Hindawi
Scientific Programming
Volume 2020, Article ID 8897244, 18 pages
https://doi.org/10.1155/2020/8897244

Research Article

HPM: A Hybrid Model for User's Behavior Prediction Based on *N*-Gram Parsing and Access Logs

Sonia Setia,[1,2] **Verma Jyoti**,[3] **and Neelam Duhan**[3]

[1]*J. C. Bose University of Science and Technology, YMCA, Faridabad 121006, India*
[2]*Faculty of Computer Applications, MRIIRS, Faridabad, India*
[3]*Faculty of Computer Science, J. C. Bose University of Science and Technology, YMCA, Faridabad 121006, India*

Correspondence should be addressed to Sonia Setia; setiasonia53@gmail.com

Received 26 July 2020; Revised 6 October 2020; Accepted 20 October 2020; Published 6 November 2020

Academic Editor: David Ruano-Ordás

The continuous growth of the World Wide Web has led to the problem of long access delays. To reduce this delay, prefetching techniques have been used to predict the users' browsing behavior to fetch the web pages before the user explicitly demands that web page. To make near accurate predictions for users' search behavior is a complex task faced by researchers for many years. For this, various web mining techniques have been used. However, it is observed that either of the methods has its own set of drawbacks. In this paper, a novel approach has been proposed to make a hybrid prediction model that integrates usage mining and content mining techniques to tackle the individual challenges of both these approaches. The proposed method uses *N*-gram parsing along with the click count of the queries to capture more contextual information as an effort to improve the prediction of web pages. Evaluation of the proposed hybrid approach has been done by using AOL search logs, which shows a 26% increase in precision of prediction and a 10% increase in hit ratio on average as compared to other mining techniques.

1. Introduction

The World Wide Web (WWW) has become an important place for people to share information. The amount of information available on the web is enormous and is growing day by day. As a result, it is the need of the hour to develop new techniques to access the information very quickly and efficiently. For fast delivery of media-rich web content, latency tolerant techniques are highly needed, and several methods have been developed in the past decade in this regard. Among these techniques, the two most prevalent techniques are caching and prefetching. However, caching benefits are limited due to the lack of sufficient degrees of temporal locality in the web references of individual clients [1]. The potential for caching of the requested files is even declining over the past years [2]. On the other side, prefetching is defined as "to fetch the web pages in advance before a request for those web pages" [3]. The usefulness of prefetching the web pages depends upon how accurately the prediction for those web pages has been made. A good prediction model can find various applications, of which the most prominent ones are website restructuring and reorganization, web page recommendation, determining the most appropriate place for advertisements, web caching and prefetching, etc. In recent years, due to the wide scale of applications, the prediction process has gained more importance. To make predictions, several web mining techniques have been used in the past several years. Web mining [4] can be divided into three distinct areas:

(i) Web usage mining: it involves analyzing user access patterns collected from web servers better to predict the users' needs [5–7]

(ii) Web content mining: it involves extracting useful information from websites to serve the users' needs

(iii) Web structure mining: it is the study of the interlinked structure of web pages

Traditional prefetching systems make predictions based on the usage information present in access logs. They typically employ the data mining approaches like association rule mining on the access logs to find the frequent access patterns, match the user's navigational behavior with the antecedent of the rules, and then prefetch the consequent of the rules. However, this approach's problem is that a relevant page that might be of user's interest can be exempted from the prediction list if it is new or it was not frequently visited before; therefore, it does not appear in frequent rules.

On the other side, predictions based on content information present in web pages such as title, anchor text, etc. resolve these problems, but they have their own set of drawbacks. They lack the user's intent of the search, and web content alone is insufficient to make accurate predictions.

In this paper, instead of focusing only on the content, i.e., anchor texts associated with URLs (Uniform Resource Locator), the queries submitted by users recorded in web access logs have also been crucial for actual user's interest. Therefore, a hybrid prediction model (HPM) has been proposed, which incorporates both the history of the users' browsing behavior and the information content inherent in the users' queries. It is based on the Query-URL click-graph, a bipartite graph G between queries Q and URLs U, which are extracted from the access logs. Edges E in the diagram indicate the presence of clicks between queries and URLs. Weight $C_{q,u}$ is assigned to each edge, representing the aggregated clicks between query q and URL u. N-gram parsing of queries has also been used for better results as compared to unigrams. An N-gram [8] is an N-word sequence. An N-gram of size 1 is referred to as a unigram, 2-gram as a two-word sequence, also called bigrams, and size 3, i.e., 3-gram meaning a three-word sequence, trigram. For example, parsing the query "college savings plan," we get three unigrams ("college," "savings," "plan"), two bigrams ("college_savings," "savings_plan"), and one trigram ("college_savings_plan"). The reason to use the N-gram approach is that grams can capture more contextual information, which can help us to predict the frequency of such kinds of keywords.

The advantages of this prediction framework mainly lie in three aspects:

(i) First, query terms are used through the Query-URL click-graph to understand users' behavior more accurately rather than using noisy and ambiguous web page content

(ii) Second, it captures information from both usage logs and content knowledge, which increases the accuracy of prediction

(iii) Third, this framework further considers the N-gram parsing of queries, which also improves the prediction results

The paper has been organized as follows. Section 2 highlights the detailed literature review on prefetching. The proposed approach is presented in Section 3, which discusses the following:

(i) The architecture of the hybrid prediction model

(ii) The workflow of both phases, i.e., online phase and offline phase

(iii) Detailed pseudocode for the proposed method

Further, Section 4 discusses an example of the proposed work. Experimental evaluation and comparison of the proposed work with the existing approaches are provided in Section 5. Section 6 finally concludes this work with future enhancement.

2. Related Work

Web prediction is a classification problem to predict the next web page that a user may visit based on its browsing history. Several researchers have been trying to improve the prediction of users' browsing experience in the past decade to achieve the following research objectives:

(i) To improve the accuracy of prediction

(ii) To remove the scalability problem

(iii) To improve prediction time

This section talks about various techniques and methods used to develop web page predictions categorized under usage mining, content mining, and structure mining.

2.1. Prefetching Techniques Using Usage Mining. Markov model is a mathematical tool for statistical modeling, one of the popular methods used for prefetching. Generally, the Markov model's basic concept is to predict the next action, which depends on the results of previous actions. Several researchers have used this technique successfully in various literature studies to train and test user actions or predict their future behavior.

Deshpande and Karypis [9] and Kim et al. [10] investigated that high accuracy in the prediction of the next web page can also be achieved by using higher-order Markov models. Still, higher-order Markov models have high space complexity, whereas lower-order Markov models cannot capture the users' browsing behavior accurately. To solve this problem, Verma et al. [11] proposed a novel approach for web page prediction using the k-order Markov model, where the value of "k" has been chosen dynamically. In addition to this work, Oguducu and Ozsu [12] and Lu et al. [13] worked upon user sessions. User sessions were clustered and represented by clickstream trees for making predictions. But it raises a scalability problem. Further, Awad and Khalil [14] analyzed the Markov model and all-K^{th} Markov model to solve the web prediction problem to remove scalability problem. The proposed framework by [14] improved the prediction time without compromising prediction accuracy.

Zou et al. [15] found that more accurate prediction models are required; therefore, more complex prediction tasks must run. In this paper, the authors proposed the intentionality-related long short-term memory (Ir-LSTM) model, which is based on the time-series characteristics of browsing records. Further, Joo and Lee [16] proposed a framework for user-web interaction called WebProfiler.

Basically, it predicts the user's future access based on user interaction data collected by this profiler. The authors claimed that overall prediction performance using the proposed model had been improved by 13.7% on average.

Martinez-Sugastti et al. [17] presented a prediction model based on history-based prefetching approach. This model considers the cost of prediction in terms of cache hits and cache misses of the forecast to train the prediction model so that more accurate results can be achieved based on the previous cache hits. The authors claimed that, by using this model, the precision of prediction had been improved, and latency has been reduced. Veena and Pai [18] proposed the "Density Weighted Fuzzy C Means" clustering algorithm to cluster similar user's access patterns. This algorithm can be used for the recommendation system as well as the prefetching system.

2.2. Prefetching Techniques Using Content Mining. Keeping content at the epicenter of the research approach, Venkatesh [19] proposed a prefetching technique that used hyperlinks and associated anchor texts present in the web page for predictions. The probability of each link was computed by applying Naïve Bayes classifier on the anchor text concerning keywords of the user's interest. The connections with higher chances were chosen for prefetching. Further, Setia et al. [20] extended this work by considering the semantic preferences of the keywords present in the anchor text associated with the hyperlinks.

Researchers [21–23] proposed a semantically enhanced method for a more accurate prediction that integrated the website's domain knowledge and web usage data.

Authors [24, 25] found that only the user's access patterns are insufficient to predict the user's behavior. The authors [24] worked upon an individual user's behavior. Authors [25] analyzed that web pages' content should also be taken into account to capture the user's interest.

