Global Journal of Scientific Researches
Available online at gjsr.blue-ap.org
©2020 GJSR Journal. Vol. 8(1), pp. 1-7, 27 April, 2020
E-ISSN: 2311-732X

Presentation of a Novel Technique for Data Quality Improvement of Commercial Proxy Log for the Use of Efficient web mining

Akbar Keshavarzpour[1*], Kimia Bazargan Lari[2] and Haleh Homayouni[3]

1- Master of Computer Software Engineering, Department of Computer Engineering, Higher Education Institute of Apadana, Shiraz, Iran
2- Ph.D of Artificial intelligence, Department of Computer Engineering, Higher Education Institute of Apadana, Shiraz, Iran
3- Ph.D Student of Artificial intelligence, Department of Computer Engineering, Higher Education Institute of Apadana, Shiraz, Iran

Corresponding Author: Akbar Keshavarzpour

Received: 30 March, 2020 *Accepted: 14 April, 2020* *Published: 27 April, 2020*

ABSTRACT

Data Cleaning is a stage taken in the preprocessing of Web Mining, and is widely used in most Data Mining systems. Although many efforts have been made for data clearing of Web Server Logs, but there are some questions and ambiguities yet unanswered about Enterprise Proxy. With limited access to web sites, Enterprise Proxies trace web request from various clients to various web servers, which differentiates them from Web Server Logs in both location and content; therefore, most Irrelevant items like Software update requests cannot be filtered by the traditional methods of Data Cleaning. In this article we initially propose a method named EPLogCleaner that can filter out large number of irrelevant items based on the common prefix of their URLs, in this regard we do an evaluation on EPLogCleaner with a real network traffic trace acquired from an enterprise proxy. The experimental results show that our proposed method (EPLogCleaner) can improve the data quality of Enterprise Proxy logs by filtering over 30% of URL requests of them, comparing with the traditional data cleaning methods.

Keywords: Web Mining, Data Mining, Data Cleaning, Enterprise Proxy Logs.

INTRODUCTION

Exploration of Enterprise Proxy Logs play an important role for business managers and employees. As an example this exploration can be applied for the optimization of caching proxy strategies and the foresight of employees' aims and behaviors; besides, this can be used as a data source for analyzing abnormal pattern of behavior, and also help to detect internal security threats; nevertheless, considering that the size of enterprise proxy log in rapidly increasing, as a result it makes finding proper and interesting information more difficult.

During overall Preprocessing, web minding and data clearing would be considered as very important stages for the accomplishment, which are widely used in most data mining systems. There are several ways for data quality improvement that have been proposed through omission of irrelevant items as to Jpeg, gif, audio files and also web requests with wrong http response status codes [1,2,3]. Yet most of them are designed for preprocessing web server logs, while few number of articles and researches already done have discussed enterprise proxy logs. Enterprise proxy log as a specific type of proxy log that traces web requests from multiple clients to multiple web servers, Registers the real requests from intranet computers of large enterprise companies to web servers of out of intranet via proxy server. As shown in figure 1, enterprise proxy logs, in comparison to web server logs are in different conditions; therefore, comparing to web server logs show different features. The main locations are shown in the following image.

