Web Usage Data Cleaning

A Rule-Based Approach for Weblog Data Cleaning

Amine Ganibardi(✉) and Chérif Arab Ali

University of Vincennes in Saint-Denis,
2 Rue de la Liberté, 93526 Saint-Denis, France
{aganibardi,aa}@ai.univ-paris8.fr

Abstract. This paper addresses the issue of Weblog Data cleaning within the scope of Web Usage Mining. Weblog data are information on end-user clicks and underlying user-agent hits recorded by webservers. Since Web Usage Mining is interested in end-user behavior, user-agent hits are referred to as noise to be cleaned before mining. The most referenced and implemented cleaning methods are the conventional and advanced cleaning. They are content-centric filtering heuristics, based on the requested resource attribute of the weblog database. These cleaning methods are limited in terms of relevancy, workability and cost constraints, within the context of dynamic and responsive web. In order to deal with dynamic and responsive web constraints, this contribution introduces a rule-based cleaning method focused on the logging structure rules. The rule-based cleaning method experimentation demonstrates significant advantages compared to the content-centric methods.

Keywords: Web Usage Mining · Web usage data preprocessing
Weblog data cleaning

1 Introduction

Web Usage Mining is the application of data mining techniques to usage data, termed weblog data, for different purposes, e.g., system enhancement and adaptation, personalization, recommendation and advertisement [1, 2]. Depending on the analysis purpose and context, the weblog data are completed with other data [3] and are mined using different mining techniques, e.g., statistics, classification, clustering, association/sequential rules and dependency modelling. Beyond the conventional steps of a mining process, Web Usage Mining preprocessing step tackles with three critical tasks, i.e., data cleaning, data structuration into single/unique users and visits, and episodes identification [1, 4–6]. The cleaning task references the relevance of the whole mining process [7–9]. It is meant to process the raw weblog data (R.WLDB) and provide a noise-free weblog database of clicks (C.WLDB). A R.WLDB refers to data recorded by webservers within Access Logfiles (ALF). It reflects usage activity on Website servers, i.e., end-user clicks and underlying user-agent hits [1, 8]. Notice that end-user clicks and user-agent hits relate to webpages URI (Uniform Resource Identifier) and their underlying displayed view components, respectively. Since Web Usage Mining is interested in end-user behavior,

C. Ordonez and L. Bellatreche (Eds.): DaWaK 2018, LNCS 11031, pp. 193–203, 2018.
https://doi.org/10.1007/978-3-319-98539-8_15

user-agent hits are referred to as noise to be identified and removed before mining. Filtering hits from clicks is not trivial for two reasons, i.e., servers record requests interlaced in sequential order regardless of their source or type, websites resources may be set up as requestable interchangeably by end-users and user-agents [1, 8, 10].

On the basis of web usage mining reviewing papers from WEBKDD '99 up to 2017 [1–6, 11–13], the cleaning task can be mapped into three layers. The first is meant to identify clicks from hits. The second consists of selecting, from the identified clicks, those meaningful for the analysis purposes and context, e.g., the successful or unsuccessful requests, known single IP proxies and robots [3, 6, 7]. The third is performed downstream to data structuration and is rather a behavioral cleaning aimed at robot and outlier detection [3, 8, 14]. Thus, the core cleaning layer that tackles with generic noise is the identification of clicks from hits. In this regard, the most reported and implemented methods are the conventional and advanced cleaning. They are content-centric filtering heuristics based on the requested resource attribute of the R.WLDB. These cleaning methods are limited in terms of relevancy, workability and cost constraints, within the context of dynamic and responsive web content. In order to deal with dynamic and responsive web constraints, a rule-based cleaning method, focused on the logging structure rules, is introduced in the current paper. As the logging structure is insensitive to the logging content, the rule-based cleaning method experimentation demonstrates significant advantages compared to the conventional and advanced cleaning.

