Ensemble Prefetching Through Classification Using Support Vector Machine

Chithra D. Gracia and Sudha

Abstract Owing to the steadfast growth of the Internet web objects and its multiple types, the latency incurred by the clients to retrieve a web document is perceived to be higher. Web prefetching is a challenging yet achievable technique to reduce the thus perceived latency. It anticipates the objects that may be requested in future based on certain features and fetches them into cache before actual request is made. Therefore, to achieve higher cache hit rate group prefetching is better. According to this, classification of web objects as groups using features like relative popularity and time of request is intended. Classification is aimed using Support Vector Machine learning approach and its higher classification rate reveals effective grouping. Once classified, prefetching is performed. Experiments are carried out to study the prefetching performance through Markov model, ART1, linear SVM and multiclass SVM approach. Compared to other techniques, a maximum hit rate of 93.39% and 94.11% with OAO and OAA SVM multiclass approach is attained respectively**.** Higher hit rate exhibited by the multiclass Support Vector Machine demonstrates the efficacy of the proposal.

Keywords Prefetching · Classification · Machine learning · SVM · Hit rate · ART1

1 Introduction

Since, the World Wide Web (WWW) has grown to be well popular, services over the web and requirement for web objects by users has grown vividly [1]. Therefore, the latency incurred by the users in accessing the web objects over the web

C.D. Gracia
Department of Computer Science and Engineering,
National Institute of Technology, Tiruchirapalli, India
e-mail: chithragracia@gmail.com

Sudha(✉)
Department of Electrical and Electronics Engineering,
National Institute of Technology, Tiruchirapalli, India
e-mail: sudha@nitt.edu

S. Berretti et al. (eds.), *Intelligent Systems Technologies and Applications*,
Advances in Intelligent Systems and Computing 385,
DOI: 10.1007/978-3-319-23258-4_23

has also increased to a predominant issue that is to be met. Web prefetching is a promising and outstanding technique to reduce this latency. Prefetching as groups will produce more precise and accurate results. The challenge is to find a way to categorize this massive data in some meaningful structure. Prefetching in groups based on similarity measures can be obtained through data mining techniques like classification and clustering [2]. Classification is a supervised machine learning technique that groups related objects based on similarity of features or attributes. Support Vector Machine is a multivariate machine learning algorithm which supports classification.

Current literature gives subjective and descriptive ideas on prefetching and classification using SVM's. Most of the prefetching techniques relay on Markov model approaches since they are effective in predicting the next to be accessed page [3]. However, it not fast in adapting to the new patterns because the predictions are solely history based that is accessed during the previous time period. The effectiveness of using prefetching to resolve the problems in handling dynamic web pages is studied [4]. Yet, only the temporal properties of the dynamic web pages are explored.

Metadata prefetching based on relationship graphs with a significantly lesser hit rate of 70% is presented [5]. Semantic based prefetching relies on predicting the future requests based on semantics. The semantic preferences are exploited by analyzing the keywords in the URL anchor text or the documents that are previously accessed [6].

Clustering is an unsupervised data mining technique that groups data according to similarity such that intra data instances within a cluster are similar to each other and the inter data instances between clusters are much dissimilar. As told by Vakali.A. et. al, a wide range of web data clustering schemes present in the literature just focuses on clustering of inter-site and intra-site web pages[7]. Clustering can be based on statistical and evolutionary algorithms.

Classification, is assigning a class label to a set of unclassified cases. The main idea of classification and clustering lies in finding the hidden patterns in data. Support Vector Machines (SVM's) are a new generation learning system based on recent advances in statistical learning theory and are accurate with lesser training samples [8]. The tuning of the parameters of binary support vector machines in multiclass decomposition using genetic algorithm is performed [9]. SVM is used to predict the to be visited objects in order to optimize cache usage [10]. A Multiclass Support Vector Machine classifier approach in hypothyroid detection is performed [11]. However, all the aforementioned works focus on various applications other than prefetching. To our knowledge, no work is reported in literature that classifies the web objects through multiclass classification for prefetching.

