Noname manuscript No.
(will be inserted by the editor)

Content-Aware Web Robot Detection

Athanasios Lagopoulos · Grigorios Tsoumakas

Received: date / Accepted: date

Abstract Web crawlers account for more than a third of the total web traffic and they are threatening the security, privacy and veracity of web applications and their users. Businesses in finance, ticketing and publishing, as well as websites with rich and unique content, are the ones mostly affected by their actions. To deal with this problem, we present a novel web robot detection approach that takes advantage of the content of a website based on the assumption that human web users are interested in specific topics, while web robots crawl the web randomly. Our approach extends the typical user session representation of log-based features with a novel set of features that capture the semantics of the content of the requested resources. In addition, we contribute a new real-world dataset, which we make publicly available, towards alleviating the scarcity of open data in this field. Empirical results on this dataset validate our assumption and show that our approach outranks state-of-the-art methods for web robot detection.

Keywords Web Robot · Crawler · Semantics · Supervised Learning · Latent Dirichlet Allocation

1 Introduction

Web (ro)bots constantly request resources from web servers across the Internet, without human intervention, indexing and scraping content with an aim to

This research is co-financed by Greece and the European Union (European Social Fund-ESF) through the Operational Programme Human Resources Development, Education and Lifelong Learning in the context of the project Strengthening Human Resources Research Potential via Doctorate Research (MIS-5000432), implemented by the State Scholarships Foundation (IKY).

A. Lagopoulos · G.Tsoumakas
Aristotle University of Thessaloniki, Thessaloniki, Greece
E-mail: lathanag@csd.auth.gr, greg@csd.auth.gr

make information reachable and available on demand. Recent industry reports show that 37.9% (42.2%) of all the web traffic in 2018 (2017) was generated by web robots, affecting every industry all over the world (Networks, 2019; Dots, 2018).

Bots may access web applications for beneficial reasons, such as indexing and health monitoring (Doran et al., 2013). However, around half of the bot traffic is considered to be malicious, threatening the security and privacy of a web application and its users. With an ultimate goal to monetize the information requested, they perform actions such as price and content scraping, account take over and creation, credit card fraud and denial of service attacks (Foundation, 2018). Businesses in finance, ticketing and education sectors are the ones most affected by these actions and need to deal not only with security issues but also with the unfair competition deriving from such fraudulent practices. Furthermore, another common threat that web applications need to deflect is analytics skewing, which is caused by otherwise benign bots. Websites with unique and rich content, like data repositories, marketplaces and digital publishing portals, see their reports and metrics altered, rendering their validity questionable (Greene, 2016). In addition to this, social bots may unethically influence social dialogue and contribute further to the spread of fake news (Ferrara et al., 2016). Therefore, the detection of web robots and the filtering of their activities are important tasks in the fight for a secure and trustworthy web.

This article introduces a novel web robot detection approach that takes into account the content of a website. The key assumption of the proposed approach is that humans are typically interested in specific topics, subjects or domains, while robots typically crawl all the available resources uniformly (Rude and Doran, 2015; Brown and Doran, 2018) and regardless of their content. Based on this assumption, our main contribution is a novel representation for web sessions that quantifies the semantic variance of the web content requested within a session. Correspondingly, our main research question is whether such a content-aware representation can improve over state-of-the-art approaches that neglect content.

This work is an extension of a previously published conference paper (Lagopoulos et al., 2018). Specifically, in this work we present two additional content-aware representations that capture the semantics of the content. Furthermore, we contribute a dataset consisting of log file entries obtained from our university's library search engine in two forms: (i) the raw log files as obtained from the server, and (ii) their processed form as a labeled dataset of log entries grouped into sessions along with their extracted features. We make this dataset publicly available, the first one in this domain, in order to provide a common ground for testing web robot detection methods, as well as other methods that analyze server logs. Finally, we introduce a simple but very effective baseline method for web robot detection based on supervised learning with features proposed in previous studies.

The rest of this paper is structured as follows: After a discussion of the related work in Section 2, we introduce our approach to extracting content-

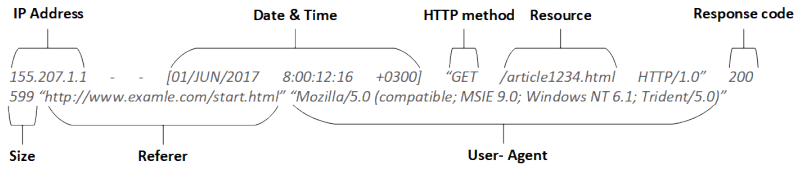

Fig. 1 Example of an entry in a web server access log file.

based features from web sessions in Section 3. In Section 4, we describe our real-world case study and the corresponding dataset, while in Section 5 we discuss the results of our study and compare our method with the state of the art. Finally, in Section 6, we conclude this work and draw future directions.

2 Related Work

2.1 Session Identification & Features

An important first step in web robot detection approaches is *session identification*, which is concerned with chunking the clickstream into sessions. The standard procedure groups together requests that share the same IP address and user-agent string, which are then broken into sessions by applying a timeout threshold (Stassopoulou and Dikaiakos, 2009). Various timeout thresholds have been investigated in the past, such as 10 minutes (AlNoamany et al., 2013), 30 minutes (Tan and Kumar, 2004) and using dynamically adaptive thresholds ranging from 30 minutes to 60 minutes (Stassopoulou and Dikaiakos, 2007). Kang et al. (2010) used the unique user id that is stored in cookies to identify users and, consequently, doesn't rely solely on server log files like most of the related approaches.