This paper is structured as follows: Sect. 2 depicts the cleaning methods and their limitations. Section 3 introduces the rule-based cleaning concept and method. Section 4 presents the experimental results analysis and evaluation.

2 Cleaning Methods Analysis

2.1 Data and Formalism

To better depict the cleaning method, the weblog data content, formats, attributes, and the underlying formalism are presented in the following. The most used ALF formats are the Common Log Format (CLF) and the Extended Common Log Format (ECLF) that contain information entries on usage activity [8]. The ALF entries content described in Table 1 represents the R.WLDB content. Given a R.WLDB of *(n)* requests *(REQ)* involving 7 entries instantiation recorded in an ALF as follows: REQ_n = *{127.0.0.1, −, frank, 10/Oct/2000:13:55:36 −0700, GET/apache.gif HTTP/1.0, 200, 2326,* http://apache.org/example.html*, Mozilla/4.08 (Win*; ...; Brow*)}*. Notice the following useful attributes and their abbreviation for the upcoming analysis: *[GET]* the request method (REQ.MET), *[apache.gif]* the requested resource (REQ.RES), *[apache]* the requested resource name (REQ.RES.NAM), *[gif]* the requested resource type (REQ.RES.TYP), *[apache.org]* the referring resource root (REF.RES.ROO), *[example.html]* the referring resource (REF.RES), *[example]* the referring resource name (REF.RES.NAM), *[html]* the referring resource type (REF.RES.TYP), *[Mozilla/4.08 (Win*; ...;Brow*)]* the user-agent (USE.AGE). Each attribute occurrence is indexed $\mathrm{req_n(att_m)}$. E.g., req.res = apache.org in REQ_n above is indexed $\mathrm{req_n(req.res)}$.

Table 1. Logfiles entries.

Entry	Description
IP address	IP address of the client (remote host) which made the request to the server
Client identity	RFC 1413 identity of the client determined by ident on the client's machine
Login	Login ID of the end-user requesting a resource (HTTP authentication)
Date and time	The time that the server finished processing the request
Request	The requested resource, request method, and http protocol version
Status code	Status code that the server sends back to the client (success, error, etc.)
Size	Indicates the size of the object returned to the client
Referrer	Indicates the resource that the client reports having been referred from
User-agent	Client browser information that it reports about itself (Browser and OS)

2.2 Related Methods

Conventional Cleaning. The conventional cleaning is based on the assumption that end-user clicks relate to html or equivalent web resource types as recommended by the World Wide Web Consortium (W3C) [1, 12]. All requests that relate to non-html resource types are assumed as user-agent requests and referred to as hits/noise. Thus, the cleaning attribute is the REQ.RES.TYP. In practice, undesired REQ.RES.TYP are set within a filtering knowledge database (FLT.KDB) that serves the cleaning heuristic. Thus, they are removed (FILTER-OUT) from the R.WLDB to come up with a cleaned weblog database (C.WLDB). The related cleaning process is given in Algorithm 1.

Algorithm 1. Conventional cleaning algorithm

```
01  INPUT DATA
02   R.WLDB
03   FLT.KDB
04  PROCESS
05   SCAN R.WLDB
06        IF req_n(req. res. typ) ∈ FLT. KDB FILTER-OUT req_n
07  OUTPUT DATA
08   C.WLDB
```

Advanced Cleaning. The advanced cleaning is based on prior knowledge on website URIs or a real-time extraction of those embedding clickable resources [10, 12]. The purpose is to set up a validation knowledge database (VLD.KDB) that contains all valid requestable resources imbedded in the website URIs and intended for end-user clicks. All requests that contain resources belonging to the VLD.KDB are referred to as clicks and are filtered into (FILTER-IN) the C.WLDB. Thus, the cleaning attribute is the REQ.RES. The related cleaning process is given in Algorithm 2.

