SEMANTIC WEB PREFETCHING SCHEME USING NAÏVE BAYES CLASSIFIER

P.VENKETESH

Computer Science and Engineering department,
PSG College of Technology, Coimbatore, India
venkip_ms@yahoo.co.in

DR.R.VENKATESAN

Information Technology department,
PSG College of Technology, Coimbatore, India
ramanvenkatesan@yahoo.com

L.ARUNPRAKASH

Computer Science and Engineering department,
PSG College of Technology, Coimbatore, India
arunprakashl@gmail.com

Web prefetching is an effective technique to minimize user's web access latency. Web page content provides useful information for making the predictions that is used to perform prefetching of web objects. In this paper we propose semantic prefetching scheme that uses anchor texts present in the web page to make effective predictions. The scheme applies Naïve Bayes Classifier for computing the probability values of anchor texts, based on which the predictions are generated and given as input to the prefetch unit. Predictions are dynamic and are based on the browsing pattern exhibited by the user in a session. The browsing pattern of user should be specific towards a particular topic of interest to achieve good hit rate. When user has long browsing sessions, predictions and prefetching are more effective and helps to increase the hit rate and minimize access latency.

Keywords: Prefetching, Naïve Bayes classifier, Predictions, Anchor links

1. Introduction

The usage of Internet over the years has tremendously increased and the users are leveraging its benefits to access variety of services provided over this network. Due to the massive growth of Internet, network load and access time have increased dramatically causing substantial delay in providing services to the user. To overcome these limitations, web prefetching is considered as an effective solution where it focuses on minimizing the response time when web documents are retrieved. Web prefetching loads web documents into a web cache, which the user is likely to access in the near future. It requires predicting the list of web documents to be prefetched for satisfying the user requests.

Previous research on web prefetching focused on using the history of client access patterns to make predictions. The access patterns were represented using URL graph and based on graph traversal done by search algorithm the prefetching predictions are computed. These schemes suffer from the drawback of not able to prefetch web objects that are newly created or never visited before. To overcome these limitations, keyword based semantic prefetching approach [6, 16] was introduced. It could predict the web objects to satisfy the future requests based on semantic preferences of past retrieved web objects. Neural networks was trained using the keyword set to make predictions, and the research was motivated by the fact that the web user's surfing behavior was often guided by the keywords in anchor texts of URL that refer to a web object. The anchor link (URL) represents relationship between two web documents or two parts of the same web document. Anchor text provides relevant descriptive or contextual information to users about the contents related to a particular anchor link.

In this paper we present the semantic prefetching scheme for making the predictions by applying Naïve Bayes Classifier to compute the probability value of anchor texts present in a web page. Naïve Bayes Classifiers are fast, accurate and simple to implement. They are used to make predictions due to its simplicity and ability to learn dynamically whenever user access anchor links to view web pages. Our work focuses on extracting anchor links from a displayed web page, convert the anchor text associated with each anchor link into set to tokens (keywords), apply Naïve Bayes Classifier over the tokens to compute probability, generate preference list based on the probability to perform prefetching of web objects. When the user access an anchor link, now the system tries to satisfy the user request using web objects stored in the prefetching cache. Performance of the proposed scheme is evaluated by observing the browsing patterns of users in different web sessions. The results show that the hit rate improves when a user is involved in longer browsing sessions and access the web links relevant to a particular topic of interest.

The rest of this paper is organized as follows: Section 2 presents the related work carried out in web prefetching domain. Section 3 provides overview of the scheme used to achieve semantic prefetching and section 4 explains the implementation of prefetching scheme. Section 5 deals with the experimental setup and presents the evaluation results. Section 6 concludes the paper with remarks on future work.

2. Related Work

Prefetching has been applied to a variety of distributed and parallel systems to hide communication latency [1]. Crovella and Barford [2] analyzed the effect of prefetching on the network performance by considering network delay as the primary cost factor. A simple transport rate controlled mechanism was proposed to improve the network performance. The usage of anchor text to index URL's in Google search engine was suggested by Brin and Page [3]. The research focused on effective usage of additional information present in the hypertext. Chakrabarti et al [4] designed and evaluated an

automatic resource compilation system that could perform analysis of text and links to determine the web resources suitable for a particular topic.