2.3. Prefetching Techniques Using Structure Mining. Web link analysis [26] proved to be an important factor in performing a good quality web search. It can also calculate how the web pages are related to each other. Link analysis approaches are divided into two types: "explicit link analysis" and "implicit link analysis." Hyperlinks present on the web page are called explicit links. It has been proved by Davison [27] that hyperlink information can help a lot in web search. Web designers design the structure of the links and embed the links in the website. Therefore, in the case of the "explicit link analysis" technique, the user follows the design that the website designer was responsible for making any web page important, e.g., Kleinberg's HITS [28]. However, in the "implicit link analysis" technique, the importance of a web page is not determined by the web page designer, but it is done by the users who are accessing that web page. The higher the number of users accessing the web page, the more influential the page is. Whenever a user accesses a web page, an implicit link is developed between the user and the corresponding web page. Further, pages are visited by the user in a sequential manner, forming implicit associations one after another. So, in the latter case, the web page is essential from the user's point of view. An example of the implicit link analysis approach is DirectHit [29]. Researchers [26] used both techniques, i.e., "explicit link analysis" and "implicit link analysis," and further improved the search accuracy by 11.8% and 25.3%, respectively.

Authors [30–32] found that the poor structure of the website may degrade the performance of any algorithm which works upon the structure of the website for user navigation. Sheshasaayee and Vidyapriya [30] proposed a framework to reorganize the website using splay trees, a self-balancing data structure. Further, Thulase and Raju [32] extended this approach by using concept-based clustering. Vadeyar and Yogish [31] developed farthest first clustering-based technique to reorganize the website.

Table 1 describes in brief different methods for prefetching technique with appropriate justification in the context of research work.

A critical look at the above table highlights the fact that each of the existing prefetching techniques proposed by researchers has its drawbacks. Either these techniques are lacking in making the right set of prediction or the choice of parameters is not sufficient or the cost involved in making such predictions is very high.

2.4. Problem Statement. A precarious look at the literature highlights the following areas of improvements:

(i) Most of the techniques utilize the browsing history of users stored in client logs, proxy logs, or server logs in the literature. The information found in any type of access logs varies according to the format of the records. Administrators select the log data in their way. But due to insufficient information present in logs, inaccurate predictions are derived, rendering the prefetching approaches to work inefficiently. These techniques cannot predict those web pages which are newly created or never visited before.

(ii) Web pages' content information has also been widely used for predictions as a solution to the above-said problem. These techniques use the content information such as titles, anchor text, etc. which do not provide sufficient details of the user's interest and thus cannot be considered alone for prediction algorithms to work.

(iii) Structure mining-based prediction techniques depend only upon how website structure has been designed. The reorganization of the website structure for user navigation increases computational cost.

It leads to the following main problems of prediction:

(i) Less accurate prediction results and, therefore, less precision

(ii) Low hit ratio of predicted pages and, therefore, more consumption of network bandwidth

Table 1: Prefetching technique with various methods and their justification.

Sr. No.	Method used	Literature reference	Description	Justification in the context of research work
1.	Markov model	[9–13]	It is a well-known approach for pattern recognition. It determines the next state from the current state based on the orders of the Markov chain	The main problem is lack of prediction accuracy with lower-order chain, while high complexity with the higher-order chain. However, this approach does not suit the current research context
2.	Prediction by partial match	[15, 16, 33]	The PPM model uses a set of previous objects to predict the next item in a particular stream	It is a restricted version of Markov chain that provides prediction based on the only selected set of objects and selection of a right set of objects is a very challenging task, so this kind of vision is not also; it limits the result as it does not cover all the objects, thereby ruling it out of the scope of current work
3.	Cost function	[14, 17]	Prediction of future requests has been made based upon certain factors like the popularity and lifetime of web objects	A very less popular approach for pattern determination as the cost functions vary from time to time, thereby reducing the contribution in making the right set of prediction. So this approach is also not suitable in the context of the proposed research
4.	Data mining	[18]	It is also one of the most popular approaches in the modern era for pattern recognition of structured objects	The data mining approach consists of many techniques which are ideal for pattern generation task. But the proposed research is not working upon pattern generation task
5.	Keyword based	[19, 20, 24, 25]	Prediction is made by retrieving confidential information present in the contents of web documents	To work upon only this category is not much beneficial since it does not deal with multiple user transactions
6.	Integration of domain knowledge	[21–23]	It works by the integration of domain knowledge with other methods of prefetching; semantics are taken into account	It gives useful information based on semantics but increases prediction time as well as extra overhead
7.	Implicit link analysis	[26, 29–32]	In the “implicit link analysis” technique, the importance of a web page is determined by the users who navigate the web page	It is a significantly less popular approach for pattern determination. Extra work is required to reorganize the structure of the website as per user navigation
8.	Explicit link analysis	[26–28]	In the “explicit link analysis” technique, the importance has been given to the design that has been structured by the designer who makes any web page more important or less important	It gives useful information based on hyperlink structures of the web

To improve the prediction technique, a hybrid prediction model is proposed in this work, which utilizes the best of both the information, i.e., the usage information and the content information of the web pages. The poor structure of a website may degrade the performance of such kind of techniques. Therefore, we are not considering structure mining for our proposed approach.

3. Proposed Hybrid Model

This work uses the Query-URL click-graph concept, which enables incorporating crucial contextual information in the prediction algorithm. In general, the workflow of our proposed approach (shown in Figure 1) is carried out in two phases, which is discussed as follows:

(i) Offline phase: the offline phase works at the backend and runs periodically to update the logs. Since it is a hybrid model, the input to this phase is the access logs and the content information of the web pages. The combined data from both sources is then put to use by using various intermediary steps to make a relevant prediction of users’ behavior. The output of this phase is the weighted logs (WL) that contain the weighted N-grams corresponding to the respective URLs.

(ii) Online phase: the online phase involves both the proxy and the client. While users interact with the system, the system predicts users’ behavior according to the user’s information. This information is matched with the information collected from the logs in the offline phase.

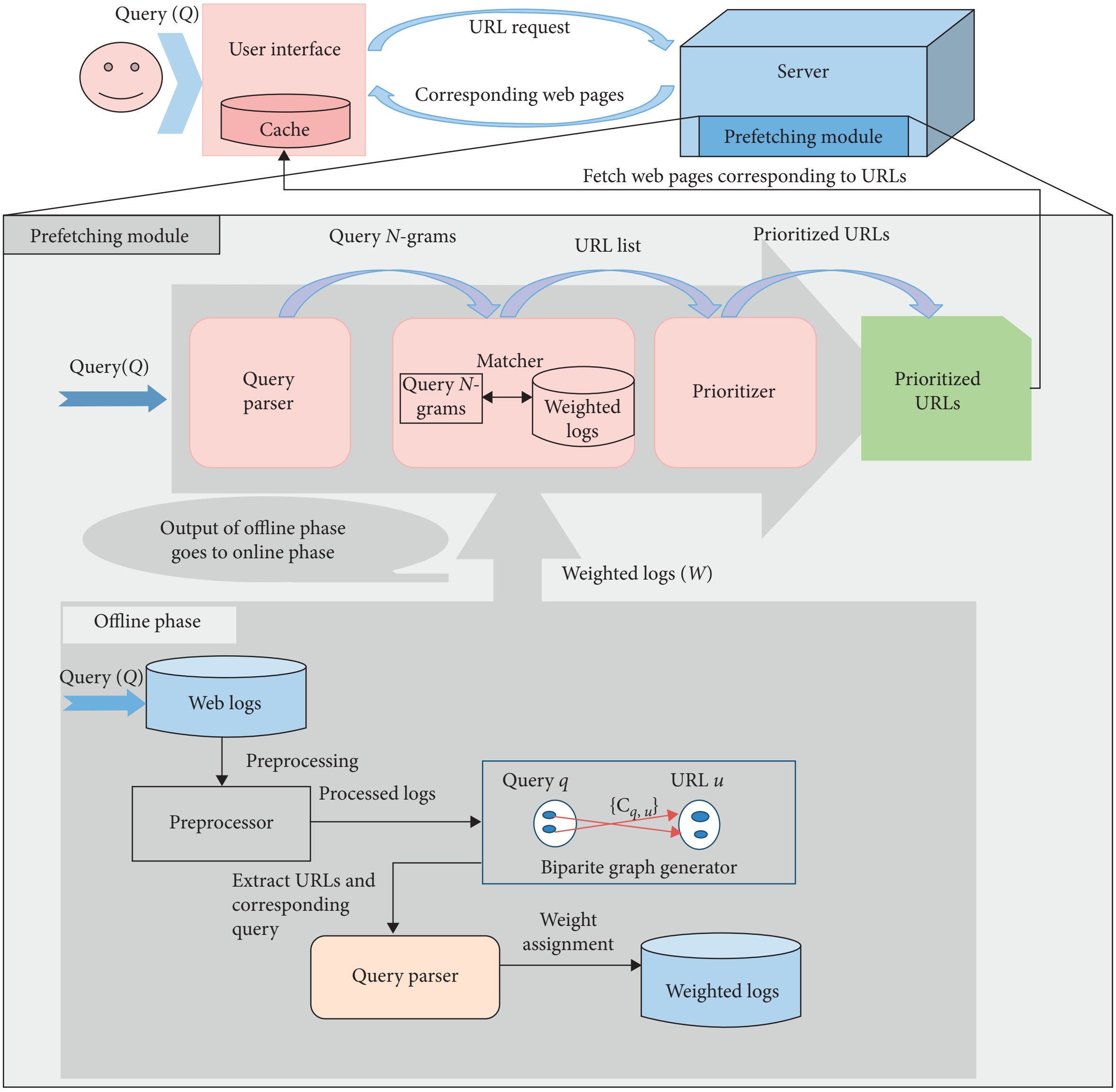


Figure 1: Architecture of hybrid prediction model.

3.1. Work Flow of the Offline Phase. This phase works in several steps, as follows:

(1) Preprocessing: initially, the offline phase considers access logs. Logs contain an entry for each request of the web pages made by the client. Various fields [34] of the records are anonymous user id, requested query, date and time at which the server is accessed, item rank, and URL clicked by the user corresponding to the requested query.