The session identification step is followed by the feature extraction step. For each identified session, a number of features are extracted based on the variety of information found in the entries of web server access logs. Such information includes the *IP address* of the host that made the request to the server, the *date and time* that the request was received, the *resource* requested, the *HTTP method* used, the HTTP response code sent back to the client, the *size* of the returned object, the *Referrer HTTP request header*, which is the page that links to the resource requested and the *user-agent String* that identifies the client's browser. Features extracted solely from web server access logs can be referred as log-based features. Figure 1 shows a typical sample entry of a server access log.

Most modern web applications are built with technology, such as JavaScript, that enables them to track and gain additional knowledge about their users, beyond that found in web server access logs. For example, HTML geolocation enables the identification of the country a request originates from, demographic details can be obtained from users with an account and advanced interfaces

Table 1 Universal features found in the literature.

ID	Feature Name	Description
1	Duration	Total time elapsed between the first and the last request.
2	Total Requests	The total number of requests.
3	Average Time	Average time between two consecutive requests.
4	SD Time	Standard deviation of the time between two requests.
5	% Repeated	Percentage of repeated requests.
6	Total Pages	The total number of pages requested.
7	%GET	Percentage of requests with HTTP method GET.
8	%POST	Percentage of requests with HTTP method POST.
9	%HEAD	Percentage of requests with HTTP method HEAD.
10	%OTHER	Percentage of requests with any other HTTP method.
11	%Night	Percentage of requests made between 12am and 7am.
12	%Unassigned	Percentage of requests with unassigned referer ("-").
13	%Images	Percentage of image file requests.
14	Width	Width of the traversal in the URL space.
15	Depth	Depth of the traversal in the URL space.
16	%2XX	Percentage of requests with a successful status code (2xx).
17	%3XX	Percentage of requests with a redirection status code (3xx).
18	%4XX	Percentage of requests with a client error status code (4xx).
19	%5XX	Percentage of requests with a server error status code (5xx).
20	%Consecutive	Percentage of consecutive sequential HTTP requests.
21	SD Depth	Standard deviation of page depth across all requests.
22	Image Ratio	Requests ratio of HTML pages to image files.
23	Data	Total bytes transferred between the server and the client.
24	PPI Score	Average popularity index of each page found in a session.
25	SF Referer	Switching Factor on unassigned referer field.
26	SF File Type	Switching Factor of file type.
27	Loop Penalty	Penalty for each backward and forward navigation or loop.
28	Max Barrage	Maximum number of embedded resources in a web page.

can tell if the request originates from a web service. However, such information is usually application dependent and prone to legislation changes.

Tables 1 and 2 present the features that have been proposed over the years on web robot detection taking into consideration the capability of extracting them from standard web server logs using the standard session identification procedure. Table 1 describes the features that can be used universally on any type of web application and are easily extracted from server logs. Table 2 describes the application dependent features proposed in the past, which are not suitable for general purpose studies and are therefore not adopted in this study. To the best of our knowledge, we are the first to consider content-based features in a machine learning approach for web robot detection.

2.2 Detection Approaches

Several data mining techniques have been proposed in the past based on a variety of learning algorithms and input features. Tan and Kumar (2004) used decision trees to train a model using 25 different features that were extracted from each user session. The feature vector included percentages of the different resource types (images, multimedia, HTML, etc.), time statistics (average time, total time, etc.), request types (GET, POST, HEAD, etc.) and other (IP, user-agent, etc.). Bomhardt et al. (2005) used neural networks and included features like the total number of bytes and the percentage of response codes (200, 2xx, 404, etc.). Stassopoulou and Dikaiakos (2009) used a heuristic semi-automatic method to label the training data and introduced a Bayesian approach to classify the sessions. A Bayesian approach was also followed by Suchacka and Sobkow (2015) along with two different criteria for labeling sessions as robots. Stevanovic et al. (2012) experimented with a variety of machine learning algorithms (C4.5, RIPPER, k nearest neighbors, Naive Bayes, Bayesian Network, SVM and Neural Network) and introduced two novel features considering the page depth of a session's requests and the sequentiality of HTTP requests. Finally, other approaches used request patterns to identify robot sessions. Kwon et al. (2012a) used a simple but effective technique by creating a pattern table from request types and Doran and Gokhale (2016) introduced a novel approach that can be used for real-time detection of web robots based on a first-order discrete time Markov chain model.

In contrast with the above supervised approaches, Stevanovic et al. (2013) and Ansari et al. (2017) used unsupervised neural networks, modified adaptive resonance theory 2 (ART2) and self-organizing map (SOM) to detect

Table 2 Application dependent features found in the literature.