Davison [5] conducted a detailed analysis that focused on examining the descriptive quality of web pages and the presence of textual overlap in web pages. A keyword-based semantic prefetching approach was designed by Cheng and Ibrahim [6, 16] that could predict future requests based on semantic preferences of past retrieved Web documents. The scheme was evaluated by considering the Internet news services. Neural network was applied over the keyword set to predict future requests. R.Cooley et al [7] developed a quantitative model based on support logic that used information such as usage, content and structure to automatically identify interesting knowledge from web access patterns.

The effectiveness of link-based and content-based ranking method in finding the web sites was analyzed by Craswell et al [8]. The results indicated that anchor texts are highly useful in site finding. Davison [9] proposed a text analysis method that examined web page content to predict user's next request. The algorithm used text in and around the hypertext anchors of selected web pages to determine user's interest in accessing web pages. Chen et al [10] proposed a framework that used link analysis algorithm for exploiting both explicit (hyperlinks embedded in web page) and implicit (imagined by end-users) link structures. The framework had the ability to analyze interactions between users and the web.

PageRank and HITS (Hypertext Induced Topic Selection) were the most popular webpage ranking algorithms. HITS emphasized on mutual reinforcement between the authority and hub web pages, whereas PageRank emphasized on hyperlink weight normalization. Ding et al [11] generalized the concepts of mutual reinforcement and hyperlink weight normalization into a unified framework. Nadav and Kevin [12] investigated anchor text to observe its relationship to titles, frequency of queries satisfied and the homogeneity of results obtained. Analysis indicated the anchor text resembled real-world queries in terms of its term distribution and length. Zhuge [13] defined semantic links between resources to establish a high-level single semantic image to improve the quality of search result sets. The mathematical notations and formal structure of the semantic link network was presented in [15].

Pierrakos et al [14] clearly analyzed and presented the web usage mining process such as data collection, data preprocessing and pattern discovery that could be applied for web personalization. Jung [17] carried out semantic outlier detection and segmentation using online web request streams to infer the relationships among web requests. A user-support mechanism based on knowledge sharing with users through collaborative web browsing was proposed in [18]. It mainly focused on extracting user's interests from their own bookmarks. Alexander Pons [19] proposed a technique that semantically bundled objects from slower loading web pages with objects of faster loading web pages. It was done to prefetch objects for the client's system prior to accessing the slower loading web page. Naresh and A.K. Sharma [20] carried out analysis on the web pages of different categories from Open Directory Project (ODP) [21]. They suggested the use of cohesive

and non-cohesive text present near the anchor text to extract information about the target web page.

3. Semantic Prefetching Scheme

The proposed scheme is responsible for making efficient predictions of web objects to be prefetched for satisfying the user's future requests with low latency. It is based on the concept of client-side prefetching, where the client directly prefetches web documents from the server and stores it in local cache to service user requests. It significantly reduces the latency when servicing user requests, since there is no network latency for retrieving locally cached documents. The prefetching scheme consists of the following components: Tokenizer, Prediction unit, Prefetching unit and the Prefetching cache.

Tokenizer

When user is viewing a web page, Tokenizer parses the web page to extract anchor links (URL) and its associated anchor text. It then identifies the tokens (keywords) from each anchor text of a link. A token is considered as the meaningful word within anchor text of a URL. When a user clicks anchor link in a web page, then tokenizer moves the tokens of that particular anchor text into user token repository. The repository has collection of tokens with their frequencies, where token frequency indicates the number of times a particular token is seen in the anchor text of links selected by the user. When a token occurs for the first time, new entry is created in the repository with initial count value as 1. For the existing tokens its count value gets incremented. The user token repository is used by the prediction unit to compute probability values of anchor links that are not accessed by the user in a web page.