Algorithm 2. Advanced cleaning algorithm

```
INPUT DATA
 R.WLDB
 VLD.KDB
PROCESS
 SCAN R.WLDB
      IF req_n(req. res) ∈ VLD. KDB FILTER-IN req_n
OUTPUT DATA
 C.WLDB
```

2.3 Limitation Analysis

In order to achieve high quality outputs, the conventional and advanced cleaning need an exhaustive prior knowledge, extra-weblog and extra-cost factors in addition to the cleaning process cost. The conventional cleaning need exhaustive prior knowledge and extra-weblog data related to resource types set up as requestable by end-users or those intended for user-agents to be referred to as noise. Since there are no mandatory standards in this regard, the construction and updating of this prior FLT.KDB represent an additional cost factor besides the cleaning process cost. In addition, the conventional cleaning becomes obsolete in the case of websites based on frames without filetype extensions used as a cleaning attribute. The advanced cleaning is based on a VLD.KDB that consists of the REQ.RES embedded within the website structure. Thus, it needs extra-weblog data related to the website structure. In case of dynamic web design including automatic personalization and structure adaption, where the website content and structure are shifting continuously, the advanced cleaning becomes overlapping for servers and even impossible in case of personalized content [3]. This is due to the fact that the construction and updating of the VLD.KDB need a continuous extraction of the website URIs intended for end-users. Such a process is very complex to perform, represents an extra-cost factor, and may miss new/cancelled content if not synchronized with the cleaning process in real-time. Finally, the fact that the two methods need an exhaustive prior knowledge related to the website content, structure, and its resources intended for end-users, to set up the filtering heuristic, represents a serious weakness. Obtaining such an exhaustive prior knowledge within the analytics perspective of server owners is not obvious because server owners do not have necessarily this knowledge, unlike the website owners [15].

3 Rule-Based Cleaning Approach

3.1 Targeted Contribution Concept and Method

The targeted contribution is meant to provide a cleaning method that meets the advanced cleaning advantages (noise-free) without any need for extra-weblog data, prior knowledge, or extra-cost factors beyond the cleaning process. Such a method

aims to be workable within the context of dynamic and responsive Web in both of the analytics perspectives, i.e., server and website owners. This cleaning method filters the R.WLDB on the basis of the logging structure rules. Since the logging structure depends only on the logging format, the proposed method is insensitive to the repercussions of the dynamic web in terms of websites structure and content. Thus, the rule-based cleaning targets to overcome relevancy, workability and cots constraints of the advanced and conventional cleaning.

The method consists of three main steps, i.e., (1) the identification of logging structure features related to clicks request out of hits, (2) the abstraction of the features into cleaning rules, (3) the implementation of the rules within a cleaning algorithm to be experimented and evaluated. The rule-based cleaning is based on the rules related to the regular features of the clicks logging structure. Thus, a browsing simulation itemset drawn on a predefined relevant browsing items is performed through a dedicated software (Webserver Stress Tool 8.0.0.1010) on a mirrored website (Simple English Wikipedia) within a localhost server (Apache server) set to generate an ALF in the ECLF. The relevant browsing items concern the different manner of URIs browsing by an end-user, e.g., access by click on links from a search engine or a third-party website, typing the URI in the browser search bar, backward and bookmark navigation. The generated ALF is collected and the requests related to the browsed URIs (relevant item/end-user clicks) are highlighted within their underlying hits (irrelevant item/user-agent hits). The identified regular features in terms of the REQ.RES and REF.RES attributes of the clicks logging structure are abstracted to infer rules underlying to clicks out of hits. The inferred rules are combined into a filtering process that filters-in the relevant items into the C.WLDB. An implementation algorithm under R within Apache Spark API is set to perform a cleaning test on third-party websites ALF. The outcomes of the rule-based cleaning are compared to the conventional and advanced cleaning.