ID	Feature Name	Description
29	Trap File	Number of trap file requests (Zabihi et al., 2014). Trap files may differ or not exist depending on the application.
30	Resolve	Time taken to serve a request (Lee et al., 2009). Computing the time taken to serve a requests requires monitoring from the client side.
31	SF Bytes	Switching factor on number of bytes transferred from clients to the server (Kwon et al., 2012b). Definition is unclear.
32	%File	Percentages of different file types requested, such as .exe, .pdf and other (Tan and Kumar, 2004; Stevanovic et al., 2013). Some web applications do not contain these kind of files at all, rendering their usefulness questionable.
33	Multi IPs	Indication of multiple IPs (Tan and Kumar, 2004). This feature cannot be used together with the standard session identification procedure.
34	Multi User-Agents	Indication of multiple user-agent strings (Tan and Kumar, 2004). This feature cannot be used together with the standard session identification procedure.

humans and robots and further analyze the behavior of malicious and non-malicious web robots, while Zabihi et al. (2014) used the DBSCAN clustering algorithm with just four different features. More recent approaches are based on fuzzy rough set theory and dynamically select the features used to describe web visitors (Zabihimayvan et al., 2017; Zabihimayvan and Doran, 2018; Hamidzadeh et al., 2018). Finally, Kang et al. (2010) proposed a semi-supervised approach to take advantage of the unlabelled data and a novel method that uses CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) to generate training data.

Table 3 links together previous approaches with the algorithms they used and the features extracted as presented in Tables 1 and 2.

3 Extracting Content-Aware Features from Sessions

The fundamental assumption of this work is that typically humans look for specific information on a particular topic or subject during a session, while web bots go through the content of a website without favoring specific pages or content. Building a web robot detection approach on top of this assumption requires measuring the semantic (in)coherence of the content visited during a

Table 3 Previous web robot detection approaches along with the algorithm/method used and their respective features as presented in Tables 1 & 2.

Publication	Algorithm or Method	Features' IDs
Tan and Kumar (2004)	C4.5 Decision Tree	1-15, 32-34
Bomhardt et al. (2005)	Logistic Regression, Decision Trees, Neural Networks	1-13, 16-19
Stassopoulou and Dikaiakos (2009)	Bayesian Network	1, 13, 18, 22, 32
Lee et al. (2009)	Characterization Metrics	6-10, 12, 18-19, 23-24, 30, 32
Kang et al. (2010)	Semi-Supervised Bayesian Network	2, 6, 33 and other
Kwon et al. (2012b)	C4.5 Decision Tree	25, 26, 31
Stevanovic et al. (2012)	Decision Trees, SVM, Bayesian Network, Multilayer Perceptron, K-NN	2, 9, 12, 18, 20-22, 32
Doran and Gokhale (2016)	Discrete-Time Markov Chains	Resource Types
Stevanovic et al. (2013)	Self Organizing Maps (SOM), Modified Adaptive Resonance Theory 2 (ART2)	2, 9, 12, 18, 20-24, 32
Zabihi et al. (2014)	DBSCAN	17, 27, 28, 29
Zabihimayvan et al. (2017)	Fuzzy Rough Sets + Markov Clustering algorithm	1-3, 6-15, 17-18, 20-29, 31-34
Hamidzadeh et al. (2018)	Fuzzy Rough Sets + Self Organizing Maps (SOM)	1-3, 6-15, 17-18, 20-29, 31-34

session. To achieve this, we start with topic modeling of the content of a website using latent Dirichlet allocation (LDA) (Blei et al., 2003). LDA describes each document or, in this case, each web resource, as a probability distribution over a user-defined number, k, of topics, where each topic is a probability distribution over words. With LDA, we can extract human-interpretable topics from a corpus, in contrast with other topic modeling algorithms such as Latent semantic analysis (LSA). The Biterm Topic Modelling (BTM) was also considered but since a web page may contain a great amount of text it was quickly discarded. In general, LDA was chosen for its modularity, interpretability, and ability to produce sentence/paragraph/document vectors out-of-the-box.

Consider a session, S, comprising n requests for web pages. Let p_{ij}, be the probability of topic j, $1 \leq j \leq k$, for the web page associated with request i, $1 \leq i \leq n$. Let also $\mathbf{p}_i$ be a vector containing the probability distribution over the k topics for the web page associated with request i. Figure 2 illustrates such vectors. A simple way to produce a vector of content-based features for a session is to sum or average the $\mathbf{p}_i$ vectors associated with that session. Each session is then represented as a vector of length k. We refer to these techniques as CBS (content-based sum) and CBA (content-based average) respectively. The feature vectors in CBS and CBA are defined as follows:

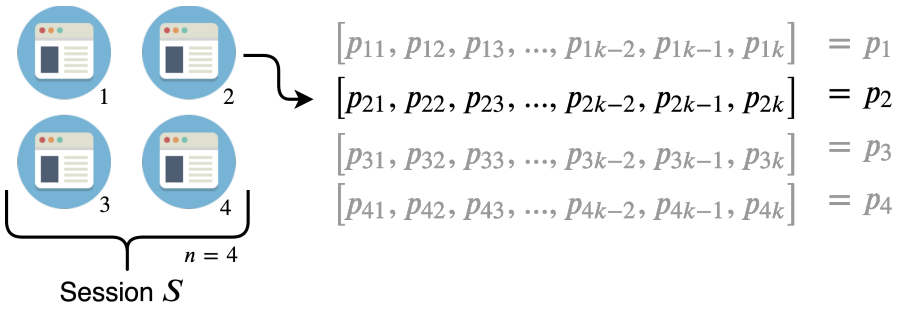

Fig. 2 A diagram depicting the content vectors of a web session S comprising $n = 4$ requests for web pages, where k is the number of LDA topics.