Prediction Unit

It is responsible for computing the probability value of each anchor link using Naïve Bayes Classifier. The advantages of using Naïve Bayes Classifier [22] for computing the probability values are: a) simple mechanism to compute values for the specified data b) requires minimal storage, since it stores only token counts and c) incremental update whenever new data is processed. Anchor text associated with each link is taken and the tokens from it are compared with tokens stored in the user token repository to compute the probability value. The anchor links are then arranged based on the probability value to be given as input to the prefetching unit.

Prefetching Unit

The web objects that are required to satisfy the user requests with low latency are retrieved by the prefetching unit from the web server and stored in the local cache. The selection of web objects for prefetching is based on the preference list generated as output by the prediction unit. Prefetching is normally performed when the client is viewing a

web page (i.e. 'idle' time). Prefetch requests are given low priority than the regular user requests, so whenever user makes a request the prefetching unit suspends any ongoing prefetch activity. It is possible to prefetch only limited number of links at any time because of small time period to perform prefetch before user makes a new selection.

Prefetching Cache

The web documents that are prefetched from the server are stored in the prefetching cache to satisfy the user's future requests. To eliminate the caching impact due to temporal locality exhibited in the user access patterns, prefetching cache is managed separately from the browser's in-built cache. When new web documents need to be stored in the prefetching cache, it selects documents that are not accessed for a long time and purge it whenever there is insufficient space in the cache.

4. Implementation

When user views a web page, the anchor links in that page form a pool of URLs that the user might visit next. Semantic prefetching system is designed to efficiently identify a set of anchor links in the web page that reflects user's interests. Figure 1 shows the process of collecting anchor links from the web page and creating preference list to prefetch the web objects for satisfying user requests.

The steps illustrated in Figure 1 are explained as follows:

(1)   User requests a web page by providing its URL in the web browser.

(2)   The requested web page is retrieved and displayed to the user.

(3)   The displayed web page is scanned to extract the list of anchor links and its associated anchor text for evaluation.

(4)   The anchor text associated with each link is processed to generate the set of tokens, where 'token' corresponds to individual words in the anchor text.

(5)   The tokens of an anchor text are added to the user token repository when the user access particular anchor link.

(6)   For each token its count value is maintained in the user token repository.

(7)   The probability of each anchor link is computed by applying Naïve Bayes Classifier on the anchor text (i.e. set of tokens) with reference to the tokens maintained in user token repository.

(8)   Anchor links are sorted based on the computed probability value and placed in the preference list.

(9)   Prefetching unit takes anchor links from the preference list and retrieves web objects from the web server, which are then placed in the prefetch cache.

(10)   When the user clicks an anchor link in a web page, the system first verifies its availability in the prefetch cache.

(11)   If it is available in the prefetch cache, then web page will be displayed.