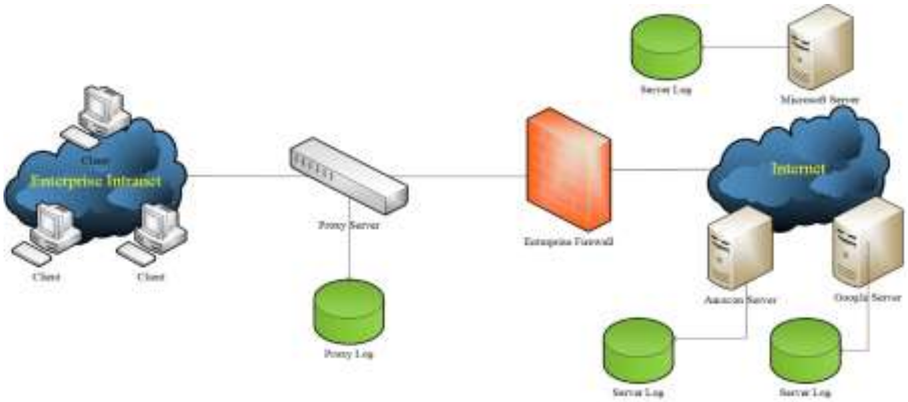


Figure 1. the different location of proxy log in comparison to server log

(1) Multiple Web Servers: web server log is the access log of users surveying in a specific website, while the enterprise proxy log has no limitation in websites. From one side, it helps researchers to access a large amount of records, and to the other these records make the users' behavior pattern difficult. The other advantage of enterprise proxy log is that it provides support of platforms parallization in a way that these parallel platforms can be implemented in multi-processing environments.
(2) Fixed User Group: for server web, the location of user group depends on many factors like of web content updating frequently. But from the perspective of enterprise proxy, most user groups remain fixed for a long duration, which itself helps to trace users' behavior.
(3) unfamiliar with website information: informing enterprise proxy of Classification and typology website pages is impossible, such that this action increases the difficulty of data clearing by enterprise proxy log.
(4) The rapid increase of new pages: like enterprise proxy it not only increases records scales, but also makes data clearing method to be self-adaptive so as to examine these new pages. But for every website, the growth speed of new pages is predictable.
(5) Diversifying: for a website manager, having access to site usually is accompanied with the website title. But when an enterprise proxy log is considered as entrance, the website title is would be non-real. People, when using enterprise proxy log shall define various standards of categorization for the purpose of discovering users' behavior pattern.

Considering all these features, the traditional ways for data clearing of server logs in the preprocessing of enterprise proxy logs is inefficient. In this article we have presented our studies and observations in which relevant items have similar time characteristics in enterprise proxy logs; whereas the irrelevant items might have same prefix in their URLs. According to our evaluations and observations, the first method named as EPLogCleaner is presented which, in comparison to the traditional filtering methods of data clearing, can filter out over 30% of URL requests. In our proposed method, the cost of error rate is very low; the rest of the article is structured in a way that we get to review the related affairs in section 2. In section 3, we would explain about our new filtering method named as EPLogCleaner. In the section 4, the experimental results are presented, and finally in section 5, the results and further studies would be presented.

Tasks related to study

Exploration of web data is one of the most challenging tasks of data mining researchers and data managers, because lots of heterogeneous data with poor structuralism are accessible on the web [4]. Describing data features shows the importance and difficulty of data clearing in web mining. Regarding the difference in data sources, there are 3 features in data clearing. Server log, proxy log and client log. But among all of these data sources, proxy logs are the most complicated and the most vulnerable logs of users' data in the log file [5]. As Z. Pabarskaite [6] has indicated in his article, preprocessing of log files is a very difficult and complicated task, and includes about 80% of web exploration time. In the recent years many works have been done about data clearing of server web logs; nonetheless, data clearing of proxy logs attract less attention than to importance guarantees. In this regard, although there are multiple diverse features, but the method of server logs data clearing is still prominent and enlightening. N. Tyagi et al., [1] provide an Algorithmic technique for data preprocessing in exploring web usage, which considers requests for graphic web contents or any other file that might enter web page; or also trace sessions already done by robots. Althoe this group has discussed the importance of proxy log, but the method it has taken for clearing data has been so simple; besides, D. Tanasa, B. Trousse examined categorization of useless data which need to be cleared, and categorized them into two groups. One was assigned for requests without analyzed sources like images, and multimedia files; the other was assigned for requests made by robots. Nevertheless, some irrelevant information as to requests with wrong status codes can pass their method and develop unnecessary computations in the future task. Regarding this matter, the research group of … [3] has developed a tool named LODAP for preprocessing web log file which provides three preprocessor including data clearing, data structuring and data filtering. Data clearing has been done based on access method, status codes, multimedia things and requests made by robots. By the way, most software updating, and requests for analyzing network behavior might be done after the sub-stage of data clearing; therefore, their technic is inefficient and incomplete. Some researchers are focused on methods of proxy log data clearing; at this end, the research group of Y. Zhang et al [7], explains multiple features among proxy log and server log, which provide a data clearing technic for enterprise proxies. Although the comparison of proxy log and server log is Convincing but, they use a large number of experimental values as threshold in experiment without explanation and theoretical support, which make the experimental result as doubtful. As a result, the existing methods used either inefficient data clearing technic, which ignore proxy log features, or large number of experimental values without detailed discussions. According to long-term observations of enterprise proxy log, this article has focused on data clearing issue, and has proposed a new filtering method named as EPLogCleaner for the improvement of data qualities with omitting irrelevant specific requests which share similar time features of proxy logs. In our proposed method, we concluded that such requests are more regular and periodic than to the others, and can be used as characteristic in distinguishing them from others. Using such features, researchers decrease most irrelevant items in proxy logs, while still guarantee that the remained items are valid.