Each access log entry is preprocessed to remove stop words and extract the requested query, clicked URL corresponding to the requested query.

The processed information gets stored in the form of processed logs (PL).

(2) Bipartite graph generation: a bipartite graph between queries Q, and URLs U, taken from PL, is generated. The bipartite graph has been chosen because it helps us to improve readability. This new representation naturally bridges the semantic gap between queries and web page content and encodes rich contextual information from queries and users' click behaviors for prediction. This helps to reduce the space and computational complexity as it eliminates the need to scan the logs each time. Also, click count of the queries for the respective URLs is calculated as the graph is being generated in order to reflect the users' confidence in the query, i.e., how close the queries are connected with the clicked URLs. The edges between Q and U indicate the presence of clicks

between queries and their corresponding URLs. The generated bipartite graph is known as Query-URL click-graph (C-graph). The nomenclature for the generated C-graph is as follows:

(i) $Q = \{q_1, q_2, \ldots, q_m\}$.

(ii) $U = \{u_1, u_2, \ldots, u_n\}$.

(iii) $\langle C_{q,u}\rangle$ is an edge depicting number of clicks between Q and U.
Consider an example having $Q = \{q_1, q_2\}$ and $U = \{u_1, u_2, u_3\}$. A sample C-graph is depicted in Figure 2.
Here, the label on the Edge $\langle q_1, u_1\rangle$, i.e., $C_{q1,u1}$, depicts that the URL u_1 has been clicked five times corresponding to the query q_1.

(3) Query parsing: queries present in C-graph are parsed into N-grams that describe the URLs' content, resulting in N-gram associated click-graph (NC-graph).

(4) Weight assignment: weights are assigned to each N-gram in the query, present in NC-graph, based on the number of times a query has been clicked, which is depicted on the edges by $C_{q,u}$ in C-graph. The same click count is assigned to each N-gram of query, i.e., $C_{n,u}$, which is equivalent to $C_{q,u}$, where $\langle C_{n,u}\rangle$ is an edge depicting the number of clicks between N-gram n and URL u. For example, query q_1 is parsed into N-grams n_1 and n_2 which results in NC-graph depicted in Figure 3. As we can see in Figure 2, $C_{q1,u1} = 5$; therefore, its N-grams, i.e., $C_{n1,u1} = 5$, and $C_{n2,u1} = 5$.

Corresponding to each URL "u," a weighted vector is defined that comprises the weighted N-gram $w_{n,u.}$ Further, $W_{n,u}$ is computed by adding click count of the N-grams ($C_{n,u}$) coming from different queries for that URL.

Finally, weighted N-grams are normalized to rescale the values by using

$$W_{n,u} = \frac{w_{n,u}}{\sum_{v\in Vu} C_{v,u}},\qquad V_u = \left\{V \in N_q : N_q \in \langle q, u\rangle\right\}, \tag{1}$$

where $w_{n,u}$ is divided by the summation of click counts of all the terms corresponding to all the queries representing the URL u,where

u represents the URL

n represents one N-gram for the query

v is a term

V_u defines all the words belonging to N-grams about the different queries representing the URL u

N_q represents all the N-grams of the query q

$w_{n,u}$ represents weight of N-gram n in the URL u

$C_{v,u}$ represents click count of each term for the URL u

All the processing is done in temporary memory, and finally, it outputs weighted logs, which contain the URLs and their corresponding N-grams and their associated weights. The schema of access logs (AL), processed logs (PL), and weighted logs (WL) is shown in Figure 4.

The description of different attributes is given in Table 2.

It is important to note here that the offline phase runs periodically to update access logs. On every periodic update, only the fragment containing new entries in access logs is considered for further processing, and accordingly, weighted logs are updated. This job is done by the Incremental Module, a submodule of the prefetching module, as depicted in Figure 5.

3.2. Work Flow of the Online Phase. The online phase can be discussed in five major steps, as follows:

(1) Query initiation at interface: user enters a query according to his interest, which goes to the server through a proxy using the HTTP GET method. The server responds with the list of URLs corresponding to the respective query.

(2) Parser activation: while the user views the current page, the proxy server uses this query for further processing at the back end. This initializes the parser that parses this query into N-grams called query terms stored in set T. The resulting query terms are used to find the relevant URLs (from the weighted logs (WL)) corresponding to the respective query.

(3) Matcher activation: this phase takes as input the query terms from T from the online phase and weighted logs (WL) from the offline stage. The weights of URLs corresponding to the users' query are calculated by comparing the users' query terms T with the weighted N-grams of URLs in WL. This process is carried with the help of (2):

$$W_u = \sum_{t\in T} W_{t,u} * I_{t,u}, \tag{2}$$

where

W_u represents the weight of each URL,

W_t represents the weight of each term present in the URL,

$I_{t,u}$ is a vector for each URL, i.e.,

$$I_{t,u} = \begin{cases} 1, & \text{if } t \text{ present in URL } u, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

(4) Prediction list generation: these weights are then fed to the prediction unit. It prioritizes the URLs based on their weights generated in step 3. A prediction list of URLs corresponding to the user query based on this prioritization is generated.

(5) Prefetching: prefetcher prefetches the predicted URLs and stores them in the cache.

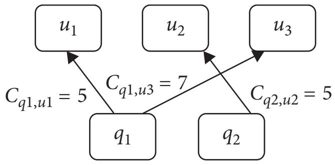


Figure 2: Example of C-graph.

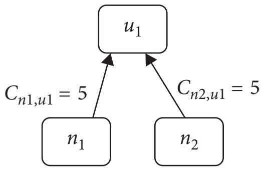


Figure 3: Example of NC-graph.

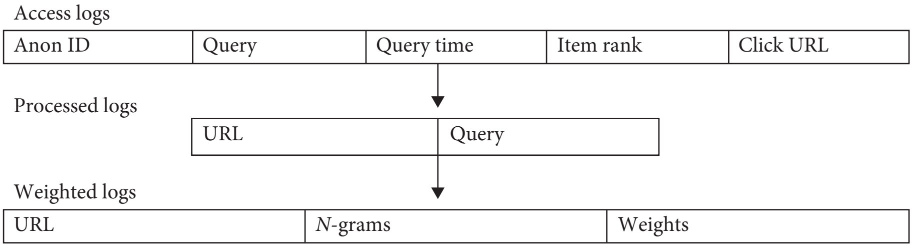


Figure 4: Schema of logs used for proposed approach.

Table 2: Attributes of schema and their description.

Attribute	Description
AnonID	An anonymous user ID number
Query	The query issued by the user
QueryTime	The time at which the query was submitted for search
ItemRank	If the user clicked on a search result, the rank of the item on which they clicked is listed
ClickURL	If the user clicked on a search result, the domain portion of the URL in the related work is listed
N-grams	Parsed query in *N*-grams
Weights	Count of a query clicked for URL

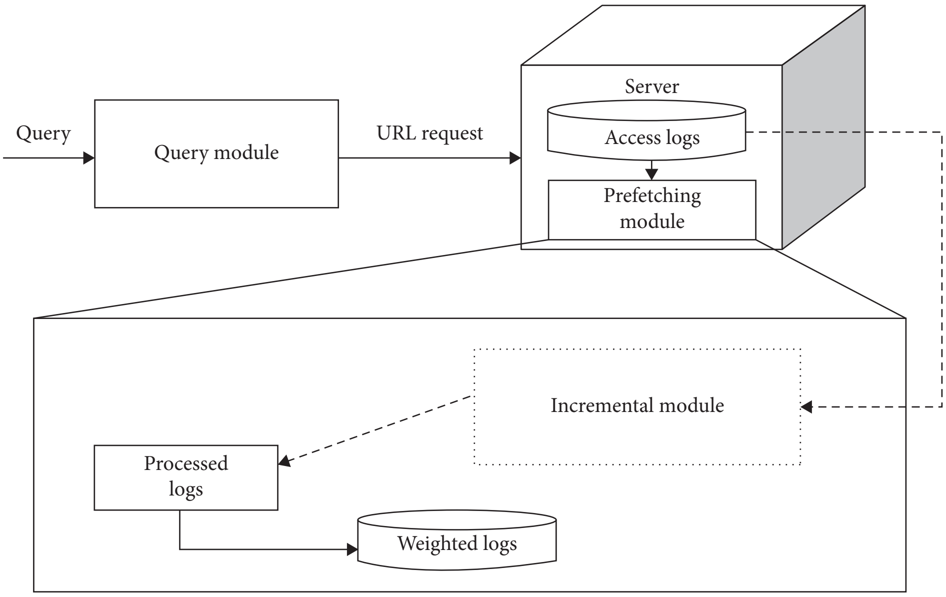


Figure 5: Incremental Module.

3.3. Pseudocode for Proposed Algorithm. The pseudocode for the proposed approach is as: Given in Algorithms 1–6.

4. Example Illustration

This section explains the offline and an online phase steps with the help of some sample of URLs, submitted queries present in the processed logs, and their respective clicks, i.e., the number of times URL has been clicked.

4.1. Preprocessing Phase

(i) In the first phase, preprocessing is done by removing stop words. A sample of preprocessed logs is shown in Table 3.