(12)   If web objects are not available in the prefetch cache, then it is retrieved from the web server and displayed to the user.

```
                    ┌─────────────────────┐
              1     │    Web Browser       │
                    ├─────────────────────┤
                    │ User enters the URL  │
                    └─────────────────────┘
                              │ 2
                              ▼
              ┌─────────────────────┐                    ┌─────────────────┐
              │ Web page displayed  │                  5 │  User Tokens    │
              └─────────────────────┘                    └─────────────────┘
                        │ 3                                       │ 6
                        ▼                                         ▼
              ┌─────────────────────┐                    ┌─────────────────┐
              │    Anchor Links     │                    │   Token Count   │
              ├─────────────────────┤                    └─────────────────┘
              │ 1. . . . . . . .    │                             │ 7
              │ 2. . . . . . .      │                             ▼
              │ 3. . . . . . .      │     Naïve Bayes    ┌─────────────────┐
              └─────────────────────┘     Classifier ──▶ │  Anchor Link    │
                        │ 4                                │  Probability    │
                        ▼                                 └─────────────────┘
              ┌─────────────────────┐                             │ 8
              │   Anchor text to    │                             ▼
              │      Tokens         │                    ┌─────────────────┐
              └─────────────────────┘                    │ Preference List │
                                                         ├─────────────────┤
          10          ┌─────────────────────┐         9 │ 1. . . . . . . . │
              ───────▶│   Prefetch Cache    │◀──────────│ 2. . . . . . . . │
                      └─────────────────────┘           │ 3. . . . . . . . │
                            │ 11                         └─────────────────┘
                            ▼
                      ┌─────────────────────┐          ┌──────────┐
                      │  Display web page   │◀───────▶ │ Internet │
                      └─────────────────────┘   12     └──────────┘
```
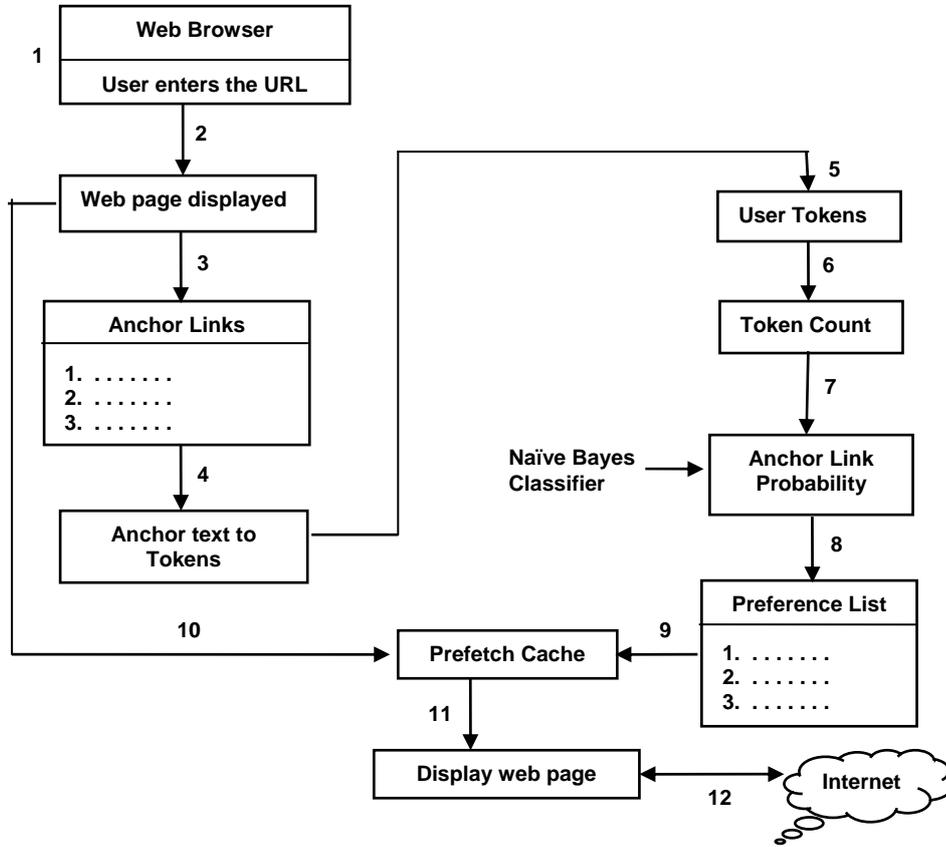
Fig.1. Steps for Predicting and Prefetching the Web objects

4.1 User Token Repository

A transient user token repository is maintained at the client side that stores information exhibiting the user's browsing interest in a particular topic. The information maintained in the user token repository exhibits both user and session characteristics, where a session represents the time interval elapsed between start and end of user's browsing instance. During a browsing session the user clicks several anchor links to access web pages of his interest. Anchor text associated with anchor links that receive click event will be retrieved and stored in the user token repository as independent tokens (words). Trivial characters and tokens such as prepositions and conjunctions are eliminated from the anchor text before they are stored in the user token repository.