SVM being a case-based classifier (non parametric) does not require any prior knowledge other than the training samples [12]. Moreover, training using SVM is faster with high generalization ability and avoids premature convergence. Taking these positive factors of SVM into consideration, an attempt is made to classify the web objects into classes to prefetch the related web objects of the class that the request corresponds to. To study the performance of classification, both binary and multi-class classification is intended. The following section explains in brief the classification techniques.

2 Overview of Binary and Multiclass SVM

The primary intuition behind SVM is to maximize the margin 'm' between the hyperplanes. Fig. 1 shows the hyperplanes and the margin between them. The idea is to find the function of the optimizing hyperplane between the classes.

2.1 Binary SVM

The purpose of SVM is to find the hyperplane that best separates the classes. If the number of classes is two, then binary classification is performed.

The hyperplane is characterized by the decision function [14]

$$f(x) = \mathrm{sgn}(< w, \Phi(x) > + b)$$

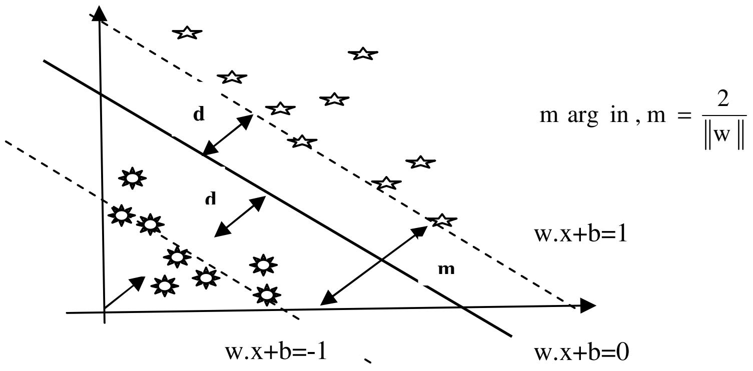


Fig. 1 Optimal separating hyperplane with maximum margin

where

w	- weight vector orthogonal to the hyperplane
x	- current input
$\Phi(x)$	- kernel transforming input data into feature space
b	- scalar that represent the margin of the hyperplane
<,>	- dot product
sgn	- signum function which extracts the sign of a number (i.e.,) returns 1 if value > 0 and -1 otherwise

If 'w' has unit length (norm=1), then $(< w, \Phi(x) >)$ is the length of $\Phi(x)$ along the direction of 'w'.'w' leads to largest 'b'. 'w' is scaled by $\|w\|$ to obtain a unit vector where, $\|w\| =< w, w >$.The normal vector 'w' that leads to largest 'b' of the hyperplane is to be found.

2.1.1 Linearly Separable Points

These are the points that can be linearly separated by a hyperplane in the input space itself. Let $x, y \in R^n$ where $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ The optimal separating hyperplane is found by [12]

$$\underset{w \in H, b \in R}{\text{maximize}} \min\{\|x - x_i\| \mid x \in H, <w, x> + b = 0, i = 1, ..., m\}$$

Margin $\frac{2}{\|w\|}$ is to be maximized. It is obtained by solving the objective function,

$$\underset{w \in H, b \in R}{\min \ \text{imize}} \quad \tau(w) = \frac{1}{2}\|w\|^2$$

subject to $y_i(<w, x_i> + b) \geq 1$ for all $i = 1, ..., m$

where ' τ ' is an objective function. The constraints ensure

$$f(x_i) = +1 \quad \text{for} \quad y_i = +1$$
$$f(x_i) = -1 \quad \text{for} \quad y_i = -1$$

To solve the optimization which is in primal form, it is converted to its dual form by applying the Lagrangian multipliers $\alpha_i \geq 0$, that leads to the dual optimization problem.