4.2. Bipartite Graph Generation Phase

(i) Calculate click count $C_{q,u}$ for each pair of query q and URL $u<q \in Q,\ u \in U>$ using processed logs. After calculating the click counts, a Query-URL click-graph (C-graph) is generated as discussed in step 5 of algorithm BipartiteGraphGen (); e.g., let $<q_1, u_1>$ edge is created with label $C_{q1,u1}$, i.e., 10. Similarly, $<q_5, u_1>$ and $<q_8, u_1>$ edges are created with labels 10 and 5, respectively.

(ii) Further in step 7 of BipartiteGraphGen (), the queries are parsed into N-grams by using $n = 3$ as shown in Figure 2; e.g., q_5 is parsed into 3-grams (gov, college, gov-college).

(iii) According to the algorithm's next step 8, N-gram associated click-graph (NC-graph) is generated as depicted in Figure 6.

4.3. Weight Calculation Phase

(i) The same click count is assigned to each N-gram in the query for each URL based on click count of queries as in step 6 of WeightCalculator(), e.g., with the URL u_1 associated queries, and their labels are $q_1 \longrightarrow 10$, $q_5 \longrightarrow 10$, $q_8 \longrightarrow 5$.

Against each query, parsed N-grams are $q_1 \longrightarrow \{\text{ymca}\}$, $q_5 \longrightarrow \{\text{gov, college, gov-college}\}$, $q_8 \longrightarrow \{\text{best, college, best-college}\}$.
Thus, each N-gram will get the respective label of its query, i.e., (ymca:10), (gov: 10, college: 10, gov-college:10), (best:5, college:5, best-college:5).

(ii) In the next step, weights are assigned to each distinct N-gram associated with URL u in NC-graph by adding click count of the N-grams coming from different queries for that URL; e.g., weighted N-grams corresponding to URL u_1 are (ymca: 10, gov: 10, college: 15, gov-college: 10, best: 5, best-college: 5)

(iii) Perform normalization as in step 10 of Weight-Calculator() $W_{\text{ymca},u1} = 10/(10 + 10+15 + 10+5 + 5) = 0.22$. The normalized weighted N-grams for their respective URLs are shown in Figure 7.

4.4. Online Phase

(i) In the online phase, when the user submits a query, e.g., "ncrgov college," it is parsed in 3-grams as discussed in step 3 of Matcher () algorithm and shown in Figure 7.

(ii) Further, weights of URLs are calculated corresponding to the user's query as per step 7 of the Matcher() algorithm, e.g., $W_{u1} = 0 + 0.22 + 0.33 + 0.22 + 0 + 0 = 0.77$. To calculate the weight of u_1, weights of the user's query terms (ncr, gov, college, ncr-gov, ncr-college, gov-college, ncr-gov-college) are taken from the weighted N-grams of the URL u_1: (ymca: 0.22, gov: 0.22, college: 0.33, gov-college: 0.22, best: 0.11, best-college: 0.11) if they are present in that URL; otherwise, it is considered 0.

(iii) Based on the calculated weights of URLs, the system gives the prioritized list of URLs, as depicted in Figure 8. For further processing, the prioritized list will be passed to the prefetching engine.

Thus, the proposed approach predicts by considering the content information and the information collected using logs instead of directly deriving the frequent patterns from the access logs. Therefore, this process indicates those web pages that are not frequently visited before making more accurate predictions.

In the next section, the proposed approach's performance evaluation is carried out with a unigram approach. It has been observed that the proposed hybrid approach significantly improves performance.

5. Experimental Evaluation

The effectiveness of the proposed prediction model is illustrated by implementing and testing with a large dataset. To explore the performance of prediction, Microsoft Visual Studio 12.0 in conjunction with SQL server 2012 is used. In this section, we first list the measures for the performance evaluation of prediction and then present the impact of the n-grams followed by comparing experimental results.

5.1. Training and Testing Data. To run the experimental cases, American Online (AOL) search logs are collected for three months spanning from 01 March 2006 to 31 May 2006. This dataset consists of 20 M web queries collected from 650 k users over three months. The dataset [35] includes (AnonID, Query, QueryTime, ItemRank, ClickURL).

```
Input: access logs (AL)
Output: Weighted $N$-grams stored in weighted logs (WL) of order $m \times n$
Begin
    Read (AL);
    PL ← Preprocess (AL);//PL = Processed Logs
    NC-graph ← BipartiteGraphGen (PL); //NC-graph = N-gram associated click-graph
    WL ← WeightCalculator (NC-graph); //W Lis weighted logs stored in form of $m \times n$ weight matrix
    Return (WL);
End
```

Algorithm 1: Weight generator.

```
Input: access logs (AL)
Output: processed logs (PL)
Begin
    Read AL;
    Extract session id, query, clicked URL from AL;
    PL ← Remove stop words from each log record;
    Return PL;
End
```

Algorithm 2: Preprocess.

```
Input: processed logs (PL)
Output: $N$-gram associated click-graph (NC-graph)
Begin
    Read (PL);
    Q ← Read queries from PL;
    U ← Read URLs from PL;
    Calculate click count $C_{q,u}$ for each pair $\langle q \in Q, u \in U \rangle$ using PL;
    $C$-graph ← create an edge between $\langle q, u \rangle$ with label $C_{q,u}$;
    For each query $q \in Q$ do
    $N_{\mathrm{q}}$ ← Parser ($q$); //parsing of query into $N$-grams
    NC-graph ← Create an edge between $\langle q, N_q \rangle$
    EndFor
    Return (NC-graph);
End
```

Algorithm 3: BipartiteGraphGen.

```
Input: query $q$
Output: $N$-grams associated with query ($q$), i.e., ($N_{\mathrm{q}}$)
Begin
    Read $q$;
    $N_{\mathrm{q}}$ ← Extract $N$-grams from $q$;
    Return $N_{\mathrm{q}}$;
End
```

Algorithm 4: Parser.

```
Input: $N$-gram associated click-graph (NC-graph)
Output: weighted $N$-grams corresponding to distinct URLs stored in matrix WL
Begin
    Create a matrix WL of order $m \times n$//$m \longrightarrow$ no. of distinct N-grams of all the queries of PL and $n \longrightarrow$ no. of URLs of PL
    $W_{i,j} = 0$;//elements of WL
    For each URL $u \in U$ in NC-graph do
        $W_{n,u} = 0$//weight of $N$-gram associated with query $q$ corresponding to URL u
        For each $N$-gram $n \in N_q$ in NC-graph do
            $C_{n,u} = C_{q,u}$; //$n \in N_q$
            $w_{n,u} + = C_{n,u}$;
        End For
        For each $N$-gram $n \in N_q$ in NC-graph do
            $W_{n,u} = w_{n,u} / \sum_{v \in Vu} C_{v,u}$ and $V_u = \{V \in N_q : N_q \in \langle q, u \rangle\}$//normalization of calculated weights
            Store in WL;
        EndFor
    EndFor
    Return WL;
End
```

Algorithm 5: Weight calculator.

```
Input: user's query (UQ), weighted logs (WL)
Output: prioritized URLs List (PUL)
Begin
    PUL = Ø
    Read UQ;
    T ← Parser (UQ);
    For each URL $u \in U$ in WL do
        $W_u = 0$//weight of URL $u$
        For each term $t \in T$ do
            $W_u = \sum_{t \in T} W_{t,u} * I_{t,u}$
        EndFor
    EndFor
    If $W_u! = 0$
        PUL = PUL ∪ $u$
    Sort elements of PUL;
    Return PUL;
End
In the next section, an example concerning the above-proposed work is presented.
```

Algorithm 6: Matcher.

The dataset is divided into two subsets, one for training and the other for testing in the proportion of 80 : 20. The training set has been used to build a prediction model while a testing set comprising various query sets has been used to run multiple test cases. A snapshot of the web access logs is displayed in Figure 9.

5.2. Implementation. Initially, access log file is preprocessed to extract the meaningful entries such as queries and the requested URL and removal of stop words is done. Further queries are parsed into N-grams as shown in Figure 10.

In the next step, weights are assigned to the N-grams. Further, weights are normalized, which is the output of the offline phase, as shown in Figure 11.

In the online phase, when the user submits the query to the server, the prefetching module is also used to predict the user's behavior. A list of prioritized URLs has been given by the online phase to be fetched in the cache before the user's request, as shown in Figure 12.

5.3. Performance Evaluation. In literature [33, 36], prediction performance is measured using two primary

Table 3: Sample of preprocessed logs.

URL	Query after removing stop words
http://www.ymcaust.in	Ymca
http://www.amity.edu	Ncr college
http://www.ymcaust.in	Gov college
http://www.galgotias.org	Top university
http://www.gdgoenka.edu	Ncr college
http://www.ymcaust.in	Ymca
http://www.amity.edu	Amity
http://www.gdgoenka.edu	Top university
http://www.galgotias.org	Galgotias
http://www.amity.edu	Best college
http://www.amity.edu	Amity
http://www.ymcaust.in	Ymca
http://www.gdgoenka.edu	Top university
http://www.galgotias.org	Galgotias
http://www.amity.edu	Best college
http://www.amity.edu	Amity
http://www.ymcaust.in	Ymca
http://www.amity.edu	Ncr college
. . .	. . .