Table 1. User Token Repository

Token	Count
academics	12
admissions	6
applied	2
apply	3
checklist	1
computer	5
courses	1
deadlines	1
engineering	2
future	4
graduate	5
information	1
prospective	4
requirements	1
science	3
students	6

The user token repository is implemented using a hash table that stores the list of tokens and the occurrence count for each token. Whenever the probability value of an anchor text need to be computed for taking prefetching decisions, the tokens stored in the user token repository are used. A sample user token repository used for computing the probability value of anchor text is shown in Table 1. The size of user token repository should be selected appropriately. If the repository size is too small, then it will force the system to purge legitimate tokens. If the repository size is too large, then it will be filled with trivial tokens wasting space and time during evaluation.

4.2 Computing Probability Value

The probability value for each anchor link is computed by the prediction unit using Eq. (1) conceived based on Naïve Bayes Classifier [23].

$$P_r(U|A) \quad = \frac{P_r(A|U) \cdot P_r(U)}{P_r(A)} \tag{1}$$

U   = User Token Repository
A   = Anchor Text associated with an Anchor Link
$P_r(U|A)$ = probability that an anchor text is in user token repository
$P_r(A|U)$ = probability that for given user token repository the tokens of an anchor text
                   appears in that repository
$P_r(U)$ = probability of user token repository
$P_r(A)$ = probability of occurrence of a particular anchor text

In our scheme we use only one repository type (i.e. user token repository), so P_r (U) takes a constant value 1. P_r (A) also takes a constant value and acts just as a scaling factor for P_r (U | A), which could be omitted during the computation. Based on these factors, Eq. (1) is simplified to:

$$P_r(U|A) = P_r(A|U) \cdot 1 \tag{2}$$

The probability P_r (A | U) in Eq. (2) is computed using the following steps:

1) The anchor text of a URL will be split into tokens.

$$\text{Anchor Text} = \{Token_1, Token_2, Token_3 \ldots Token_m\}$$

2) Compute the probability of each token relative to user token repository

$$P_r(Token_i|U) = \frac{Count \ of \ Token_i \ in \ U}{Total \ Count \ of \ Tokens \ in \ U} \tag{3}$$

where i = 1 to m, [m = number of tokens]

3) Compute the probability of anchor text using Eq. (4), which performs the product of individual token probabilities.

$$P_r(A|U) = \prod_{i=1}^{m} \left[C + P_r(Token_i|U) \right] \tag{4}$$

The variable 'C' takes value 1 and it is added to the individual token probability irrespective of whether the token of anchor text is present in the user token repository or not. The reason for adding 'C' to the token probability is to avoid two cases: a) probability of anchor text to be less than individual token probability b) product value becoming zero because few tokens of anchor text not present in the user token repository. If none of the tokens in an anchor text are present in the user token repository, then its product value will be 1 because of adding value 'C' to each token probability. When either few or all the tokens of anchor text are present in the user token repository, then its product value will be greater than 1. Based on these criteria, anchor links having probability value greater than 1 are considered for inclusion in the preference list. Since we need to add 'C' value to the individual token probabilities, the probability of anchor text is not scaled within the range of 0 to 1 and it takes only value greater than 1.

4.3 Preference List

When a web page is displayed on the browser, prediction unit computes the probability for anchor links embedded in that web page. Based on the computed probability values, preference list is created and fed as input to the prefetch unit. The preference list is implemented as a priority queue that stores anchor links. The prefetch unit makes use of browser's idle time to prefetch web objects using anchor links stored in the priority queue. Anchor links with high probability values are given top priority during prefetching. When a user navigates to new web page by clicking any link on the current