3.2 Logging Structure Features Identification (Step 1)

The attributes and functions of the identified regular features of the logging structure are exhibited in Table 2. Three generic features (a, b, c) are identified within the ALF structure: (a) Website access by URN (homepage/Domain Name/Uniform Resource Name) takes on an empty requested resource attribute, between the request and http protocol attributes. (b) Website browsing by URLs (content page) that is identified by the relationship between the requested resource/content page URL (Uniform Resource Locator) attribute and the referring resource attribute of its components. (c) Interfering noise with the regular features cited above that takes on responsive requests, namely, Style Queries (SQ) interlaced within the logged referring resource at the ALF referrer entry.

3.3 Cleaning Rule (Step 2)

The identified regular features can be formalized in three rules, i.e., URN access/clicks, URL access/clicks, and regular interfering noise/hits.

Table 2. Regular logging features.

Request	Requested Resource [name & type]	Referring Resource [name & type]	Activity
(a) HOMEPAGE ACCESS *f* (REQ.RES n)			
R n	Empty (URN)	Internal V External Referer/Page	Click
R n+1	Media 1	Home Page/URN	Hit
…	…	Home Page/URN	…
…	Media m	Home Page/URN	Hit
(b) CONTENT PAGE ACCESS *f* (REQ.RES n & REQ.RES n+1)			
…	Content Page (URL)	Internal V External Referer/Page	Click
…	Media 1	Content Page/URL	Hit
…	…	Content Page/URL	…
…	Media m	Content Page/URL	Hit
(c) INTERFERING NOISE *f* (REF.RES.TYP n) [style query type]			
…	Homepage/Content Page (URI)	Internal V External Referer/Page	Click
…	Media name	*Media style query*	Hit
…	…	Content Page/URL	Hit
R n+x	Media m	…	Hit

$$\text{URN ACCESS} \Rightarrow \text{req}_n((\text{req.met } \&req.pro) \neq \varnothing \,\&\, req.res = \varnothing) \tag{1}$$

$$\text{URL ACCESS} \Rightarrow \text{req}_n(\text{req.res}) = \text{req}_{n+1}(\text{ref.res}) \tag{2}$$

$$\text{INTERFERING NOISE} \Rightarrow \text{req}_n(\text{ref.res.typ}) \in \text{SQ} \tag{3}$$

The automatic inference and fill-in of the website DN within the empty attribute (req.res) bring the URN access rule to the URL access rule. The automatic inference of the DN given in (Eq. 4) is argued in the following. The referrer entry of an ALF takes on internal and external referrers (URIs). One click is referred by one external referrer and as many internal referrers as the number of the pointed page components. Since the internal referring resources (URIs) consist of the same root (Website DN), and the external referring ones relate to different roots, the most frequent root within the referrer entry is the Website DN. Thus, the website DN is the most frequent referring resources root.

$$\text{DN} = \arg.\max_{\text{freq}}(\text{REQ } (\text{ref.roo})) \tag{4}$$

The automatic inference of the style/media queries noise (SQ) given in (Eq. 5) is argued in the following. Considering that style queries are the unique page component of SQ type that may appear within the referrer entry, it is the least frequent resource type within the intersection of the requested and referring resource types.

$$SQ = \arg.\min_{freq}(REQ.RES.TYP \cap REF.RES.TYP) \quad (5)$$

Equations 4 and 5 represent statistical properties of the logging structure within the ECLF of an ALF. They have been tested and validated within the 6 ALFs that served in the experimentation section (Sect. 4). Thus, we make the assumption that they are generalizable within the ECLF of ALFs. Thus, after the automatic inference of the DN and the SQ, we end up with two rules only, i.e., access/clicks and noise rules.

$$ACCESS \Rightarrow req_n(req.res) = req_{n+1}(ref.res) \quad (6)$$

$$NOISE \Rightarrow req_n(ref.res.typ) \in SQ \quad (7)$$

3.4 Ruel-Based Cleaning Algorithm (Step 3)

Algorithm 3 draws the rule-based cleaning. The rule-based algorithm begins by parsing the weblog data to, first, infer the DN and fit it in the empty attributes where the requests method and http protocol are not missing. Then, it infers the SQ type that may appear in the referring entry. Once the SQ type is inferred, the requests whose referrers contain the SQ resource type are removed. Thus, the access rule can be applied to filter-in requests referred to as clicks into the C.WLDB.