EPLogCleaner method

Prior to introducing EPLogCleaner, first of all three fundamental assumptions are provided which have been applied in our study

1. in company at least one camputer is on, and at every time is connected to the internet so as to make possible the access to unambiguous HTTP requiests for analisation. This assumption can be easilly confirmes, Because most employees regularly keep their campupers on, eve if they are not present in the company, but they do so for dowloding films at nghit, or to avoid the problem of on /off. In the worst statue we can keep this assumption artificially
2. For every HTTP request made from company at nights, in case there is no possibility for tracing, it can be considered as an automatic request, which is defined as a request that is not established by human active behaviors, but is established by applicable computer programs automatically. Automatic requests among which two exhibitive types are Software Automatic Updating and Kinetic Requests, are irrelevant items. Because they get to no betterment in exploration efficiency; besides, a big part of irrelevant items is in enterprise proxy logs.
3. If an URL get in access by specific automatic requests for several time, every request for URL access, regardless of the time of that, would be automatically met with high probability.

In our research each item of the enterprise log, includes the following 6 fields: client address (Host name, or IP address), request time, access way, URL in access, HTTP statute code and the size of web page. According to 3 above hypothesis, EPLogCleaner consists of 3 stages that are shown in figure 2. The first stage is named standard filter, which utilizes a thorough usage of traditional techniques for filtering some irrelevant items as to multimedia files and wrong accessing information. The main idea is supervising the content of each field and omitting some obviously nonsense items. For example, accessed URL is the identifier of files like jpg, jpeg or avi; also access URLs don't use GET, as a result, HTTP400 statute code is indicator of non-proper request. It's notable that as the authenticity of similar technics has been determined as valid, standard filtering cannot filter out related items [3]. The rest of items will be saved in a file named Standard Log.

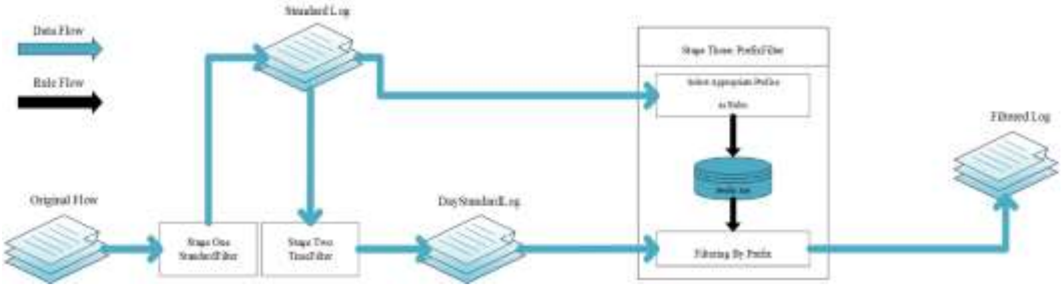


Figure 2. flowchart of processing EPLogCleaner

The parameter of TimeFilter which works in the second stage is aimed to omit non-traceable requests that are produced by computers without human operations, in long durations of time at night. Considering the second hypothesis, we know that it's the want of all automatic requests. It's obvious that each timespan which is considered as nigh timespan is a key factor that influences the authenticity and effectiveness of final filtering. During the time that timespan doesn't include working time; it can be such considered that the filtering result of this stage is without False-Positive. After this stage, all automatic requests made in