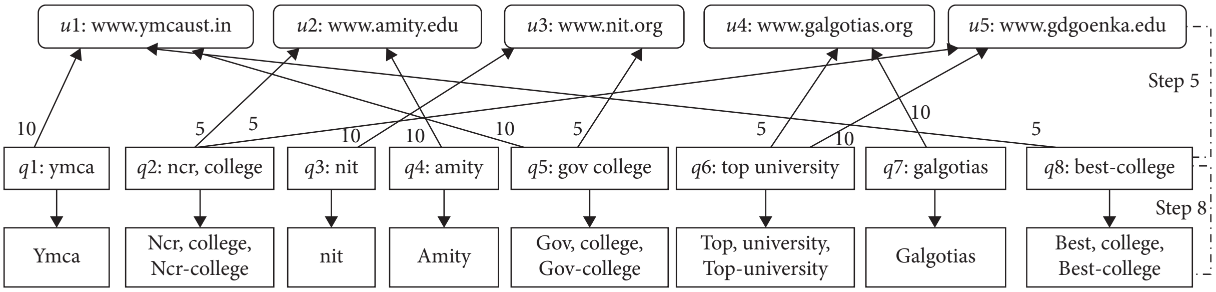
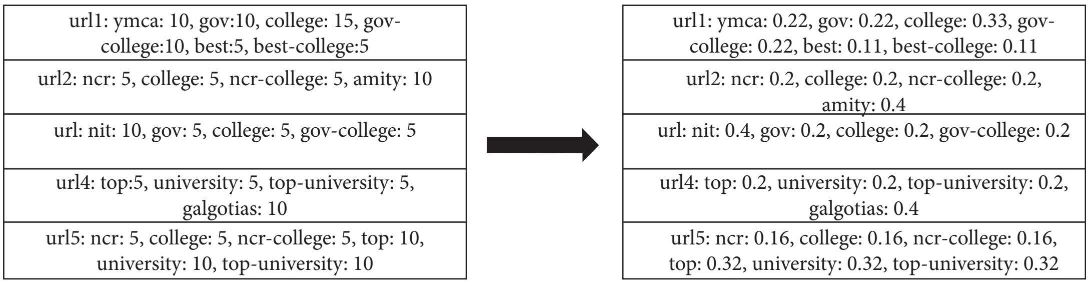


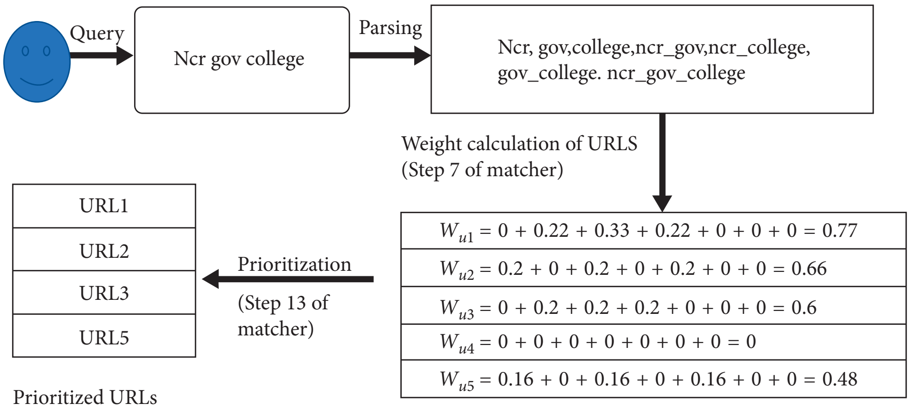
Figure 6: Generation of NC-graph.



url1: ymca: 10, gov:10, college: 15, gov-college:10, best:5, best-college:5
url2: ncr: 5, college: 5, ncr-college: 5, amity: 10
url: nit: 10, gov: 5, college: 5, gov-college: 5
url4: top:5, university: 5, top-university: 5, galgotias: 10
url5: ncr: 5, college: 5, ncr-college: 5, top: 10, university: 10, top-university: 10

url1: ymca: 0.22, gov: 0.22, college: 0.33, gov-college: 0.22, best: 0.11, best-college: 0.11
url2: ncr: 0.2, college: 0.2, ncr-college: 0.2, amity: 0.4
url: nit: 0.4, gov: 0.2, college: 0.2, gov-college: 0.2
url4: top: 0.2, university: 0.2, top-university: 0.2, galgotias: 0.4
url5: ncr: 0.16, college: 0.16, ncr-college: 0.16, top: 0.32, university: 0.32, top-university: 0.32

Figure 7: Generation of normalized weights.


Query
Ncr gov college
Parsing
Ncr, gov,college,ncr_gov,ncr_college, gov_college. ncr_gov_college
Weight calculation of URLS (Step 7 of matcher)
Prioritization
(Step 13 of matcher)
Prioritized URLs

$W_{u1} = 0 + 0.22 + 0.33 + 0.22 + 0 + 0 + 0 = 0.77$

$W_{u2} = 0.2 + 0 + 0.2 + 0 + 0.2 + 0 + 0 = 0.66$

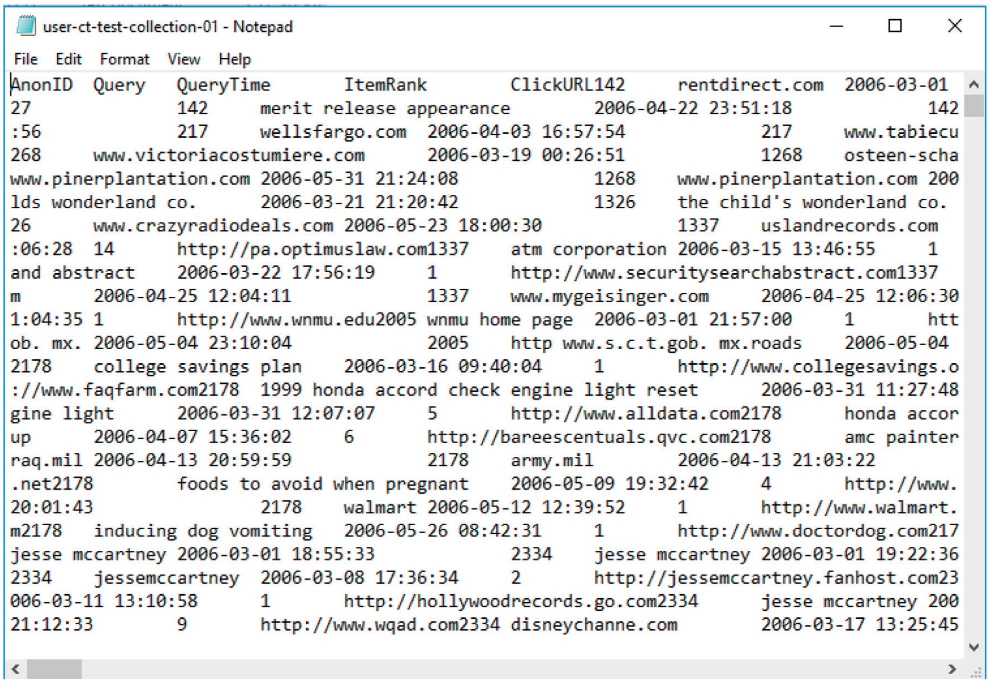
$W_{u3} = 0 + 0.2 + 0.2 + 0.2 + 0 + 0 + 0 = 0.6$

$W_{u4} = 0 + 0 + 0 + 0 + 0 + 0 + 0 = 0$

$W_{u5} = 0.16 + 0 + 0.16 + 0 + 0.16 + 0 + 0 = 0.48$

URL1
URL2
URL3
URL5

Figure 8: Generation of prioritized URLs based on the users' given query.



```
AnonID  Query    QueryTime        ItemRank        ClickURL142     rentdirect.com  2006-03-01
27               142     merit release appearance         2006-04-22 23:51:18             142
:56              217     wellsfargo.com  2006-04-03 16:57:54              217     www.tabiecu
268     www.victoriacostumiere.com      2006-03-19 00:26:51              1268    osteen-scha
www.pinerplantation.com 2006-05-31 21:24:08              1268    www.pinerplantation.com 200
lds wonderland co.      2006-03-21 21:20:42              1326    the child's wonderland co.
26      www.crazyradiodeals.com 2006-05-23 18:00:30              1337    uslandrecords.com
:06:28  14       http://pa.optimuslaw.com1337    atm corporation 2006-03-15 13:46:55     1
and abstract     2006-03-22 17:56:19     1       http://www.securitysearchabstract.com1337
m       2006-04-25 12:04:11              1337    www.mygeisinger.com     2006-04-25 12:06:30
1:04:35 1        http://www.wnmu.edu2005 wnmu home page  2006-03-01 21:57:00     1       htt
ob. mx. 2006-05-04 23:10:04              2005    http www.s.c.t.gob. mx.roads    2006-05-04
2178    college savings plan     2006-03-16 09:40:04     1       http://www.collegesavings.o
://www.faqfarm.com2178  1999 honda accord check engine light reset      2006-03-31 11:27:48
gine light       2006-03-31 12:07:07     5       http://www.alldata.com2178      honda accor
up      2006-04-07 15:36:02     6       http://bareescentuals.qvc.com2178       amc painter
raq.mil 2006-04-13 20:59:59              2178    army.mil        2006-04-13 21:03:22
.net2178         foods to avoid when pregnant    2006-05-09 19:32:42     4       http://www.
20:01:43                 2178    walmart 2006-05-12 12:39:52     1       http://www.walmart.
m2178   inducing dog vomiting    2006-05-26 08:42:31     1       http://www.doctordog.com217
jesse mccartney 2006-03-01 18:55:33              2334    jesse mccartney 2006-03-01 19:22:36
2334    jessemccartney   2006-03-08 17:36:34     2       http://jessemccartney.fanhost.com23
006-03-11 13:10:58       1       http://hollywoodrecords.go.com2334      jesse mccartney 200
21:12:33         9       http://www.wqad.com2334 disneychanne.com        2006-03-17 13:25:45
```

Figure 9: A snapshot of the web access logs.