$$\text{CBS: } \mathbf{p}_{\text{sum}} = \left\langle \sum_{i=1}^{n} p_{i1}, \sum_{i=1}^{n} p_{i2}, \ldots, \sum_{i=1}^{n} p_{ik} \right\rangle \tag{1}$$

$$\text{CBA: } \mathbf{p}_{\text{avg}} = \frac{1}{n}\mathbf{p}_{\text{sum}} \tag{2}$$

Finally, we propose 5 new handcrafted features deriving from the vectors $\mathbf{p}$ of a session S that we believe can express the semantic (in)coherence of the content visited during a session. We refer to this method as CBF (content-based features) and the proposed features are:

– *Total Topics* (TT). The number of topics with non-zero probability.

$$\mathrm{TT} = |\{(i,j) : 1 \leq i \leq n, 1 \leq j \leq k, p_{ij} \neq 0\}|$$

The higher the total number of topics with non-zero probability in all requests of a session, the lower the semantic coherence of the session.

– *Unique Topics* (UT). The number of unique topics with non-zero probability.

$$\mathrm{UT} = |\{j : 1 \leq j \leq k, \sum_i p_{ij} \neq 0\}|$$

This feature measures the semantic inconsistency of a session too, but without counting the same topic twice.

– *Page Similarity* (PS). The ratio of unique topics with non-zero probability over all the topics with non-zero probability.

$$PS = \frac{UT}{TT}$$

This feature models the dissimilarity of the different pages visited during a session. The lower its value, the more semantically similar the requested resources.

– *Page Variance* (PV). The semantic variance of the pages of a session.

$$PV = \frac{\sum_i \sqrt{\sum_{j=1}^{k} (p_{ij} - \overline{\mathbf{p}}_j)^2}}{n},$$

where $\overline{\mathbf{p}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{p}_i$ is the mean of the $\mathbf{p}_i$ vectors that are associated with each request of the session. This feature computes the mean Euclidean distance of the topic distribution of the resource of each request with that of the mean topic distribution. The lower this distance, the higher the semantic similarity of the requested resources in the session.

– *Boolean Page Variance*. It is a boolean version of PV, where prior to its calculation we set all non-zero p_{ij} values to be equal to 1.

Our proposed method for web robot detection is a simple supervised learning approach that combines *log-based* features with the *content-based* features introduced above. Specifically, we use the log-based features 1-28 mentioned in Section 2.1 in tandem with the content-based features proposed in the current section, as extracted by the methods, namely CBS, CBA and CBF. The complete feature vector is created by concatenating the log-based features with the features extracted from one of the content-based methods. The size of the vector is $k + 28$ for the CBS and CBA methods and 33 $(5 + 28)$ for the CBF method. The resulting feature vector can then be given as input to a learning algorithm.

4 Real World Case Study

Our real world case study is concerned with web robot detection in the search engine of the library of the Aristotle University of Thessaloniki[1]. First, we present and give details on the dataset we obtained and use in all our experiments. Then, we discuss our preprocessing steps and session identification approach and finally, we introduce our labeling procedure.

4.1 Dataset

Our data come from the search engine of the library and information center of the Aristotle University of Thessaloniki in Greece. The search engine enables users to check the availability of books and other written works, and search for digitized material and scientific publications. The server logs obtained span an entire month, from March 1st to March 31, 2018, and consist of 4,091,155 requests with an average of 131,973 requests per day and a standard deviation of 36,996.7 requests. In total, there are requests from 27,061 unique IP addresses and 3,441 unique user-agent strings. Figure 3 shows the request distribution over time. We notice some repeating patterns, where five long spikes are followed by two shorter ones. As indicated by the graph, the long spikes belong to weekdays while the short ones belong to weekends. Figure 4 gives us more details on the distribution of requests by presenting a heatmap of the requests by day of week and hour of day, where the size of the circle indicates the average number of requests on a specific day (Monday, Tuesday, etc.) and time (00:00-23:59). We can see that the peak time is between 9-12 am during the weekdays while during the night the number of requests is minimal. This is not surprising since $\approx 87\%$ of the requests are coming from Greece, while in total there are requests from 92 different countries[2].

Besides the log files, we also use the text content found in the requested web pages. We scraped all the texts with sufficient length or semantic value found on a web page. Such text is the title and description of a library record visited by the user, the similar items proposed by the search engine and any search query submitted by the user. In total, there are 575,071 unique text records and each of them is associated with a request. The remaining requests consider other types of files (non HTML) or pages with no semantic value such as the home page, login pages, contact page, etc.

The dataset is publicly available in Zenodo [3]. Server logs were anonymized by masking the last 6 digits of the IP address and the last part of the URLs (after the last /). This will not prevent others from using the dataset since they can still identify sessions and extract all the features described in the study.

[1] `http://search.lib.auth.gr/`

[2] The country codes are obtained from the IP address using the Geolite2 database (`https://dev.maxmind.com/geoip/geoip2/geolite2/`)

[3] `https://zenodo.org/record/3477932`

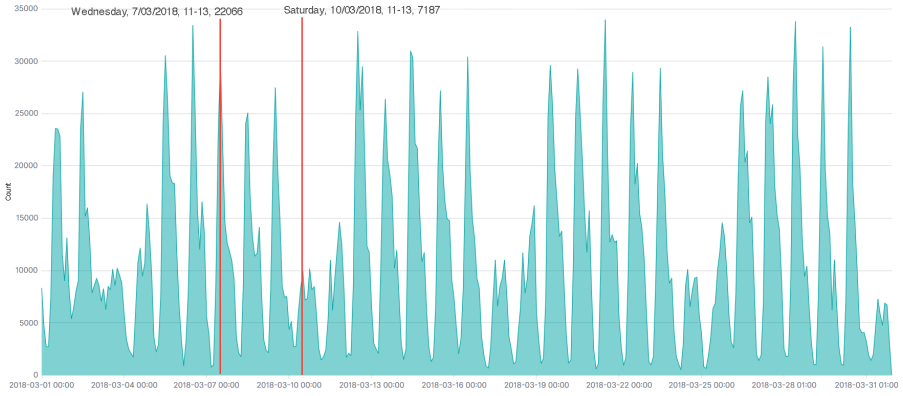

Fig. 3 Request distribution over time in 2 hour intervals.