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i(<x_i, w> + b) - 1)$$

The Lagrangian 'L' must be maximized with respect to ' α_i ' and minimized with respect to primal variables 'w' and 'b'. At the saddle point, the partial derivatives of L with respect to primal variables must be 0. Hence,

$$w = \sum_{i=0}^{m} \alpha_i y_i x_i \ , \quad \sum_{i=0}^{m} \alpha_i y_i = 0$$

The training points with non zero ' α_i ' are called support vectors. As per Karush-Kuhn-Tucker condition (KKT), ' α_i ' that are non zero at the saddle point corresponds to the constraints that are precisely met. Hyperplane in dual optimization problem is

$$f(x) = \text{sgn}(\sum_{i=1}^{m} y_i \alpha_i < x_i, x > + b)$$

Where ‘b’ is computed by KKT conditions. When training errors are encountered, slack variables ’ ξ_i ’ is introduced and the objective function to be solved is

$$\tau(w,\xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i$$

subject to $y_i(<w, x_i> +b) \geq 1-\xi$ for all $i=1,...,m$
where $0 \leq \alpha_i \leq C$. Finally, $\sum_{i=0}^{m}\alpha_i y_i = 0$

2.1.2 Non Linearly Separable Points

The points that are not separable linearly by a hyperplane in the input space are mapped to a higher dimensional feature space as in Fig. 2. The kernel function $'\Phi'$ implicitly maps the training data in the input space to a higher dimensional feature space and constructs a separating hyperplane in the feature space. It is given by the kernel trick $K(x,x') =< x, x' >$. The decision function obtained [14] is

$$f(x) = \text{sgn}(\sum_{i=1}^{m} y_i\alpha_i K(x,x') + b)$$

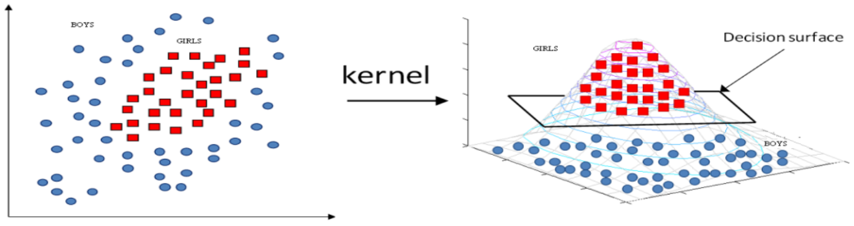


Fig. 2 Kernel mapping from input space to feature space

2.2 Multiclass SVM

If the number of labels is greater than two, then multiclass classification is performed. In the One Against All (OAA) approach, to have a classification of ‘M’ classes, a set of binary classifiers is constructed where each training separates one class from the rest. The output obtained will be the maximum of all the decision functions that is obtained [14] by

$$\underset{j=1,...,M}{\text{argmax}}\ g^j(x) \quad \text{where} \quad g^j(x) = \sum_{i=1}^{m} y_i\alpha_i^j K(x,x_i) + b^j$$

and the decision function is $f^j(x) = \text{sgn}(g^j(x))$. This approach computes ‘M’ hyperplanes.

In the One Against One (OAO) approach, pairwise classification is performed. Two classes are chosen and a hyperplane is constructed between them. This process is repeated for each pair of classes in the training set. The output is the decision function of the hyperplane with lesser number of support vectors.

3 Proposed Work

The objective of the proposal is to minimize the user perceived latency (client) while accessing the web server. Prefetching using lists tend to consume more memory with low hit rates, because the list is large in size. Especially, with the current Internet supporting multiple types of web objects like audio, video, images and text files creating lists would worsen this further. So, it will be ideal if the web objects are classified into groups which would minimize the group size. Minimizing the group size accounts to higher hit rate as only the objects related to a specific group are prefetched. As grouping is possible through classification, classification of multiple web objects is attempted through Support Vector Machine. The proposal comprises of three phases namely; preprocessing of web logs, classification and prefetching. Fig. 3 depicts the various tasks carried out in each of the phases and the co-ordination between them. Each of these phases is dealt briefly in the succeeding section.

Phase I: Data Preprocessing

The access logs consists of ten fields namely; timestamp, elapsed time, client's IP address, log tag with HTTP code, type of request(method), URI, user identification, hierarchy data and host name and content type. The log is pre-processed to extract the necessary fields. Preprocessing of data involves data formatting, filtering and extracting required fields in a format suitable for classification. As the log files are unstructured, identification of the various fields is carried out through data formatting.Data filtering involves elimination of entries that have unsuccessful HTTP status codes. Only URL's with status code 200 are considered. Requests with question marks in URL's, cgi-bin is discarded. As the data extracted is free of labels, it is to be presented in a form suitable for labelling. So, it is displayed in the form <*a1 a2 a3*> where *a1* denotes the requested URL (web object), *a2* the time of request and *a3* the access count. An example of preprocessed data is shown in Table 1.