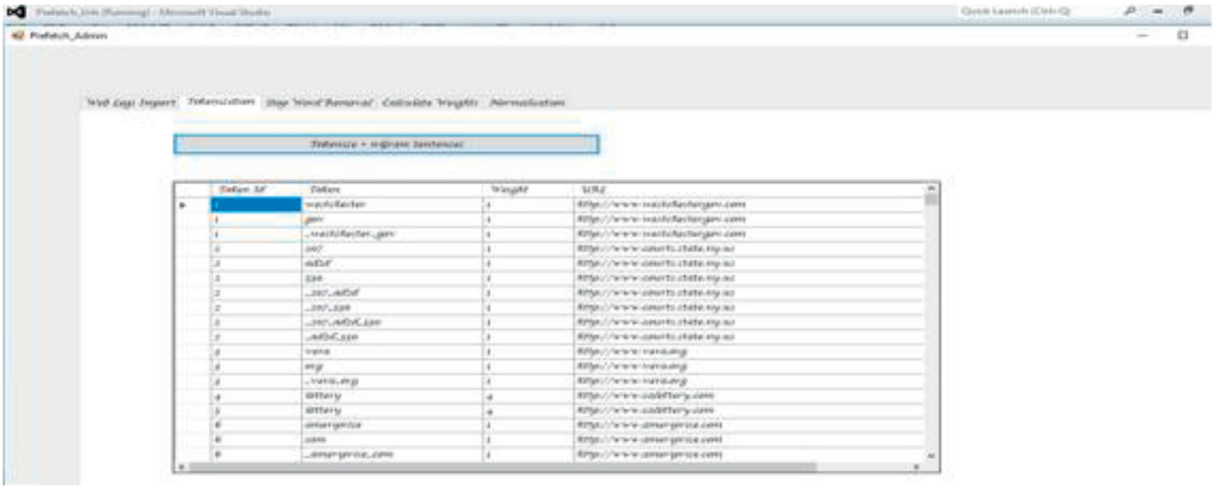


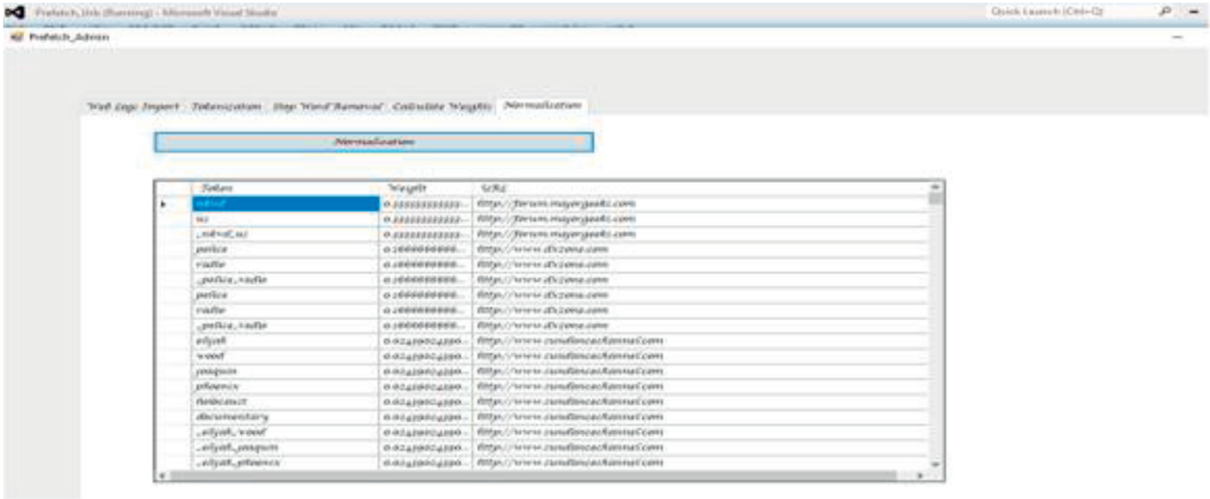
Figure 10: Parsing queries into N-grams.

performance metrics: precision and hit ratio. In our work also, we have used these parameters to measure the accuracy of prediction:

(i) Precision: precision is useful to measure how probable a user will access one of the prefetched pages. Precision is calculated by taking the percentage of the total number of requests found in the cache to the number of predictions.

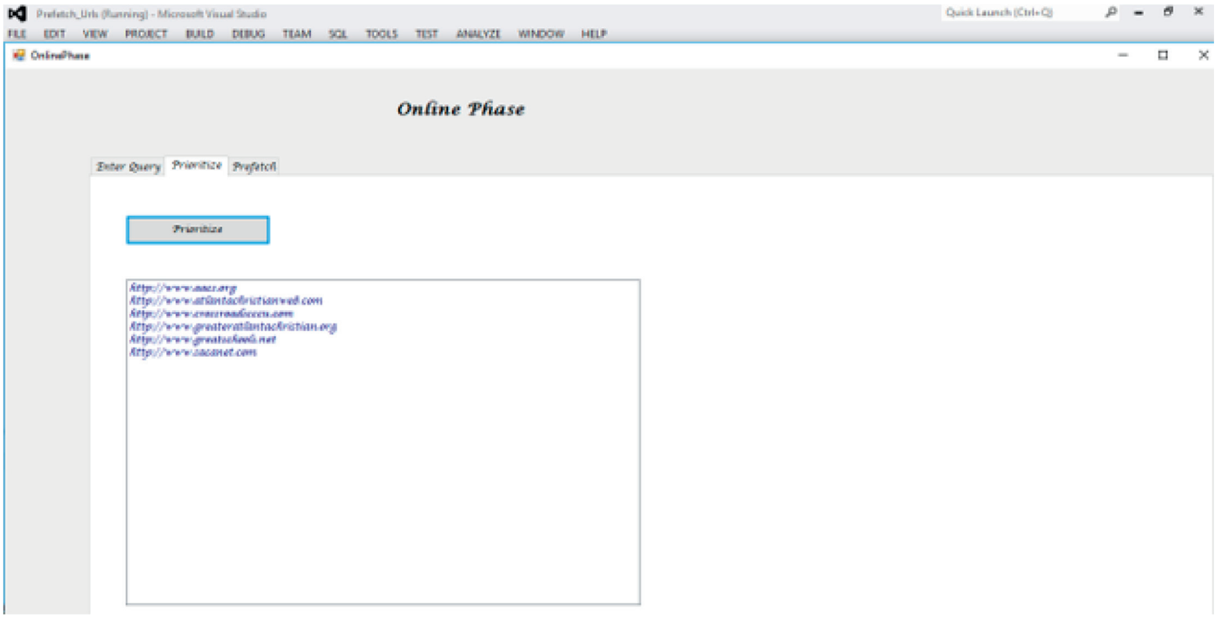
$$\text{precision} = \frac{\text{total number of requests fetched by the cache}}{\text{total predictions}}. \tag{4}$$

(ii) Hit ratio: hit ratio is useful to measure the probability of the user's request fulfilled by the



	Token	Weight	URL
▸	[illegible]	0.333333333333...	http://forum.mayorgeeks.com
	ui	0.333333333333...	http://forum.mayorgeeks.com
	_[illegible]_ui	0.333333333333...	http://forum.mayorgeeks.com
	police	0.166666666666...	http://www.dx2one.com
	radio	0.166666666666...	http://www.dx2one.com
	_police_radio	0.166666666666...	http://www.dx2one.com
	police	0.166666666666...	http://www.dx2one.com
	radio	0.166666666666...	http://www.dx2one.com
	_police_radio	0.166666666666...	http://www.dx2one.com
	elijah	0.02439024390...	http://www.sundancechannel.com
	wood	0.02439024390...	http://www.sundancechannel.com
	joaquin	0.02439024390...	http://www.sundancechannel.com
	phoenix	0.02439024390...	http://www.sundancechannel.com
	holocaust	0.02439024390...	http://www.sundancechannel.com
	documentary	0.02439024390...	http://www.sundancechannel.com
	_elijah_wood	0.02439024390...	http://www.sundancechannel.com
	_elijah_joaquin	0.02439024390...	http://www.sundancechannel.com
	_elijah_phoenix	0.02439024390...	http://www.sundancechannel.com

Figure 11: Weighted *N*-grams.