web page, then the priority queue will maintain new set of anchor links to be prefetched. This helps to eliminate the prefetching of irrelevant links during the user browsing session. Figure 2 shows the process of adding anchor links to the priority queue for prefetching the web objects.
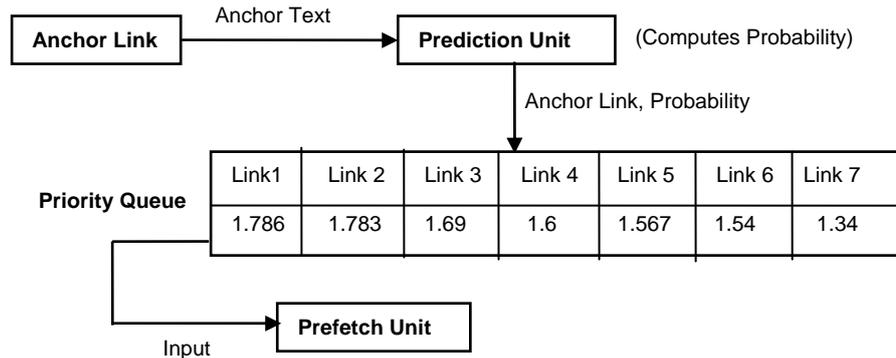

Fig.2. Implementation of Priority Queue for Prefetching

5. EVALUATION

Performance of the proposed prefetching scheme cannot be evaluated efficiently using trace based simulations. It is due to the fact that semantic technique tries to capture individual user interests in a single browsing session, but the web traces could not give a comprehensive view of client's interests. An efficient way to capture the required information would be to extract user interests at the client side. It is done by capturing the user interests using information obtained through user's browsing patterns. The user accesses web pages in two ways: a) pass URL request directly in the web browser and b) navigate across the web pages using anchor links embedded in a web page. In our semantic prefetching approach, we extracted user interests by exploiting the navigational behavior (web page access using anchor links) exhibited by the user.

The performance of the proposed scheme is tested by performing user interest based browsing, where an individual user is assigned the task of obtaining information about a particular topic of his interest. For the evaluation purpose we used an open source browser-cxbrowser [24] developed in C#, using which the users will access the web pages. The proposed prefetching scheme is implemented as an add-on to the browser. It provides facility for the user to configure prefetching settings based on his requirements. The user can enable or disable prefetching during a browsing session. The user requests in a browsing session are maintained in a log file for performance analysis.

The results are taken by monitoring the browsing pattern of two different users - User_A and User_B. User_A takes the role of a prospective student visiting several university websites for gathering information related to his topic of interest. User_B takes the role of a person visiting news portals for gathering information of his interest. The websites considered for evaluation possess identical semantic structure and content.

The anchor links to be prefetched are predicted based on the probability values computed using Eq. (4). The prefetch operation takes place only during user's idle time (i.e. when user not browsing), so depending on the user's access pattern the number of links prefetched will vary dynamically. In Figure 3, x-axis denotes the pages accessed by user in a browsing session and y-axis denotes the number of anchor links that are predicted and prefetched. When the user initially starts a browsing session, the user token repository will be empty and it cannot make any predictions of web objects. As the user starts browsing web pages by clicking anchor links, then user token repository will be filled with tokens based on which it starts making the predictions. If the user moves from current page to next page quickly, then less number of web objects is prefetched. In some cases the user spends long time in a particular web page, which increases the idle time causing more number of links to be prefetched. When a web page contains only small amount of information that are relevant to user interests, then minimum predictions are generated for that web page.
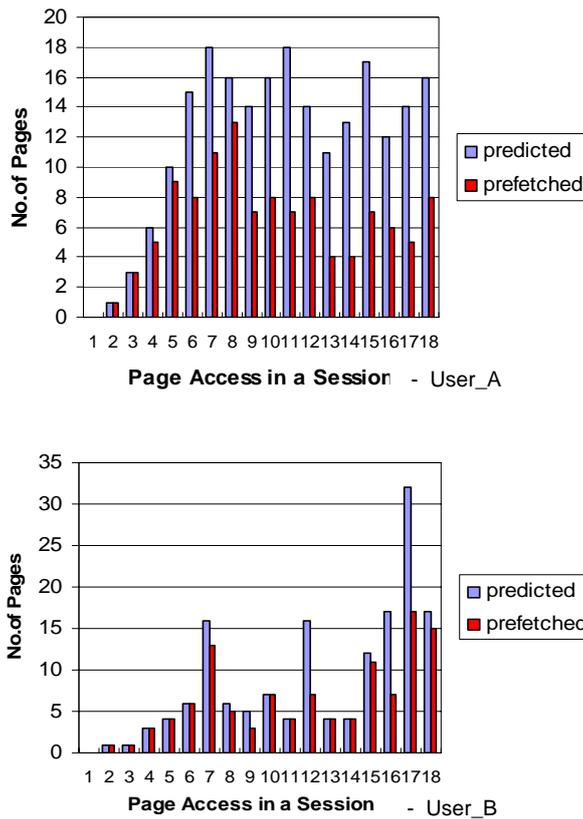