Table 1 Preprocessed log file

URL_Request	***Timestamp***	***Access count***
/academics/departments/ece/programmes/btech/curriculum/	0.125000	13
/home/students/facilitiesnservices/hostelsnmess/hostels/	0.541666	2

Phase II: Classification and Training

The SVM uses (i) training data to generate the learned model (classifier) and (ii) when test data is given as input to the learned model, it classifies the data. Labelling is performed based on the relative popularity of URL's and their timestamp.The relative popularity (RP_i) of the 'ith' URL is calculated using Eqn. 1. Then the weighted average, w_{avg} of all URL's is found using Eqn. 2. It is the weighted average of the relative popularity of all unique URL's. This measure is used as a threshold to decide the URL's that are to be prefetched. This avoids prefetching of all objects corresponding to the entire class. By this only the URL's whose access count is above the threshold (those URL's in the class that are more frequently accessed) are prefetched to achieve high hit rate.

$$\text{Relative Popularity of a URL}(RP_i) = \frac{\text{Number of accesses to the 'i}^{th}\text{' URL}}{\text{Highest access count in the log}} \tag{1}$$

$$\text{Weighted Average, } w_{avg} = \frac{\sum_{i=1}^{N} RP_i * w_i}{\sum_{i=1}^{N} w_i} \tag{2}$$

Binary classification is done by taking the access count of the URLs into consideration and rules are framed to assign labels. The rules framed are as in Eqn.3.

$$\begin{array}{ll} \textit{Rule } 1: & IF\ (RP_i > w_{avg})\ THEN\ \ URL_i \leftarrow label\ 1 \\ \textit{Rule } 2: & IF\ (RP_i < w_{avg})\ THEN\ \ \mathrm{URL}_i \leftarrow label\ 2 \end{array} \tag{3}$$

Based on these rules the URL's are labeled. To train the SVM the data is organized in the form <*x1 la* > where '*x1*' represent the relative popularity of the web object and 'la' denotes the label of the URL.

Further, multi classification is intended for which more labels are to be included. Hence, the feature set is extended to accommodate more rules and thereby more labels. So, timestamp at which the URL is accessed is also taken in addition to the Relative popularity. The rules framed are given in Eqn.4.

$$\begin{array}{ll} \text{Rule } 1: & \text{IF}(\text{RP}_i > \text{w}_{avg}) \text{ and time}_i = \text{recent time}_i \pm 2 \text{ THEN URL}_i \leftarrow \text{label } 1 \\ \text{Rule } 2: & \text{IF}(\text{RP}_i > \text{w}_{avg}) \text{ and time}_i <> \text{recent time}_i \pm 2 \text{ THEN URL}_i \leftarrow \text{label } 2 \\ \text{Rule } 3: & \text{IF}(\text{RP}_i < \text{w}_{avg}) \text{ and time}_i = \text{recent time}_i \pm 2 \text{ THEN URL}_i \leftarrow \text{label } 3 \\ \text{Rule } 4: & \text{IF}(\text{RP}_i < \text{w}_{avg}) \text{ and time}_i <> \text{recent time}_i \pm 2 \text{ THEN URL}_i \leftarrow \text{label4} \end{array} \tag{4}$$

where recent time$_i$ is the recently accessed time of the 'i'th URL and the constant '2' refers to 2 hours. Based on the rules framed above, the URL's are labeled. To train the SVM, the data is converted to the pattern < *la x1 x2* > where 'la' denotes the label of 'i'th URL, '*x1*' represents the relative popularity of 'i'th URL, *x2* denotes the time of request of 'i'th URL. A snippet of the data set which is to be trained is displayed in Fig. 4.