Standard Log at night, will be filtered and the remained items will be saved in Day Standard Log file. The aim of the third stage, is omitting automatic requests made during day, which exist in Day Standard Log file. A simple, direct method for getting this aim is monitoring the accessed URL field for each item in Day Standard Log file, and filtering that, in case URL is accessible during nights. However, this method, based on the third assumption which is without False-Positive, is highly inefficient; because there are thousand types of dynamic URL that increases the number of ULs rapidly in time, and the number of URLs is large enough for a Rule Set. for stating the problem, access URL prefixes are used during nights as filtering rules at this stage because there is usually same type of resources with same prefixes; nevertheless, efficiency is along with costs of some wrong filtering of several related items, as a result, there is need for a replacement relationship between efficiency and wrong filtering by accurate choice of proper prefixes as rules. To decrease wring filtering as much as possible, the two following measures are proposed. Firstly a path is chosen which is before the last slash in the URL as its prefix. Since always several URLs are run under one path, and always due to a hierarchy URLs are similar to each other [8]. Secondly the threshold k is introduced, and a prefix is selected as the rule; only when the ratio of night items with similar URL prefixes, and the number of all requests are more than threshold k. As the value of k efficiently impacts on the filtering ratio and accuracy ration, a set of experiments were done to study k parameter and the relationship between k value, filtering ratio and accuracy of the work. In our article the pseudo code of prefixes selection in the algorithm 1 has been shown. In this algorithm Hash Table has been used for operations like search, rapid increase of prefixes; and the method of linked list is used for resolving Hash collision. Each Node in the list is a sample of simple data structure which has 4 members. 1) Prefix, 2) prefix @night (the number of overnight requests that their URLs have same prefixes), 3) Prefix @Day (the number of over-day requests that their URLs have same prefixes), and also 4) Next (indicating the next node of the list)

Experiments Results and Experimental Implementations

In this section, the application of EPLogCleaner is shoed for filtering out irrelevant items; besides, we have evaluated or research based on 2 criteria: Filtering Rate and Precision Rate. The filtering rate is the number of requests filtered by our research, and the number of requests; precision rate is the ratio of number of filtered requests which are actually irrelevant items and the number of requests filtered by our research. A proposed method to achieve precision rate is finding a log the requests of which

Have been relevantly or irrelevantly labeled. On one side, as far as our existing knowledge and information express, presently there is no log file already labeled publically, and it’s due to the fact that URLs might contain sensitive and private information [9]; On the other side, manual labelling of each request, specifically for a great log file is quite time taking, and the result wouldn’t be that much valid. As stated above all requests filtered by standard filter and TimeFilter are really irrelevant items. Merely Prefix Filter could filter out several relevant requests and their URLs shall not be accessible overnight.

Algorithm number 1- pseudo code for selection of prefixes as rule.

```
Require: DayStandardLog DSL, threshold k;
Ensure: Prefix Set PS;
PS = NULL;
for each item $x \in \boldsymbol{DSL}$ and $x$ occurs at night do
    if x cannot be traced then
        if $prefix(x)$ is already in the hash table then
            $node[x].prefix@night++$;
// $prefix(x)$ is the prefix of $x$'s URL, $node[x]$ is the structure node that contains $prefix(x)$ in the hash table
        else
            initialize a new structure node containing $prefix(x)$, and add it into the hash table;
        end if
    end if
end for
for each item $x \in \boldsymbol{DSL}$ and $x$ occures in the daytime do
    if $prefix(x)$ is already in the hash table then
        $node[x].prefix@day++$;
    end if
end for
for each structure node in the hash table do
    if $\frac{node.prefix@night}{node.prefix@night+node@day} > k$ then
        add the prefix in node into the prefix set PS ;
    end if
end for
```