4.2 Session Identification

As a first step we apply a session identification process that first groups together requests with the same IP address and user-agent string and then applies a timeout threshold to break the groups into sessions. The timeout threshold was set to 30 minutes as the literature suggests.This process identified 74,970 sessions. Furthermore, we ignore sessions with a blank user-agent since they were found to be sessions with timed-out requests (408 response code). This led to 67,352 sessions.

The sessions created are quite inconsistent both in terms of number of requests and duration.They have an average (median) of 54.53 (16) requests, while there are sessions with more than 5,000 requests. Their average (median) duration is 672.9 (27) seconds and the average time between two consecutive requests is 34.65 seconds.

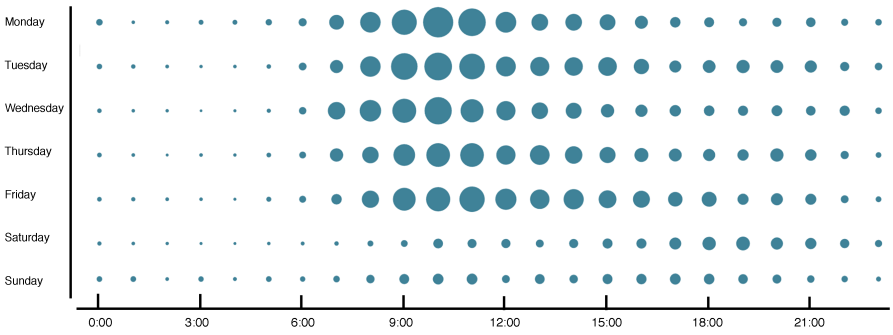

Fig. 4 A heatmap showing the number of requests by day of week and hour of day.

We used the Elastic Stack[4] (Elasticsearch and Logstash) to parse and identify the fields of the server logs.

4.3 Session Labeling

The session labeling procedure is a very difficult task that is seldom perfect. In order to achieve a high quality of labeling, we followed a 4-stage labeling procedure. During the first stage, we labeled each session using the Browscap user-agent parser [5]. This parser takes as input a user-agent and returns, among other information, one of the following agent types: Library, Email Client, Media Player, Feed Reader, Application, Browser, Bot and Unknown. All sessions whose user-agent was identified as Bot were considered robots.

In the second stage, we used two lists containing regular expressions that match with the user-agent string of known bots. The first one[6] is the official list of user agents that are regarded as robots/spiders by project COUNTER[7], which provides a code of practice that helps librarians and publishers record and report online resource usage stats in a consistent and credible way. The second one[8] is a regularly updated list that is used by the open source web analytics software Matomo[9]. All sessions whose user-agent string matched one of the regular expressions were considered robots in addition to those in the first stage.

In the third stage, we simply checked if the session contains a request to the *robots.txt* file. Usually, there are no external or internal hyperlinks leading to this hidden resource. A request to this file indicates that the session belongs to a robot. This file defines the resources a web robot can access and harmful bots typically ignore it. This step is a standard labeling process Stevanovic et al. (2013).

In the fourth and final stage, in an effort to label more sessions, we manually labeled the user agents that were marked as Unknown from the first stage, were not identified as robots by the two lists and did not visit the robots.txt resource. For each unique user-agent, we searched the web for a related application. If the application can access websites without human intervention, we considered this user-agent a bot (e.g. the Papers application[10]). Furthermore, all user agents associated with a programming library (e.g. HttpClient[11]) or with custom names and uncommon format (e.g. dummy) were also considered bots.

[4] `https://www.elastic.co/`
[5] https://browscap.org/ - Version 6000031
[6] github.com/atmire/COUNTER-Robots - Accessed 28-Mar-2019
[7] www.projectcounter.org - Accessed 15-July-2019
[8] https://bit.ly/2XSDjzI - Accessed 28-Mar-2019
[9] matomo.org - Accessed 15-July-2019
[10] www.readcube.com/papers/ - Accessed 27-March-2019
[11] hc.apache.org - Accessed 27-March-2019

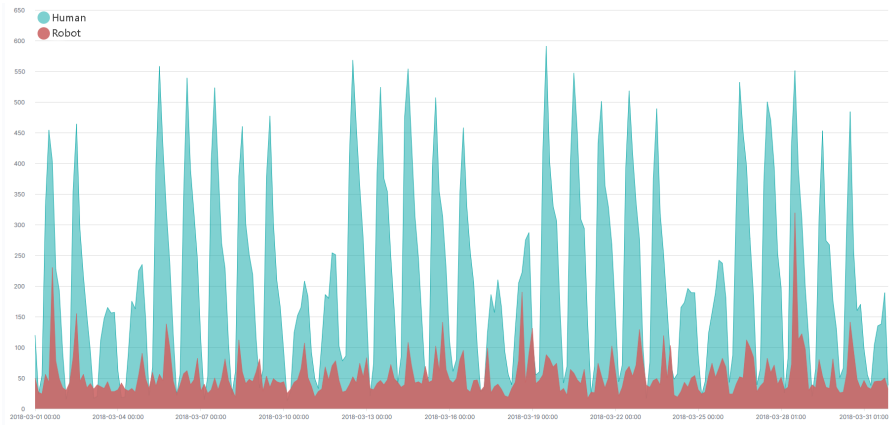

Fig. 5 Human and Robot session distribution over time using the starting timestamp in 3 hour intervals.