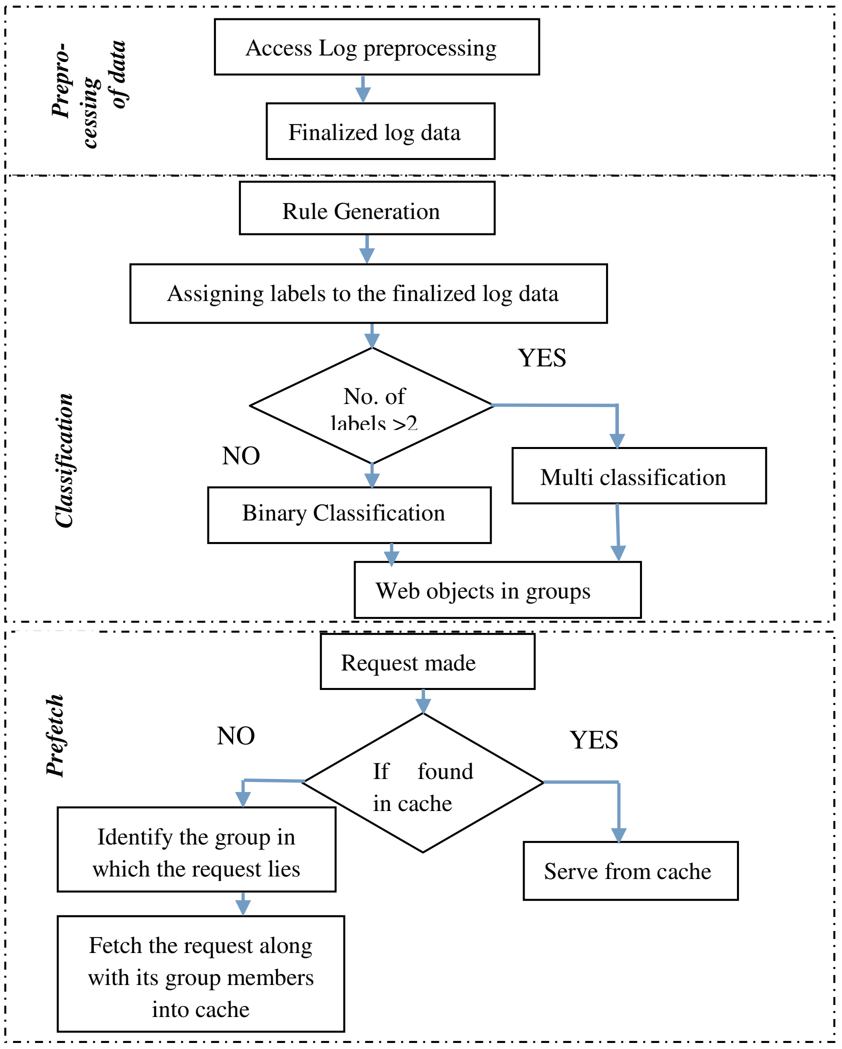
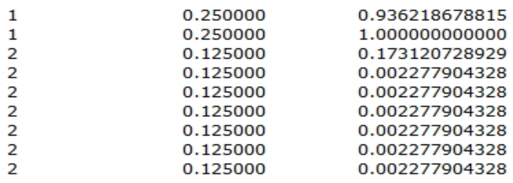


Fig. 3 Flowchart of the proposed model



1	0.250000	0.936218678815
1	0.250000	1.000000000000
2	0.125000	0.173120728929
2	0.125000	0.002277904328
2	0.125000	0.002277904328
2	0.125000	0.002277904328
2	0.125000	0.002277904328
2	0.125000	0.002277904328
2	0.125000	0.002277904328

Fig. 4 Sample of the log file features and their labels

Once the dataset is labeled, classification focusing both on binary and multiclass SVM's is performed.

Training with Binary SVM and Multi Class SVM
The binary classification is performed by separating the two classes with a linear optimal separating hyperplane that maximizes the margin. Further, when a new data is given, the data is classified based on the sign value of the hyperplane function which is discussed in section 2.