Table 1. statistical information of different logs

logs	File size (gigabytes)	Number of prefixes (10^6)	Number of requests (10^6)
Main log	14.28	17.59	41.15
Standard log	10.3	15.96	24.86
daily log	13.67	16.43	23.85
Filter logs	7.93	8.23	19.29

In this article, all requests filtered by Prefix Filter and those their URLs are not accessible overnight, are regarded as filtered items with relevant items, and a number of them are used for calculating precision rate. Nevertheless, in our opinion this value is much bigger than the real value of the precision rate; because most filtered requests, according to our analysis have been done by Prefix Filter of irrelevant items. Thus, real world traffic trace obtained from the borderline router of commercial intranet is used. Trace includes 1 month traffic in the time range of October 20, 2018 to November 20, 2018. In this tracing generally there are 17.59 million different URLs that have been accessed for 41.15 million times by 106 IP addresses, which forms a commercial proxy log of 14.28 gigabytes. Table 1 shows the statistical information over the main log, standard log, daily log and filter log which includes the file size, number of prefixes and their corresponding access times. Here, threshold k is regulated on 0.9 and time span has been specified in between 8 to 10 p.m as night. In table 1, it's shown that EPLogCleaner can filter out $1 - 9.29\ /41.15 = 77.43\%$ requests and with a precision rate of 95.5%. That's to say our EPLogCleaner is proper for improving the quality of commercial proxy logs; then, the impact of threshold k and night duration of N over filtering rate and precision rate are evaluated. In general night duration is defined as time/ hr that night covers. For the ease of comparison, the late night is limited to 8 a.m. thus, if the very night is set on 10. p.m, in that case, time duration representing as parameter N would be 10. Due to the fact that relevant items could be filtered only in the third stage of Prefix Filter, where k and N stand, the corresponding values will be compared in the daily log parameters, at the time of calculation of filtering rate and precision rate. Figures 3 and 4 demonstrate the oscillation of filtering rate and minimum of precision rate for different k tresholds on our experimental data. All values overnight are obtained as $N = 10$

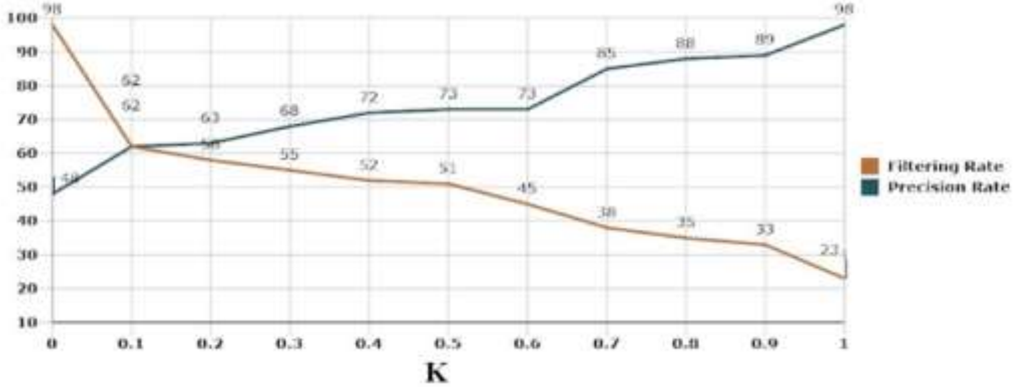
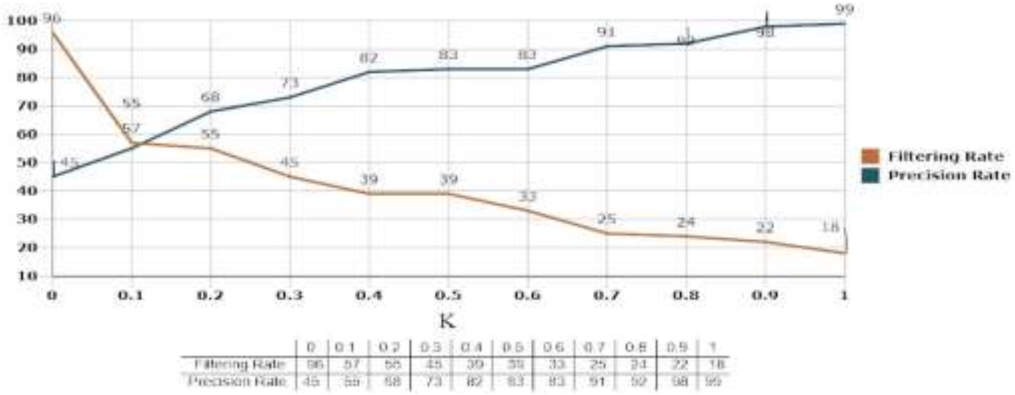