Fig.3. Predictions and Prefetch for User_A and User_B

In Figure 4, we show the hit rate achieved during different browsing session durations based on the browsing patterns of User_A and User_B. In the x-axis we show the user session duration in minutes and in y-axis the percentage of hit rate achieved. When user has long browsing sessions (i.e. 50, 60 or 70 minutes), then user token repository will store large number of tokens that helps in effective predictions and prefetching resulting in high hit rates. Since User_A visits only the university websites which do not change its contents often, it is able to achieve consistently high hit rates for different session durations.
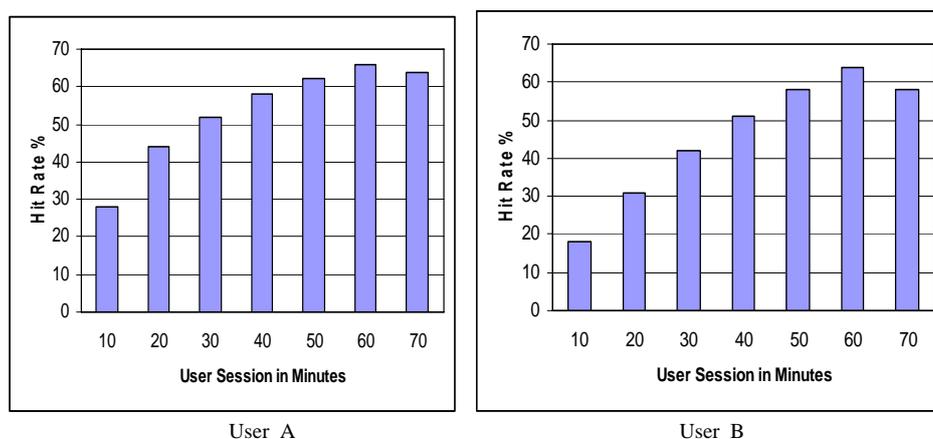

User_A   User_B

Fig.4. Hit Rate for various user sessions

6. CONCLUSION

In this paper we have presented a semantic web prefetching scheme that uses Naïve Bayes Classifier for making the predictions of web objects to be prefetched. The scheme provides efficient predictions when users are visiting web pages that contain information related to a specific topic of interest. Since predictions are done based on the computed probability value of anchor texts, it is essential that user's use this scheme only when browsing the web pages for similar content. In case the user visits web pages in random without looking for specific content, then the scheme will provide weak predictions resulting in unnecessary prefetching of web objects. The prefetching scheme helps to minimize the latency when satisfying the user requests by achieving good hit rates across different session durations.

The prefetching option could be enabled or disabled by the user based on his interests. In our work we considered only the anchor texts of URL for making the predictions. In some cases the URL contain anchor texts with either a single token or the anchor texts are missing, which may negatively impact the predictions. This limitation could be handled by considering the text in and around the anchor link for making the predictions. The proposed scheme is evaluated by browsing only the news and university

websites; it could well be extended to other websites also. The predictions are done based on general information obtained from web sites without focusing on specific category based information. The efficiency of proposed scheme can be compared with other learning approaches and similarity measures such as fuzzy logic and support vector machines.