Algorithm 3. Rule-based algorithm

```
INPUT DATA
 R.WLDB
PROCESS
 [req_n(req.met &req.pro = Ø)] ← arg.max_freq (REQ (ref.roo))]
 SQ=arg.min_freq (REQ.RES.TYP∩ REF.RES.TYP)
 WHILE req_n(ref.res.typ = SQ) FILTER-OUT req_n WEND
       IF req_n(req.res) = req_n+1(ref.res) FILTER-IN req_n
OUTPUT DATA
 C.WLDB
```

4 Experimentation and Results Discussion

4.1 Experimental Data and Validation Reference

The rule-based cleaning algorithm as well as the conventional and advanced cleaning ones have been tested on a representative sample of 6 ALFs of third-party websites. The description of the experimental data is given in Table 3. This panel aims to be representative of the different practices in terms of web design, web content, and clicks/hits ratio. The involved ALFs relate to: a public administration (AL1.GOV), commercial websites (AL2.COM, AL3.COM, AL4.COM, AL5.COM), and our Laboratory website (AL6. LAB). The size of the ALFs is given in requests number.

They have been processed with the involved cleaning methods through their related Algorithms 1, 2 and 3, implemented under R within Apache Spark API.

Table 3. Experimental data description.

ALF	Website DN	ALF Source	Size	Clicks	Hits	Ratio
AL1.GOV	www.khanyounis.mun.ps	https://www.mosa.gov.ps/khanyounis.mun.ps/log/access.log	26 168	2 564	23 604	1/9
AL2.COM	almhuette-raith.at	http://www.almhuette-raith.at/apache-log/access.log	31 433	13 108	18 325	1/2
AL3.COM	www.facades.fr	http://igm.univ-mlv.fr/~cherrier/download/L1/access.log	5 945	987	4 958	1/5
AL4.COM	www.boring-log.com	https://www.scisoftware.com/boring-log.com/logs/apache-access.log	17 676	3 164	14 512	1/4
AL5.COM	www.megapeloteros.com	http://salablanda.com.ar/megapeloteros.com.access.log	680	48	632	1/13
AL6.LAB	www.ai.univ-paris8.fr/	Laboratory	145 227	25 391	119 836	1/6

The methods outputs are evaluated under the terms of a confusion matrix where: Size (SIZ) refers to the processed ALF size given in requests number; Positive (P) and Negative (N), respectively, indicate the number of clicks and hits the ALF contains. True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) indicate the valid and invalid requests, respectively, misreferred to as clicks or hits by the cleaning algorithm; False Positive Rate ($FPR = \frac{FP}{FP+TN}$) measures the noise rate; False Negative Rate ($FNR = \frac{FN}{FN+TP}$) measures the miss rate; Accuracy ($ACC = \frac{TP+TN}{P+N}$) measures the cleaning relevance. The validation reference (P and N number/labels) is composed of valid requestable resources by an end-user within the websites of the involved ALFs. This validation reference is set by a website checker software (Xenu's Link Sleuth 1.3.8) that reports, among others, the URIs embedding clickable resources intended for end-users.

4.2 Results Analysis

The cleaning results are depicted in Table 4. The analysis of the noise rate (FPR) of the CON.CLE algorithm shows that 53% consists of frames without filetype extensions in addition to unexpected responsive filetype extensions. 47% of the miss rate relate to different formats of clickable media content (non-html resources) that have been requested by end-users. The results in terms of FPR of the CON.CLE confirm the statement of its inappropriateness for frame-based web design. The results in term of FNR demonstrate the exhaustivity challenge in terms of prior FLT.KDB. Overall, the

Table 4. Cleaning results.