Finally, we considered the sessions not labeled as bot from the above procedure as human sessions. In total, 13,494 ($\approx$ 20%) sessions were identified as bot sessions while the remaining 53,858 ($\approx$ 80%) were considered human sessions. We did not further label the sessions into good (well-behaved) or bad (harmful) bots, as other works suggest (Stevanovic et al., 2013; Zabihimayvan et al., 2017), since we believe it is extremely difficult to determine the intentions of a bot without manually inspecting its actions and current labeling methods would create noisy data.

Figure 5 shows the session distribution over time for humans and robots and Figure 6 shows a heatmap of the human and robot sessions by day of the week and hour of the day. We first notice that robot sessions do not follow the same pattern of spikes as the human sessions, where five long spikes are followed by two short ones. Additionally, we notice that human sessions take place during the weekdays and especially during the morning hours, while the robot sessions are spread across the week and day. Both types of sessions are minimal during the night but there are cases where the robot sessions exceed the human ones. The observations above show a strong diversity in traffic patterns of humans and robots.

5 Results

We contribute empirical results concerning the robustness of the proposed content-based features for web robot detection in our real-world case study. We first compare our three different content-aware approaches with each other. Then, we compare and contrast our best content-aware approach with the state of the art while we discuss an effective baseline.

All approaches were evaluated using time series splits validation which is a variation of the k-fold validation method. The number of splits was set to

10. The sessions are first sorted in a time-ordered way and in the k^{th} split the first k-folds are used as the train set and the $k+1^{\text{th}}$ fold is used as the test set. The training set contains only sessions that occurred before the test, following a real-world deployment. Successive training sets are supersets of those that come before them.

The log-based features in Table 1 can be easily extracted from the log file requests, while for the content-based features we first apply the LDA algorithm on the full corpus of the 575,071 records that were extracted during the preprocessing of our dataset. The size of the dictionary was set to 100,000 words after removing stop-words and punctuation characters. The number of topics, k, was set to 500, after experimentation with different numbers of topics ranging from 10 to 100 with a step of 5 and from 100 to 800 with a step of 50. We narrow down the number of topics in a range of 50 (475 - 525) by optimizing the perplexity. The final number of topics was chosen based on the number of documents for which the model produces topics with probability higher than the default probability for each topic, $1/k$. The higher the number of documents with non-default probabilities, the richer the information inferred by the content-aware features.

The XGBoost algorithm (Chen and Guestrin, 2016) is used as the learning algorithm in all cases as it outperforms other algorithms, such as Random Forest, SVM, and logistic regression, with which we experimented. This tree boosting algorithm is also chosen for its scalability and its integrated regularization techniques. The tree depth was set to 15 while the other parameters were set to default values.

5.1 Comparison of content-aware approaches

Table 4 shows the F-measure, Balanced Accuracy, G-mean and Jaccard similarity coefficient score of the three content-aware approaches presented in

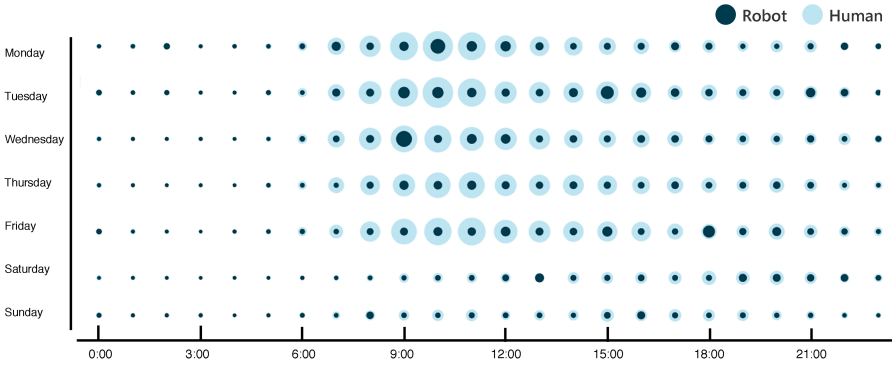

Fig. 6 A heatmap showing the number of human and robot sessions by day of week and hour of day.

Table 4 Comparison of the proposed content-aware approaches.

Approach	F-measure	Bal. Accuracy	G-mean	Jaccard
CBF	**0.9593**	**0.9601**	**0.9598**	**0.8772**
CBS	**0.9593**	0.9584	0.9580	0.8757
CBA	0.9591	0.9581	0.9577	0.8751

Table 5 Top-20 features sorted by average rank of the XGBoost importance score and the absolute Pearson correlation score between each feature and the label.