Online Phase

http://www.aacc.org
http://www.atlantachristianweb.com
http://www.crossroadsccess.com
http://www.greateratlantachristian.org
http://www.greatschools.net
http://www.sacanet.com

Figure 12: Online phase: prioritized list of URLs.

prefetched pages in the cache. Hit ratio is calculated by taking a percentage of the total number of requests found in the cache to the total number of users' requests.

$$\text{hit ratio} = \frac{\text{total number of requests fetched by cache}}{\text{total users' requests}}. \quad (5)$$

5.3.1. Observation: Impact of N-Grams. This subsection compares the proposed model with *N*-grams against the unigrams approach on the same query sets. Multiple test cases were run by setting up the different thresholds for prefetching. Here, the threshold is a fixed number of pages that are going to be prefetched. On an experimental basis, a broad scale of threshold has been taken. Test cases are discussed as follows:

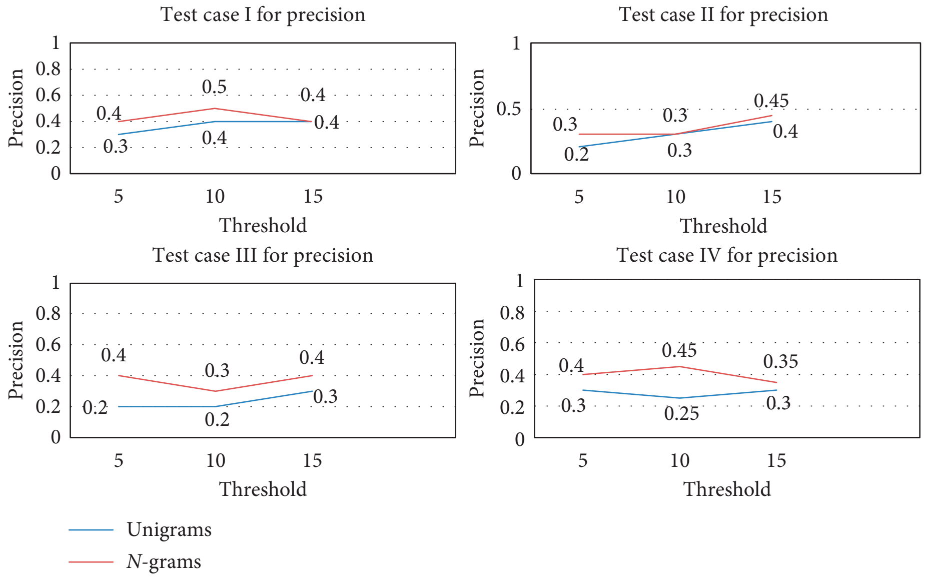


Figure 13: Precision comparison of *N*-grams and unigrams.

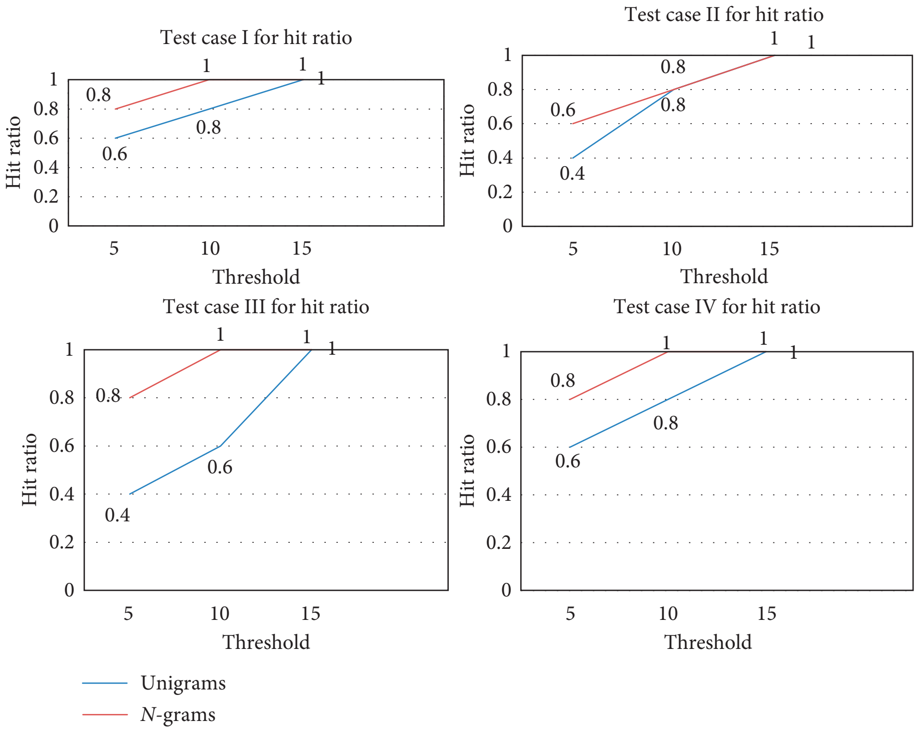


Figure 14: Hit ratio comparison of *n*-grams and unigrams.

Table 4: Comparison of unigrams and *n*-grams results for various threshold values.

	Threshold value	Unigram (%)	*N*-gram (%)	Increase % (%)
Precision	Threshold = 5	25	37	12
	Threshold = 10	28	38	10
	Threshold = 15	35	40	5
Hit ratio	Threshold = 5	50	70	20
	Threshold = 10	70	90	20
	Threshold = 15	100	100	0

Table 5: Comparison of latency.

Average time taken		Reduction (%) in time
Without prefetch	With prefetch	
751	245	**50.6**

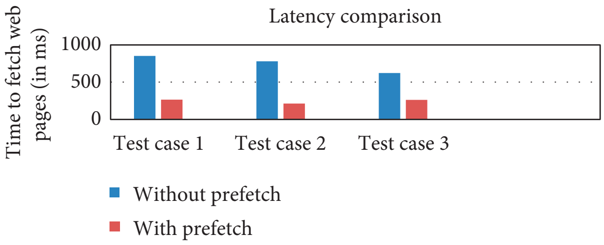


Figure 15: Latency comparison with *n*-grams prediction model.

Test case I: test the effectiveness of HPM by taking a query having two keywords. Two-keyword-based queries have been extracted from the same AOL logs to run the test case, and an approximate 55000 queries appropriate for this test case were found.

Test case II: test the effectiveness of HPM by taking a query having five keywords. Five-keyword-based queries have been extracted from the same AOL logs to run the test case, and an approximate 65000 queries appropriate for this test case were found.

Test case III: test the effectiveness of HPM by taking a query having eight keywords. Eight-keyword-based queries have been extracted from the same AOL logs to run the test case, and an approximate 50000 queries appropriate for this test case were found.

Test case IV: test the effectiveness of HPM by taking a query having more than ten keywords. Ten-or-more-keyword-based queries have been extracted from the same AOL logs to run the test case, and an approximate 20000 queries appropriate for this test case were found.

All the test cases were run by taking unigrams as well as *N*-grams of the query. Based on this, precision and hit ratio curves were plotted to evaluate the proposed model, as shown in Figures 13 and 14, respectively.

In general, models with *N*-grams yield better results than the unigrams in terms of both measures, i.e., precision and hit ratio.

It can be observed from the above graphs that the results of the HPM are much better with an approximately 9% increase on average in precision and about a 13% increase on average in the HIT ratio, as depicted in Table 4. This implies that when the threshold value is less, i.e., the window to fetch the pages for prefetching is small, better precision and hit ratio are achieved in the case of *N*-grams as compared to unigrams, although when the prefetch threshold increases up to 15, both cases' performance is the same. But the number of prefetches is more in this case, which is not a practical solution. Thus, we can conclude that our system performs better to yield the optimal results in fetching the relevant web pages while consuming less network bandwidth.

5.3.2. Observation: Impact on Latency. A series of test cases comprising the query sets from the testing set of the access logs were run with different inputs, and it is observed that, by using HPM for prefetching, the time taken to fetch the web pages is almost reduced to half of that without prefetching as shown in Table 5. Hence, latency reduction has also been achieved in an impactful manner. The same is shown in Figure 15.

The results of the graph given in Figure 15 are evaluated in Table 5.

5.3.3. Comparison between Web Usage Mining, Web Content Mining, and Hybrid Model. A comparison between these three has been made with various test cases. A series of test cases were run for several types of sessions, i.e., smaller to longer sessions. In our experiments, association rule mining and Markov model-based technique [11] have been used for the WUM technique, and the keyword-based approach [20] has been used for WCM. The proposed model performed well compared to the other two, as shown in Figure 16.

From experiments, it has been concluded that web usage mining and web content mining may perform better in longer user sessions, but in smaller sessions, these techniques do not perform well. Because usage mining-based methods make their predictions based on URLs' sequences, the longer the sequences, the better the results. Similarly, content mining-based strategies learn the user's behavior as they start surfing, and longer sessions provide better learning. However, the proposed hybrid prediction model performs well in smaller as well as longer sessions. From the graphs depicted in Figure 16, we evaluate the results in Table 6.

From the results, it can be summarized that our approach, i.e., hybrid prediction model, clearly provides better results with an approximately 26% increase on average in

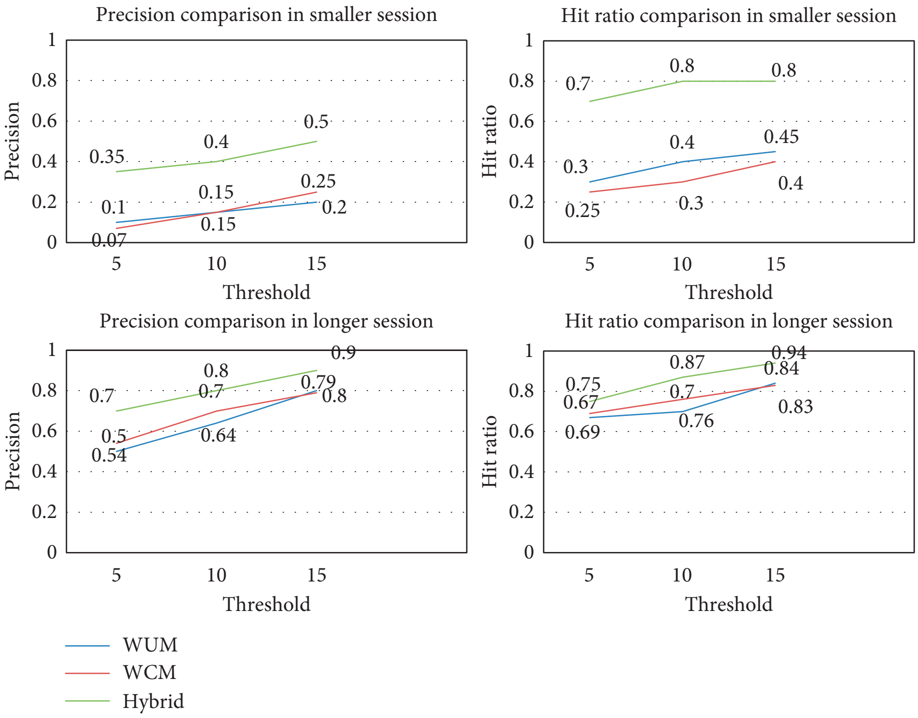


Figure 16: Comparison between WUM, WCM, and hybrid prediction model.