Figure 3. filtering rate and precision rate



	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Filtering Rate	96	57	55	45	39	39	33	25	24	22	18
Precision Rate	45	55	68	73	82	83	83	91	92	98	99

Figure 4. filtering rate and precision rate For URL requests

As shown in the figures, the decrease of k threshold limit, would decrease precision rate, whereas would increase the filtering rate; therefore, a Trade-off shall be made between filtering rate and precision. It's clear that the number of items can decrease over 30%, while the minimum of precision rate is always above 90%, when k is set equaling to 0.9. As a result 0.9 is selected as k value in the next experiment. The change in filtering rate, and the minimum of precision rate for URLs, and corresponding requests over different nights ($N = \{1,2,3,4,5,6,7,8,9,10\}$) are seen in the threshold of $k = 0.9$, and the result is shown in figures 5 and 6. As demonstrated in the figures, the filtering rate increases whereas with oscillation of night duration for URLs and corresponding requests, precision rate decreases in general except for in abnormal points where night duration is set as 5. Because there are a few number of unique URLs with filtered prefixes, which only between times of 2 to 3 can be accessed. Thus, the number of filtered URLs decreases when night time duration changes from 5 to 6, and eventually there is a slight decrease on filtering rate. As shown in figures 5 and 6, it's obvious that when night duration exceeds 8 and better than the others, there is a slight change in filtering rate and minimum of precision rate. The main reason is that the number of automatic requests over day long is fix, and the filtering rate, after a specific time on a specific value would be fixed.

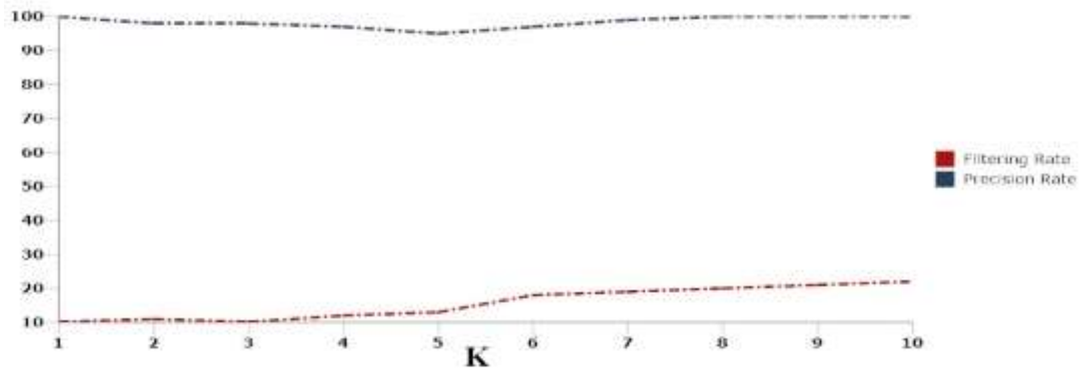


Figure 5. filtering rate and precision rate for URLs as a function of $N(K = 0.9)$