Feature Name	Rank	XGBoost Score	Pearson
%Unassigned	1	0.5798	0.8070
Max Barrage	5	0.0260	-0.3649
Depth	6	0.0131	-0.4277
Unique Topics	6	0.0119	-0.2983
Total Pages	8	0.0288	0.2536
Loog Penalty	9.5	0.1053	-0.1249
SF Referrer	9.5	0.0120	-0.3196
Page Variance	12	0.0027	-0.2097
SD Depth	13	0.0050	-0.3970
%Repeated	13.5	0.0095	-0.2134
%Consecutive	14.5	0.0043	-0.3449
Average Time	15	0.0067	0.2554
Image Ratio	16.5	0.0040	-0.3035
%Night	16.5	0.0073	0.1867
%Images	17	0.0211	-0.0410
PPI Score	17	0.0148	0.0515
Duration	17	0.0117	0.0766
Data	18	0.0088	0.0857
Total Requests	18	0.0086	-0.0870
Page Similarity	18	0.0099	0.0206

Section 3. We first notice that the content-aware approach with the hand-crafted features (CBF) achieves the best results. However, the difference with the other approaches that use the topics vectors (CBS and CBA) is minimal. All the approaches achieve very good results with the F-measure score reaching over 94%.

Furthermore, Table 5 presents the top-20 features sorted by average rank using the features' importance scores from the XGBoost classifier and the absolute Pearson correlation score between each feature and the label. Three out of the five content-based features of the CBF approach are included in the list. This justifies our initial hypothesis of the use of content-based features for web robot detection.

5.2 Comparison with the state of the art

We compare our best content-aware approach, CBF, with 4 different approaches introduced in the past and a simple supervised approach. Specifically, we compare it against PTABLE (Kwon et al., 2012a), which extracts patterns in the form of the sequence of file request types from all sessions and then creates a pattern table containing the number of matched sessions, SOM (Stevanovic et al., 2013), which uses the SOM clustering algorithm to categorize visitors characterized by 9 features, SMART (Zabihimayvan et al., 2017), which first performs a feature selection based on fuzzy rough sets out of 30 different features and then runs a Markov clustering algorithm, and finally, SSOM (Hamidzadeh et al., 2018), which uses the same feature selection method based on fuzzy rough sets, but the clustering is performed using the SOM algorithm. The simple supervised (SS) approach uses only the universal features presented in Table 1 with an XGBoost classifier.

For the SOM algorithm we use the *somoclu* library[12]. The network consisted of 100 neurons over a 10-by-10 hexagonal arrangement and was trained for 200 epochs. SMART was adapted from the original repository[13] and deployed into Python. The open source Markov Clustering library[14] was also used. The FRS_Threshold parameter was set to 30 considering the number of features and the authors' suggestion.

Table 6 shows the F-measure, Balanced Accuracy, G-mean and Jaccard similarity coefficient score for the four approaches described above along with the best approach of this work (CBF). We first notice that the best results in all measures are achieved by the CBF approach. SMART, which only uses log-based features, achieves the second best results in all measures, but the score difference between the first two approaches is quite significant. These findings are in line with our initial hypothesis that content-based features make useful representations of sessions for web robot detection. We also notice that SOM and SSOM achieve lower scores with a difference of more than 15% compared to the other two approaches. This indicates the inability of SOM to correctly classify sessions since both methods use it. We finally notice that the PTABLE method achieves the lowest performance. This is probably caused by the excessively large pattern table (almost half the number of sessions) that is created due to the length and variety of session patterns.

While the difference between SS and CBF is narrow, with an F-measure increase of 1.75%, the two approaches have statistically different performance with a p-value< 0.001 using the McNemar's test. In a real world system, even a small increase in detecting web robots can greatly reduce the cost of maintenance since an undetected bot may cause significant damage to it or lead to prolonged downtime. However, it is very interesting to note the outstanding performance of SS. Previous approaches mainly focused on sophisticated

[12] `https://github.com/peterwittek/somoclu`

[13] `https://github.com/RezaSadeghiWSU/SMART`

[14] `https://github.com/guyallard/markov_clustering`

Table 6 Comparison of state-of-the-art approaches in web robot detection with the proposed content-aware approach.

Approach	F-measure	Bal. Accuracy	G-mean	Jaccard
PTABLE	0.4723	0.5643	0.5239	0.2878
SOM	0.7899	0.7689	0.7234	0.4785
SMART	0.9127	0.9260	0.9254	0.7564
SSOM	0.7520	0.7314	0.6752	0.4090
CBF	**0.9593**	**0.9601**	**0.9598**	**0.8772**
SS	0.9428	0.9523	0.9521	0.8325

approaches and overlooked simple supervised approaches and their potential when used with appropriate features. Our results show their superiority over the previous approaches and the importance of log-based features. Thus, we recommend using the features in Table 1 along with an ensemble method, such as XGBoost, as a strong baseline in future studies on web robot detection.

6 Conclusion & Future Work

We introduced a novel way of representing web sessions in the context of web robot detection, by taking advantage of the content available in web applications. These features assess the semantic coherence of the content visited within a web session, inspired from a simple assumption: typically, humans look for specific information on a particular subject, while on the other hand, robots go through the content of a website without any preference to the actual content.