REFERENCES:

[1]   D. Culler, J. Singh, and A. Gupta, "Parallel Computer Architecture: A Hardware/Software Interface", Morgan Kaufmann, 1998

[2]   M. Crovella and P. Barford, "The Network Effects of Prefetching", Proceedings of IEEE Infocom'98, pp. 1232-1240, March 1998

[3]   S. Brin, L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", Computer Networks and ISDN systems, vol. 30, no.1-7, pp. 107-117, 1998

[4]   S. Chakrabarti, B.Dom, D. Gibson, J. Klienberg, P. Raghvan, S. Rajgopalan, "Automatic Resource List Compilation by Analyzing Hyperlink Structure and Associated Text", Proceedings of 7th International WWW conference, 1998

[5]   B.D. Davison, "Topical Locality in the Web", Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval, 2000

[6]   T.I.Ibrahim, C.Xu, "Neural net based predictive pre-fetching to tolerate WWW latency", Proceedings of the 20th International Conference on Distributed Computing Systems, 2000

[7]   R.Cooley, P.Tan, J.Srivastava, "Discovery of interesting usage patterns from Web data", Lecture Notes in Artificial Intelligence, 1836, pp.163-182, Springer, Berlin, 2000

[8]   N. Craswell, D. Hawking, S.E. Robertson, "Effective Site Finding Using Link Anchor Information", Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval, 2001

[9]   B.D.Davison, "Predicting web actions from HTML content", Proceedings of 13th ACM Conference on Hypertext and Hypermedia, 2002

[10]   Z.Chen, L.Tao, J.Wang, L.Wenyin, W.Ma, "A unified framework for web link analysis"; Proceedings of 3rd International Conference on Web Information Systems Engineering, pp. 63-72, Singapore , 2002

[11]   C.Ding, X.He, P.Husbands, H.Zha, H.Simon, "PageRank, HITS and a Unified Framework for Link Analysis"; Technical Report, 49372, Lawrence Berkeley National Laboratory, 2002

[12]   Nadav Eiron and Kevin S. McCurley, "Analysis of Anchor Text for Web Search", Proceedings of SIGIR, 2003

[13]   H.Zhuge, "Active e-document framework ADF: model and platform", Information and Management, vol.41, No.1, pp. 87–97, 2003

[14]   D.Pierrakos, G.Paliouras, C.Papatheodorou, C.D. Spyropoulos, "Web usage mining as a tool for personalization: a survey", User Modeling and User-Adapted Interaction, vol.13, no.4, pp. 311-372, 2003

[15]   H.Zhuge, "The Knowledge Grid", World Scientific Publishing Company, Singapore, 2004

[16]   Cheng-Zhong Xu and Tamer I. Ibrahim, "A Keyword-Based Semantic Prefetching Approach in Internet News Services", IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 5, pp.601 -611, 2004

[17]   Jason J. Jung, "Semantic Preprocessing of Web Request Streams for Web Usage Mining", Journal of Universal Computer Science, vol. 11, no. 8 pp. 1383-1396, 2005

[18]   J.J.Jung, "Collaborative web browsing based on semantic extraction of user interests with bookmarks", Journal of Universal Computer Sciences, vol.11, no. 2, 2005

[19]   Alexander P. Pons, "Semantic prefetching objects of slower web site pages", The Journal of Systems and Software, Vol.79, pp.1715–1724, 2006

[20]   Naresh Chauhan, A.K. Sharma, "Analyzing Anchor-Links to Extract Semantic Inferences of a Web page", Proceedings of 10th International Conference on Information Technology, pp.277-282, 2007

[21]   Open Directory http://dmoz.org/

[22]   I. Rish, "An empirical study of the naive Bayes classifier", Workshop on Empirical Methods in Artificial Intelligence, IJCAI, 2001

[23]   Ioan Pop, "An approach of the Naive Bayes classifier for document classification", General Mathematics, Vol. 14, No. 4, pp.135–138, 2006

[24]   Cxbrowser – http://cxbrowser.sourceforge.net