Table 6: Comparison of WUM, WCM, and hybrid approach for precision and hit ratio.

	WUM (%)	Hybrid (%)	Increase (%)	WCM (%)	Hybrid (%)	Increase (%)
Precision	15	41	**26**	15	41	**26**
Hit ratio	41	55	**14**	50	55	**5**

precision and almost an average of roughly 10% increase in HIT ratio.

6. Conclusion and Future Work

Predicting users' behavior in a web application has been a critical issue in the past several years. This work presented a hybrid prediction model that integrates the history-based approach with the content-based approach. History information such as user's accessed web pages is collected from access logs. Our proposed model used Query-URL click-graph derived from the access logs by using queries submitted by the users in the past and corresponding clicked URLs. This Query-URL click-graph is represented in the form of a bipartite graph. *N*-grams are generated by parsing the queries in 3-grams to give more weightage to those *N*-grams which frequently come together and are assigned weights for each URL, and URLs are prioritized by considering the query submitted by the user. The prediction model is efficient and predicts URLs based on content and history. Experimental results have shown a significant improvement in precision of 26% and hit ratio of 10%.

Future work will be devoted to the following:

(i) The prediction model developed so far precisely matches the query terms of the user's interest with the weighted logs. It would be useful to enhance the weighted logs with semantics so that semantics of content could be analyzed to increase the precision and hit ratio further.

(ii) A threshold module will be introduced to dynamically calculate the threshold value based on the server load to optimize the network bandwidth while prefetching.

Data Availability

Data are available upon request to the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] T. M. Kroeger, D. D. E. Long, and J. Mogul, "Exploring the bounds of web latency reduction from caching and prefetching," in *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, pp. 13–22, Monterey, CA, USA, December 1997.

[2] V. Almeida, A. Bestavros, M. Crovella, and A. de Oliveira, "Characterizing reference locality in the WWW," in *Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems*, pp. 92–103, Miami Beach, FL, USA, December 1996.

[3] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, “Changes in web client access patterns: characteristics and caching implications,” *World Wide Web: Special Issue on Characterization and Performance Evaluation*, vol. 2, no. 1-2, pp. 15–28, 1999.

[4] S. K. Pal, V. Mitra, and P. Mitra, “Web mining in soft computing framework: relevance, state of the art and future directions,” *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1163–1177, 2002.

[5] R. Suguna and D. Sharmila, “An overview of web usage mining,” *International Journal of Computer Applications*, vol. 39, no. 13, pp. 11–13, 2012.

[6] O. Kumar and P. Bhargavi, “Analysis of web server log by web usage mining for extracting users patterns,” *International Journal of Computer Science Engineering and Information Technology Research*, vol. 3, no. 2, pp. 123–136, 2013.

[7] N. Goel, S. Gupta, and C. K. Jha, “Analyzing web logs of an astrological website using key influencers,” *International Research Journal*, vol. 5, no. 1, pp. 2–11, 2015.

[8] D. Lee, “Methods for web bandwidth and response time improvement,” in *World Wide Web: Beyond the Basics*, M. Abrams, Ed., Prentice Hall, Upper Saddle River, NJ, USA, 1998.

[9] M. Deshpande and G. Karypis, “Selective Markov models for predicting web page accesses,” *ACM Transactions on Internet Technology*, vol. 4, no. 2, pp. 163–184, 2004.

[10] D. Kim, N. Adam, I. Im, V. Atluri, M. Bieber, and Y. Yesha, “A clickstream-based collaborative filtering personalization model: towards a better performance,” in *Proceedings of the 6th Annual International Workshop on Web Information and Data Management*, pp. 88–95, ACM, Washington, DC, USA, November 2004.

[11] J. Verma, A. Sharma, and G. Amit, “A novel approach to determine the rules for web page prediction using dynamically chosen *K*-order Markov models,” *International Journal of Research in Computer and Communication Technology*, vol. 2, no. 12, 2013.

[12] S. G. Oguducu and M. T. Ozsu, “A web page prediction model based on click-stream tree representation of user behavior,” in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2003.

[13] L. Lu, M. Dunham, and Y. Meng, “Discovery of significant usage patterns from clusters of clickstream data,” in *Proceedings of the WebKDD’05*, pp. 139–142, ACM, Chicago, IL, USA, August 2005.

[14] M. A. Awad and I. Khalil, “Prediction of user’s web-browsing behavior: application of Markov model,” *IEEE Transactions on Systems, Man, And Cybernetics—Part B: Cybernetics*, vol. 42, no. 4, pp. 1131–1142, 2012.

[15] W. Zou, J. Won, J. Ahn, and K. Kang, “Intentionality-related deep learning method in web prefetching,” in *Proceedings of the 2019 IEEE 27th International Conference on Network Protocols (ICNP)*, pp. 1-2, Chicago, IL, USA, October 2019.

[16] M. Joo and W. Lee, “WebProfiler: user interaction prediction framework for web applications,” *IEEE Access*, vol. 7, pp. 154946–154958, 2019.

[17] J. Martínez-Sugastti, F. Stuardo, and V. González, “Web browsing optimization: a prefetching system based on prediction history,” in *Proceedings of the 2017 XLIII Latin American Computer Conference (CLEI)*, pp. 1–10, Cordoba, Argentina, September 2017.

[18] K. M. Veena and R. M. Pai, “Clustering of web users’ access patterns using a modified competitive agglomerative algorithm,” in *Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 701–707, Udupi, India, September 2017.

[19] P. Venketesh, “Semantic web prefetching scheme using Naïve Bayes classifier,” *International Journal of ComputerScience and Applications*, vol. 7, no. 1, pp. 66–78, 2010.

[20] S. Setia, V. Jyoti, and N. Duhan, “A novel approach for semantic web prefetching using semantic information and semantic association,” in *Big Data Analytics*, pp. 471–479, Springer, Singapore, 2018.

[21] T. T. S. Nguyen, H. Y. Lu, and J. Lu, “Webpage recommendation based on web usage and domain knowledge,” *IEEE Transactions on Knowledge And Data Engineering*, vol. 26, no. 10, pp. 2574–2587, 2014.

[22] Y. Hu, C. Kang, J. Tang, D. Yin, and Yi Chang, “Large-scale location prediction for web pages,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 9, pp. 1902–1915, 2017.

[23] D. Yin, Y. Hu, J. Tang et al., “Ranking relevance in yahoo search,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, August 2016.

[24] Y. Deng and S. Manoharan, “Predicting web accesses using personal history,” in *Proceedings of the 2017 IEEE Conference on Open Systems (ICOS)*, pp. 7–12, Miri, Malaysia, November 2017.

[25] P. M. Bharti and T. J. Raval, “Improving web page access prediction using web usage mining and web content mining,” in *Proceedings of the 2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pp. 1268–1273, Coimbatore, India, June 2019.

[26] Z. Chen, Li Tao, J. Wang, L. Wenyin, and W.-Y. Ma, “A unified framework for web link analysis,” in *Proceedings of the Third International Conference on Web Information Systems Engineering, 2002. WISE 2002*, Singapore, December 2002.

[27] B. D. Davison, “Topical locality in the web,” in *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval—SIGIR’00*, Athens, Greece, July 2000.

[28] J. M. Kleinberg, “Authoritative sources in a hyperlinked environment,” in *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, Francisco, CL, USA, January 1998.

[29] January 2020 http://www.directhit.com.

[30] A. Sheshasaayee and V. Vidyapriya, “A framework for an efficient knowledge mining technique of web page reorganisation using splay tree,” *Indian Journal of Science and Technology*, vol. 8, no. 29, pp. 11–15, 2015.

[31] D. A. Vadeyar and H. K. Yogish, “Farthest first clustering in links reorganization,” *International Journal of Web and Semantic Technology*, vol. 5, no. 3, pp. 17–21, 2014.

[32] M. B. Thulase and G. T. Raju, “Website reorganization for effective latency reduction through splay trees and concept-based clustering,” *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, vol. 325, pp. 173–182, 2015.

[33] C. D. Gracia and S. Sudha, “A case study on memory efficient prediction models for web prefetching,” in *Proceedings of the International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS)*, pp. 1–6, Pudukkottai, India, February 2016.

[34] S. Kalaivani and K. Shyamala, “A novel technique to preprocess web log data using SQL server management Studio,”

International Journal of Advanced Engineering, Management and Science, vol. 2, no. 7, pp. 973–977, 2016.

[35] January 2020, http://www.researchpipeline.com/mediawiki/index.php?title=AOL_Search_Query_Logs.

[36] C.-Z. Xu and T. I. Ibrahim, "A keyword-based semantic prefetching approach in internet news service," *Journal of IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 5, 2004.