We performed an empirical study on real world data originating from the search engine of Aristotle University of Thessaloniki's library, which we also make publicly available. Our experiments validate our assumption that content-based features can boost the predictive accuracy of web robot detection techniques. Our supervised learning method, evaluated with a variety of measures, outperforms state-of-the-art approaches proposed in the past. Finally, our study uncovered a simple, transparent and effective baseline for web robot detection.

In the future, we aim at reaping more benefits out of the proposed concept by constructing content-based features that can better characterize the (in)coherence of a session. Toward this, we plan to explore other algorithms for extracting features of the content visited in a session by using document representations such as Doc2Vec (Le and Mikolov, 2014) and other representations based on word embeddings, such as fastText (Bojanowski et al., 2017) and BERT (Devlin et al., 2018).

Acknowledgements The authors would like to thank Theodoros Theodoropoulos and Aikaterini Nasta from Aristotle University's Central Library for their overall help on providing the data.

References

AlNoamany YA, Weigle MC, Nelson ML (2013) Access patterns for robots and humans in web archives. In: Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries, ACM, pp 339–348

Ansari ZA, Sattar SA, Babu AV (2017) A fuzzy neural network based framework to discover user access patterns from web log data. Advances in Data Analysis and Classification 11(3):519–546

Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. Journal of machine Learning research 3(Jan):993–1022

Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics 5:135–146

Bomhardt C, Gaul W, Schmidt-Thieme L (2005) Web robot detection-preprocessing web logfiles for robot detection. In: New developments in classification and data analysis, Springer, pp 113–124

Brown K, Doran D (2018) Contrasting web robot and human behaviors with network models. arXiv preprint arXiv:180109715

Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, ACM, pp 785–794

Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805

Doran D, Gokhale SS (2016) An integrated method for real time and offline web robot detection. Expert Systems 33(6):592–606

Doran D, Morillo K, Gokhale SS (2013) A comparison of web robot and human requests. In: Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ACM, pp 1374–1380

Dots G (2018) 2018 bad bot report. URL `https://www.globaldots.com/bad-bot-report-2018/`, (Last accessed 11-June-2019)

Ferrara E, Varol O, Davis C, Menczer F, Flammini A (2016) The rise of social bots. Communications of the ACM 59(7):96–104

Foundation O (2018) Owasp automated threat handbook web application version 1.2. URL `https://www.owasp.org/index.php/File:Automated-threat-handbook.pdf`, (Last accessed 20-September-2018)

Greene JW (2016) Web robot detection in scholarly open access institutional repositories. Library Hi Tech 34(3):500–520

Hamidzadeh J, Zabihimayvan M, Sadeghi R (2018) Detection of web site visitors based on fuzzy rough sets. Soft Computing 22(7):2175–2188

Kang H, Wang K, Soukal D, Behr F, Zheng Z (2010) Large-scale bot detection for search engines. In: Proceedings of the 19th international conference on World wide web, ACM, pp 501–510

Kwon S, Kim YG, Cha S (2012a) Web robot detection based on pattern-matching technique. Journal of Information Science 38(2):118–126

Kwon S, Oh M, Kim D, Lee J, Kim YG, Cha S (2012b) Web Robot Detection based on Monotonous Behavior. Proceedings of the Information Science and Industrial Applications pp 43–48

Lagopoulos A, Tsoumakas G, Papadopoulos G (2018) Web robot detection: A semantic approach. In: 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, pp 968–974

Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: International conference on machine learning, pp 1188–1196

Lee J, Cha S, Lee D, Lee H (2009) Classification of web robots: An empirical study based on over one billion requests. computers & security 28(8):795–802

Networks D (2019) 2019 bad bot report. URL `https://resources.distilnetworks.com/white-paper-reports/bad-bot-report-2019`, (Last accessed 11-June-2019)

Rude HN, Doran D (2015) Request type prediction for web robot and internet of things traffic. In: 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), IEEE, pp 995–1000

Stassopoulou A, Dikaiakos MD (2007) A probabilistic reasoning approach for discovering web crawler sessions. In: Advances in Data and Web Management, Springer, pp 265–272

Stassopoulou A, Dikaiakos MD (2009) Web robot detection: A probabilistic reasoning approach. Computer Networks 53(3):265–278

Stevanovic D, An A, Vlajic N (2012) Feature evaluation for web crawler detection with data mining techniques. Expert Systems with Applications 39(10):8707–8717

Stevanovic D, Vlajic N, An A (2013) Detection of malicious and non-malicious website visitors using unsupervised neural network learning. Applied Soft Computing 13(1):698–708

Suchacka G, Sobkow M (2015) Detection of internet robots using a bayesian approach. In: 2015 IEEE 2nd International Conference on Cybernetics (CYBCONF), IEEE, pp 365–370

Tan PN, Kumar V (2004) Discovery of web robot sessions based on their navigational patterns. In: Intelligent Technologies for Information Analysis, Springer, pp 193–222

Zabihi M, Jahan MV, Hamidzadeh J (2014) A density based clustering approach for web robot detection. Proceedings of the 4th International Conference on Computer and Knowledge Engineering, ICCKE 2014 pp 23–28, DOI 10.1109/ICCKE.2014.6993362

Zabihimayvan M, Doran D (2018) Some (non-) universal features of web robot traffic. In: 2018 52nd Annual Conference on Information Sciences and Systems (CISS), IEEE, pp 1–6

Zabihimayvan M, Sadeghi R, Rude HN, Doran D (2017) A soft computing approach for benign and malicious web robot detection. Expert Systems with Applications 87:129–140