For the non-linear cases, the Gaussian RBF kernel (Radial Basis Function) kernel is employed since it is considered better than other kernel functions as it is localized and has finite response across the range of real x-axis [13]. The RBF kernel on two samples x and y is represented as feature vectors in some input space which is given by $K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$ where $\|x - y\|^2$ is recognized as the squared Euclidean distance between the two feature vectors. 'σ' is a free parameter. Two parameters namely the kernel parameter 'γ' where $\gamma = -\frac{1}{2\sigma^2}$ and cost parameter 'C' is to be set. By altering various values for C and'γ', the generalization capability of the SVM is controlled and the training is done. It is found that choice of kernel function and suitable value for the parameters of the kernel is essential for a given data to perform better classification. A combination of binary SVMs is involved in the multi-class classification using one-against-all (OAA) and compared against one-against-one (OAO) approach. In the OAA approach, the members that belong to a class are differentiated from the rest of the classes. To perform classification of 'M' classes, each training constructs a set of binary classifiers where each class separates one class from the others. Finally, all the classifiers are combined by multiclass approach. The output of this approach corresponds to the maximum obtained from all the decision functions.

Phase III: Prefetching
When the browser is idle, the classified data is subjected to prefetch in the cache. On making a request, the URL is searched in cache and if found, the URL is served from the cache. If not, the web object corresponding to the URL is obtained from the server along with its corresponding class members and placed in cache.

4 Results and Analysis

The hardware platform used for the implementation is a high end workstation with i7 processor configured as server running @3.40GHz and 16GB RAM. The access logs of NITT web server (nitt.edu) is taken for classification and a snapshot is displayed in Fig. 5.

A mirror of the NITT web server is obtained to carry out the experimentation. 30 log files (corresponding to 30 days) collected from 24th August 2014 to 23rd September 2014 is used for our experimentation. The cache size is 1024MB.

```
117.193.32.150 - - [24/Aug/2014:09:42:47 +0530] "GET
/home/academics/departments/civil/ HTTP/1.1" 200 14126 "-"
"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/27.0.1453.110 Safari/537.36"
117.193.32.150 - - [24/Aug/2014:09:42:48 +0530] "GET
http://www.nitt.edu/home/academics/departments/civil/HTTP/1.1"
304 -
"http://www.nitt.edu/home/students/facilitiesnservices/tp/"
"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/27.0.1453.110 Safari/537.36"
117.193.32.150 - - [24/Aug/2014:09:42:48 +0530] "GET
http://www.nitt.edu/home/academics/departments/cse/services/"
"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/27.0.1453.110 Safari/537.36"
117.199.127.189 - - [24/Aug/2014:09:42:47 +0530] "GET
/home/students/facilitiesnservices/hostelsnmess/hostels/
HTTP/1.1" 200 22655 "http://www.nitt.edu/home/" "Mozilla/5.0
(Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/27.0.1453.116 Safari/537.36"
117.193.32.150 - - [24/Aug/2014:09:42:48 +0530] "GET
/cms/templates/corporate-final/images/tooltip/phoneicon1.gif
HTTP/1.1" 304 -
"http://www.nitt.edu/home/students/facilitiesnservices/tp/"
"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/27.0.1453.110 Safari/537.36"
117.193.32.150 - - [24/Aug/2014:09:42:48 +0530] "GET
/cms/templates/corporate-final/images/tooltip/emailicon1.gif
HTTP/1.1" 304 -
"http://www.nitt.edu/home/students/facilitiesnservices/tp/"
"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/27.0.1453.110 Safari/537.36"
```

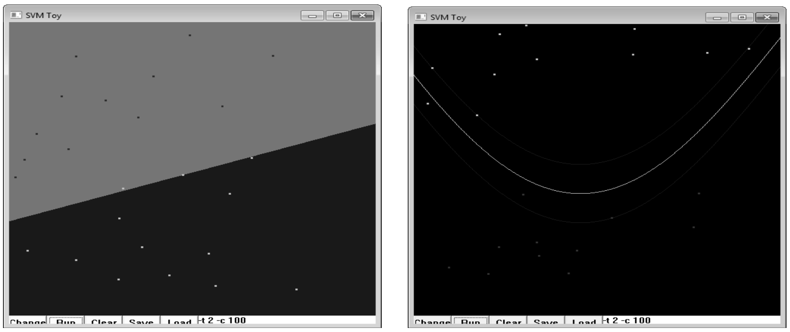
Fig. 5 Snippet of the log file for training