ALF	Size	P	N	TP	FP	TN	FN	FPR	FNR	ACC
Advanced Cleaning (ADV.CLE)										
AL1.GOV	26 168	2 564	23 604	2 564	0	23 604	0	0%	0%	100%
AL2.COM	31 433	13 108	18 325	13 108	0	18 325	0	0%	0%	100%
AL3.COM	5 945	987	4 958	987	0	4 958	0	0%	0%	100%
AL4.COM	17 676	3 164	14 512	3 164	0	14 512	0	0%	0%	100%
AL5.COM	680	48	632	48	0	632	0	0%	0%	100%
AL6.LAB	145 227	25 391	119 836	25 391	0	119 836	0	0%	0%	100%
SUM/**AVG**	*227 129*	*45 262*	*181 867*	*45 262*	*0*	*181 867*	*0*	**0%**	**0%**	**100%**
Rule-based Cleaning (R.CLE)										
AL1.GOV	26 168	2 564	23 604	2 564	0	23 604	0	0%	0%	100%
AL2.COM	31 433	13 108	18 325	13 108	0	18 325	0	0%	0%	100%
AL3.COM	5 945	987	4 958	987	0	4 958	0	0%	0%	100%
AL4.COM	17 676	3 164	14 512	3 164	0	14 512	0	0%	0%	100%
AL5.COM	680	48	632	48	0	632	0	0%	0%	100%
AL6.LAB	145 227	25 391	119 836	25 391	0	119 836	0	0%	0%	100%
SUM/**AVG**	*227 129*	*45 262*	*181 867*	*45 262*	*0*	*181 867*	*0*	**0%**	**0%**	**100%**
Conventional Cleaning (CON.CLE)										
AL1.GOV	26 168	2 564	23 604	2 252	312	22 947	657	1%	23%	96%
AL2.COM	31 433	13 108	18 325	12 725	383	6 474	11 851	6%	48%	61%
AL3.COM	5 945	987	4 958	795	192	4 852	106	4%	12%	95%
AL4.COM	17 676	3 164	14 512	3 040	124	11 715	2 797	1%	48%	83%
AL5.COM	680	48	632	43	5	627	5	1%	10%	99%
AL6.LAB	145 227	25 391	119 836	20 270	687	119 149	5121	1%	20%	96%
SUM/**AVG**	*227 129*	*45 262*	*181 867*	*39 125*	*1 703*	*165 764*	*20 537*	**2%**	**27%**	**88%**

CON.CLE is very noisy (ACC 88%). Both of the ADV.CLE and R.CLE tests give a noise-free (FPR) and a miss-free (FNR) rates. This is due to the fact that they are based on the same filtering attributes (REQ.RES) retrieved from two different sources. The R.CLE includes it in the access rule (Eq. 6) that is a function of the referring URIs within the ALF. However, the ADV.CLE retrieves it from the website URIs. Since the extracted website URIs contain all the possible requestable resources by click, the ADV.CLE provides noise-free outputs while the R.CLE reaches the same result in an optimized way, as it focuses only on the resource that has been requested by clicks identified by the access rule.

4.3 Results Evaluation

The CON.CLE is the most advantageous method in terms of relevancy balanced by workability and costs constraints. It provides high quality outputs with only one cost factor (the cleaning process). It does not need any extra-weblog data, making it insensitive to dynamic and responsive Web repercussions on the logging content. In addition, since it does not need any prior knowledge, it is workable for both website and server owner analytics perspectives. The ADV.CLE that gives also noise-free results is overlapping for

servers and proves to be unworkable in case of automatic and continuous shifting content and structure, specifically in case of personalized content. The CON.CLE is very noisy and proves to be obsolete in the case of responsive Web, based on frames. Both of the ADV.CLE and CON.CLE need exhaustive and prior knowledge on the involved website, making them limited to the website owner analytics perspective. Overall, the ADV.CLE and R.CLE can be combined where the early is optimized by the later, as they share a common filtering attribute from two different sources.

