc	0.8300	0.7079	0.9250	0.9828	0.8762	0.8341
1sj	0.7925	0.5912	0.8917	0.9598	0.8406	0.7533
1sv	0.7919	0.6813	0.9082	0.9613	0.8481	0.8093
1sd	0.7667	0.5526	0.9751	0.9996	0.8647	0.7432
1bo2	0.7089	0.4840	0.9572	0.9950	0.8238	0.6940
Average	0.778	0.6034	0.9314	0.9797	0.8507	0.7668

The averages of CCR of SVM are 0.9059 and 0.8973, while the averages of CCR of ANFIS are 0.9084 and 0.9078 for both training and testing data respectively.

As it can be observed from Table 7, the SVM produces similar CCR or slightly smaller CCR compared to ANFIS in different datasets. However, in terms of TPR or sensitivity, Table 8 and 9 clearly indicate that the SVM achieves much better TPR and Gmean for both training and testing data in all datasets. This is mainly due to the capability of SVM to predict the positive or minority class that includes the objects that will be re-accessed in the near future.

In addition to above measures, the computation time for training both SVM and ANFIS is calculated on the same computer for different datasets as shown in Table 10. As expected, SVM is much faster than ANFIS in all datasets. Thus, SVM is more suitable than ANN if the training is done online.

Table 9: TPR,TNR and Gmean of testing data

	TPR		TNR		Gmean	
Dataset	SVM	ANFIS	SVM	ANFIS	SVM	ANFIS
1uc	0.8074	0.6815	0.9228	0.9816	0.8632	0.8179
1sj	0.7817	0.5915	0.8849	0.9571	0.8317	0.7524
1sv	0.7909	0.6939	0.8919	0.9578	0.8399	0.8153
1sd	0.7638	0.5525	0.9733	0.9988	0.8622	0.7429
1bo2	0.7011	o.5082	0.9484	0.9940	0.8154	0.7107
Average	0.7690	0.6055	0.9243	0.9778	0.8425	0.7678

Table 10: The computational time (in sec) for training SVM and ANN

	Training time (sec)	
Dataset	SVM	ANFIS
1uc	39.03	2020.89
1sj	40.78	2022.76
1sv	37.09	1889.24
1sd	6.83	1643.48
1bo2	10.73	1638.22

CONCLUSION AND FUTURE WORKS

Web caching is one of the effective solutions to avoid Web service bottleneck, reduce traffic over the Internet and improve scalability of the Web system. This study proposes intelligent scheme based on support vector machine to predict the Web objects that can be re-accessed later. Thus, usage of the proxy cache can be optimized efficiently. Experimental results have revealed that the support vector machine has good performance and much faster than neuro-fuzzy system (ANFIS).

Our approach can be integrated into the most common replacement policies in web caching such LRU, LFU, GDS and GDSF. Moreover, the proposed approach can be utilized efficiently to guide the web prefetching systems.

ACKNOWLEDGMENTS

This study is supported by Ministry of Science, Technology and Innovation (MOSTI) under science Research Grant Scheme (VOT 79311). Authors would like to thank Research Management Centre (RMC), University Technology Malaysia, for the research activities and Soft Computing Research Group (SCRG) for the support and incisive comments in making this study a success.

REFERENCES

Ali, W. and S.M. Shamsuddin, 2009. Intelligent Client-Side Web Caching Scheme Based on Least recently Used Algorithm and Neuro-Fuzzy System. In: Advances in Neural Networks-ISNN 2009, Yu, W., H. He and N. Zang, (Eds.). Vol. 5552, Springer-Verlag, Berlin Heidelberg, pp: 70-79.

Chang, C.C. and C.J. Lin, 2001. LIBSVM: A library for support vector machines. http://www.csie.ntu.edu. tw/~cjlin/libsvm/.

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

Chen, T., 2007. Obtaining the optimal cache document replacement policy for the caching system of an EC Website. Eur. J. Operat. Res., 181: 828-841.

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

El-Aarag, H. and S. Romano, 2009. Improvement of the neural network proxy cache replacement strategy. Proceedings of the 2009 Spring Simulation Multiconference, (SSM'09), San Diego, California, pp: 90-90.

Foong, A.P., Y.H. Hu and D.M. Heisey, 1999. Logistic regression in an adaptive web cache. IEEE Internet Comput., 3: 27-36.

Hsu, C.W., C.C. Chang and C.J. Lin, 2009. A Practical Guide to Support Vector Classification. Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.

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

Liu, B., 2007. Web Data Mining: Exploring Hyperlinks, Contents and Usage Data (Data-Centric Systems and Applications). Springer, New York, pp: 532.

Sulaiman, S., S.M. Shamsuddin, F. Forkan and A. Abraham, 2008. Intelligent web caching using neurocomputing and particle swarm optimization algorithm. Proceedings of the 2nd Asia International Conference on Modelling and Simulation (AMS), (AICMS'08), Washington, DC., USA., pp: 642-647.

Tian, W., B. Choi and V.V. Phoha, 2002. An Adaptive Web Cache Access Predictor Using Neural Network. In: Developments in Applied Artificial Intelligence, Handtlass, T. and M. Ali (Eds.). Springer-Verlag, London, UK., pp: 450-459.

Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer, New York.

Wessels, L.D., 2001. Web Caching. O'Reilly, USA.

Wong, A.Y., 2006. Web cache replacement policies: A pragmatic approach. IEEE Network Maga., 20: 28-34.