4.1 Classification Results

These 30 log files are divided into 2 log files namely log1 and log2 each corresponding to 15 days. Log1 is used for training the classifier. Log2 is further divided into dataset1 of size 650MB with 3891 URL accesses and dataset2 of 458MB with 2904 URL accesses to test the classifier. Classification is performed using svmtoy module of the LIBSVM 3.20 software implemented in MATLAB 2013a. Experiments are carried out to study the performance of ART1, binary and multiclass SVM's. Their results are summarized below.

Test Case 1: Binary Classification

Log1 is labeled using Eqn.3 and the labeled log file is used to train both the linear and non-linear SVM binary classifier. Dataset1 and Dataset2 is used to test the accuracy of the trained classifier. Fig. 6(a) & (b) shows the binary classification output of dataset1 for both linear and non linear SVM.



(a) (b)

Fig. 6 Classification with (a) Linear SVM (b) Non-Linear SVM

From the above results, the misclassification rate with linear SVM and non linear SVM for dataset1 is found to be 13.28% and 12.01% respectively.

Test Case 2: Multiclass Classification

Log1 is labeled using Eqn.4 and the labeled log file is used to train both the OAA and OAO SVM multiclass classifiers. Dataset1 and Dataset2 is used to test the accuracy of the trained multiclass classifiers. The output obtained for dataset1 through OAA and OAO classifiers is displayed in Fig. 7(a) & (b) respectively.

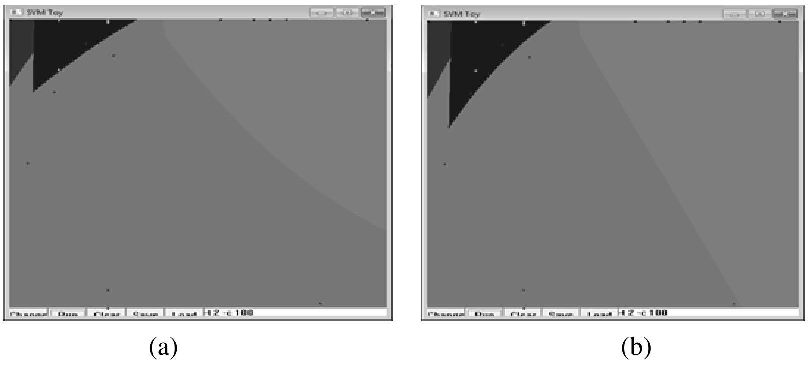


Fig. 7 Classification with (a) OAO SVM (b) OAO SVM

From the above results, the misclassification rate is found to be around 10.81% and 11.27% through OAA and OAO multiclass SVM models respectively. Similarly, classification with ART1 is also carried out. The classification rate obtained through various classification techniques with both datasets is tabulated in Table2.

Table 2 Classification rate through various techniques

Techniques		Dataset1 (%)	Dataset2 (%)
ART1 Network		85.45	86.25
Binary class	Linear SVM	86.23	86.21
	Non linear SVM	87.97	88.9
Multiclass	OAA	90.23	91.35
	OAO	89.23	90.00

From Table 2, the classification rate through multiclass SVM is inferred to be higher than the binary SVM. Further, of the multiclass classifiers OAA is found to be even much better than OAO confirming the efficacy of multiclass OAA SVM.

Further, to study the prefetching performance based on the requested web object prefetching by different methods such as Popularity Based Markov model, binary and multiclass SVM is performed. The hit rate obtained through these techniques using dataset1 and dataset2 is displayed in Table 3.