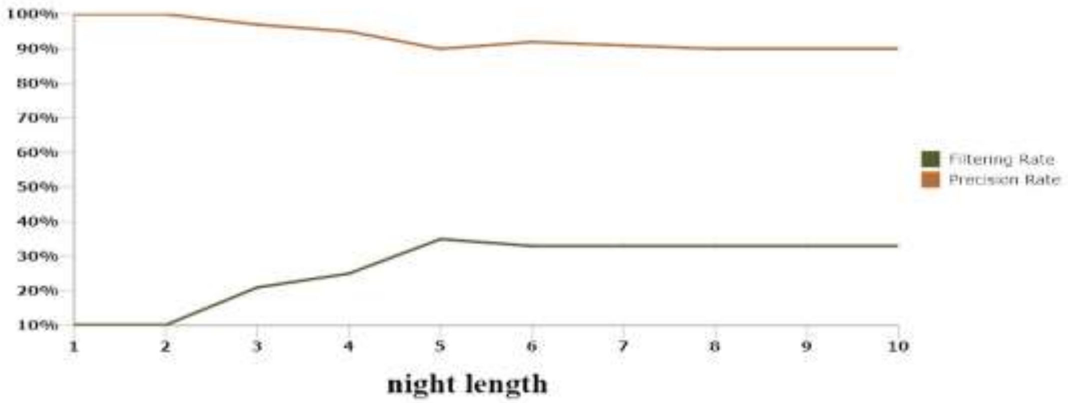


Figure 6. filtering rate and precision rate for URL requests as a function of $N(K = 0.9)$

Conclusion

Results of the experiments we have done indicate that the proposed method of EPLogCleaner can filter out over 30% of URL requests that cannot be filtered by the traditional methods of proxy logs data clearing. Though all filtered data are not necessarily valuable and relevant. In this regard some active links might add time stage to their URLs; therefore their prefixes cannot be added directly to our prefix library simply by a threshold which cause some of the irrelevant and useless data to remain in the final result; moreover, designing threshold and estimation method of precision rate is quite simple. In the future articles, we will analyze the time stage information of URL and also attaining features as for reaching a higher filtering rate. We also will work on the improvement of designing threshold and evaluation method of precision rate for the purpose of making the experiment results more accurate and more reliable.

REFERENCES

1. N. Tyagi, A. Solanki, S. Tyagi, An Algorithmic Approach to Data Preprocessing in Web Usage Mining, International Journal of Information Technology and Knowledge Management 2 (2) (2010) 279–283.
2. D. Tanasa, B. Trousse, Advanced Data Preprocessing for Intersites Web Usage Mining, IEEE Intelligent Systems 19 (2) (2004) 59–65.
3. G. Castellano, A. Fanelli, M. Torsello, LODAP: A Log Data Preprocessor for Mining Web Browsing Patterns, in: Proceedings of the 6th Conference on 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, 2007, pp. 12–17.
4. B. Singh, H. Singh, Web Data Mining Research: A Survey, in: Proceedings of the 2010 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), IEEE, 2010, pp. 1–10.
5. T. Hussain, S. Asghar, N. Masood, Web Usage Mining: A Survey on preprocessing of Web Log File, in: Proceedings of the 2010 International Conference on Information and Emerging Technologies (ICIET), IEEE, 2010, pp. 1–6.
6. Z. Pabarskaite, Implementing Advanced Cleaning and End-User Interpretability Technologies in Web Log Mining, in: Proceedings of the 24th International Conference on Information Technology Interfaces (ITI), IEEE, 2002, pp. 109–113.
7. Y. Zhang, L. Dai, Z. Zhou, A New Perspective of Web Usage Mining: Using Enterprise Proxy Log, in: Proceedings of the 2010 International Conference on Web Information Systems and Mining (WISM), Vol. 1, IEEE, 2010, pp. 38–42.
8. M. L. Berners-Lee, T., M. McCahill, Uniform Resource Locators (URL), RFC 1738.
9. K. Suneetha, R. Krishnamoorthi, Identifying User Behavior by Analyzing Web Server Access Log File, International Journal of Computer Science and Network Security 9 (4) (2009) 327–332.