5 Conclusion

This contribution draws a critical analysis of weblog data cleaning. It depicts the related cleaning methods limitations in terms of relevancy, workability and cost constraints. The limitations of the analyzed methods are due to dynamic and responsive web repercussions on the logging content. These cleaning methods are content-centric and prove to be inappropriate to such a context, specifically within the analytics perspective of sever owners. Since the proposed method is focused on the logging structure, it demonstrates significant advantages compared to the content-centric cleaning methods in terms of relevancy balanced by workability and costs constraints. Finally, despite the limited size of the experimented ALF, the fact that the rule-based cleaning is based on the logging format, make it generalizable within the ECLF of ALFs, since the format does not depend on the logging size.

References

1. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.-N.: Web usage mining: Discovery and applications of usage patterns from web data. ACM SIGKDD Explor. Newsl. **1**(2), 12–23 (2000)
2. Srivastava, M., Garg, R., Mishra, P.K.: Preprocessing techniques in web usage mining: a survey. Int. J. Comput. Appl. **97**(18), 1–9 (2014)
3. Kohavi, R.: Mining e-Commerce data: the good, the bad, and the ugly. In: Cheung, D., Williams, G.J., Li, Q. (eds.) PAKDD 2001. LNCS (LNAI), vol. 2035, p. 2. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45357-1_2
4. Facca, F.M., Lanzi, P.L.: Mining interesting knowledge from weblogs: a survey. Data Knowl. Eng. **53**(3), 225–241 (2005)
5. Langhnoja, S., Barot, M., Mehta, D.: Pre-processing: procedure on web log file for web usage mining. Int. J. Emerg. Technol. **2**(12), 5 (2012)
6. Chitraa, V., Thanamani, D.A.S.: Web log data cleaning for enhancing mining process. Int. J. Commun. Comput. Technol. **01**(03), 7 (2012)
7. Srivastava, J., Desikan, P., Kumar, V.: Web mining: Accomplishments and future directions. In: National Science Foundation Workshop on Next Generation Data Mining (NGDM 2002), pp. 51–56 (2002)
8. Pabarskaite, Z., Raudys, A.: A process of knowledge discovery from web log data: systematization and critical review. J. Intell. Inf. Syst. **28**(1), 79–104 (2007)
9. Spiliopoulou, M., Mobasher, B., Berendt, B., Nakagawa, M.: A framework for the evaluation of session reconstruction heuristics in web-usage analysis. Informs J. Comput. **15**(2), 171–190 (2003)

10. Pabarskaite, Z.: Implementing advanced cleaning and end-user interpretability technologies in web log mining. In: 2002 Proceedings of the 24th International Conference on Information Technology Interfaces, ITI 2002, pp. 109–113 (2002)
11. Dhandi, M., Chakrawarti, R.K.: A comprehensive study of web usage mining, pp. 1–5 (2016)
12. Srinivas, A.V.: A survey on preprocessing of web-log data in web usage mining. Int. J. Modern Trends Sci. Technol. **03**(02), 35–41 (2017)
13. Zhang, Q., Segall, R.S.: Web mining: a survey of current research, techniques, and software. Int. J. Inf. Technol. Decis. Making **7**(04), 683–720 (2008)
14. Spiliopoulou, M.: Web usage mining for Web site evaluation. Commun. ACM **43**(8), 127–134 (2000)
15. Zahran, D.I., Al-Nuaim, H.A., Rutter, M.J., Benyon, D.: A comparative approach to web evaluation and website evaluation methods. Int. J. Pub. Inf. Syst. **10**(1), 21–39 (2014)