Table 3 Hit rate obtained by different techniques

Techniques		Dataset1 (%)	Dataset2 (%)
Popularity Based Markov Model		88.94	89.01
ART1 Network		85.11	86.29
Binary Class	Linear SVM	89.67	88.39
	Non linear SVM	92.22	92.91
Multiclass	OAA	94.07	94.11
	OAO	93.39	93.13

From Table 3, prefetching through multiclass SVM is found to yield higher hit rate than Binary SVM, Markov model and ART1. This is because of the history based and unclassified data of Markov, large group size of ART1 and limited classes of binary SVM. Whereas, OAA SVM multiclass approach gives higher hit rate than others because of (i) its high classification rate and (ii) group size limitation. Hence, it is concluded that prefetching through ensembled web objects achieve high hit rates and especially, with multi class OAA SVM.

5 Conclusion

Web prefetching minimizes latency. Literatures present various methods of prefetching including binary classification. But none of the work is reported using multiclass classification for ensemble prefetching. Hence the proposed work focuses on group prefetching of web objects through multi-class classification and subsequent prefetching. As the log files contain unsupervised data, they are initially converted to supervised data by assigning labels. Further, training is carried out to compare the classification efficiency using ART1 neural network, binary and multiclass SVM. Further, web object prefetching is performed using the classification and Markov models. Experimental results using OAA multi-class classification yielded higher classification rate than the other techniques. Moreover, higher hit rate is observed through prefetching based on OAA multi-class classification than ART1 neural network and Markov model approach. Thus, the proposed multi-class SVM classification is found to support prefetching effectively and efficiently.

References

1. Domenec, J., Gil, J.A., Sahuquillo, J., Pont, A.: Using current web page structure to improve prefetching performance. Computer Networks **54**, 1404–1417 (2010)
2. Liao, S.-H., Chu, P.-H., Hsiao, P.-Y.: Data mining techniques and applications – A decade review from 2000 to 2011. Expert Systems with Applications **39**(12), 11303–11311 (2012)

3. Chithra, D.G., Sudha, S.: MePPM- Memory efficient Prediction by Partial Match Model for Web Prefetching. In: 3rd IEEE International Advance Computing, Conference (IACC), pp. 736–740 (2013)
4. Lam, K.Y., Ngan, C.H.: Temporal Pre-Fetching of Dynamic Web Pages. Info. Systems Journal, Elsevier, 31149–31169 (2006)
5. Gu, P., Wang, J., Jiang, H.: A novel weighted-graph based grouping algorithm for metadata prefetching. IEEE Transactions on Computers 59(1), 1–14 (2010)
6. Alexander, P.P.: Semantic Prefetching Objects of Slower Web Site Pages. The Journal of System. and Software **79**, 1715–1724 (2006)
7. Vakali, A.I., Pokorný, J., Dalamagas, T.: An Overview of Web Data Clustering Practices. In: Lindner, W., Fischer, F., Türker, C., Tzitzikas, Y., Vakali, A.I. (eds.) EDBT 2004. LNCS, vol. 3268, pp. 597–606. Springer, Heidelberg (2004)
8. Mathur, A., Foody, G.M.: Multiclass and binary SVM classification: Implications for training and classification users. IEEE Geos. and Rem. Sens. Letters **5**(2), 241–245 (2008)
9. Lorena, A.C., de Carvalho, A.C.P.L.F.: Evolutionary tuning of SVM parameter values in multiclass problems. Neurocomputing **71**, 3326–3334 (2008)
10. Ali, W., Shamsuddin, S.M., Ismail, A.S.: Web Proxy Cache Content Classification based on Support Vector Machine. Journal of Artificial Intelligence **4**, 100–109 (2011)
11. Chamasemani, F.F., Singh, Y.P.: Multi-class Support Vector Machine (SVM) Classifiers–An Application in Hypothyroid Detection and Classification. In: 2011 Sixth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA). IEEE (2011)
12. Denœux, T., Smets, P.: Classification using belief functions: relationship between case-based and model-based approaches. IEEE Trans. System Man Cybernetic. Part B: Cybernetic. **36**(6), 1395–1406 (2006)
13. Arora, M., Kanjilal, U., Varshney, D.: Efficient and Intelligent Information Retrieval. International Journal of Soft Computing and Engineering (IJSCE) **1**(6), 39–43 (2012)
14. Morariu, D.: Classification and Clustering using SVM, Ph.D Report, University of Sibiu (2005)