Web Robot Detection: A Semantic Approach

Athanasios Lagopoulos
School of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece
lathanag@csd.auth.gr

Grigorios Tsoumakas
School of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece
greg@csd.auth.gr

Georgios Papadopoulos
Atypon Systems LLC
Santa Clara, USA
georgios@atypon.com

***Abstract*—Web robots constitute nowadays more than half of the total web traffic. Malicious robots threaten the security, privacy and performance of the web, while non-malicious ones are involved in analytics skewing. The latter constitutes an important problem for large websites with unique content, as it can lead to false impressions about the popularity and impact of a piece of information. To deal with this problem, we present a novel web robot detection approach for content-rich websites, 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 representation of user sessions with a novel set of features that capture the semantics of the content of the requested resources. Empirical results on real-world data from the web portal of an academic publisher, show that the proposed semantic features lead to improved web robot detection accuracy.**

***Index Terms*—web robot detection, crawler, semantics. content analysis, supervised learning.**

I. Introduction

Web (ro)bots, also known as web crawlers, are computer programs that request resources from web servers across the Internet without human intervention. The constant growth of Web 3.0 technologies and social media generate a huge amount of valuable information ready to be accessed by both traditional web crawlers and emerging advanced robots representing Internet of Things devices, such as smart watches, cars and digital assistants [15]. As of 2016, the web traffic originated by web bots constitutes more than a half (51.8%) of the total web traffic, being in an uptrend after three years of decline [24].

Malicious bots threaten the security, privacy and performance of a web application. *Non-malicious* bots are involved in analytics skewing, affecting the reliability of metrics and, by extension, the decision making process [5]. A recent industry report [13] points out that large websites with unique content, such as blogs, on-line newspapers, e-gov portals and digital libraries are the most attractive to bots. The most common threat that such websites need to deflect is skewing: their metrics and ratings are altered, intentionally by malicious robots and unintentionally by non-malicious robots, rendering their validity questionable and giving the false impression that some piece of information is highly popular and recommended by many [10]. In addition to this, social bots contribute further to the spread of unverified information or rumors [8]. Therefore, the detection of web bots and the filtering of their activities are important tasks within the fight for a trustworthy web.

This paper introduces a novel web robot detection approach for content-rich websites. 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 irrespectively of their content, with the exception of a special class of web robots, called focused crawlers [4]. Based on this assumption, our main contribution is a novel class of features that aim at quantifying the semantic variance of the web content requested within a session. Correspondingly, our main research question is whether such features can improve the results of supervised learning approaches to web robot detection in content-rich websites.

The rest of this paper is structured as follows: After providing the background and related work in Section II, we introduce our approach on extracting semantics from sessions in Section III. In Section IV, we describe our real world case study by presenting the data and the steps taken before creating a detection model, while in Section V we discuss the results of our study proving our assumption. Finally, in Section VI, we review our approach and draw some future directions.

II. Background and Related Work

A. Web Robot Taxonomy

Several categories of malicious web robots have been defined based on their behavioral characteristics and range of capabilities. *Scrappers*, for example, collect application content and data for use elsewhere and obtain limited-availability goods and services by unfair methods [12]. *Hacker tools* are involved in credit card, credential and token cracking and target resources of the application and database servers to achieve denial of service (DDOS), while *impersonators'* functionality spans from account creation and spamming to ad fraud via false clicks and sniping by performing last minute bids for goods [21]. In addition, a taxonomy listing the automated threats on web applications [9] has been created by the community of the Open Web Application Security Project[1] (OWASP).

Similarly, a variety of different categories of non-malicious bots have been defined. *Search engine crawlers* collect infor-

[1] www.owasp.org/ - Accessed 20-Sept-2018

mation to improve their ranking algorithms, *feed fetchers* carry the information of a website to a web or mobile application, while *monitoring* bots help developers keep track of the health and function of their website [7]. Feed fetchers comprise more than 12% of the total traffic, with Facebook's mobile application feed fetcher being the most active web robot accounting for 4.4% of all website traffic [24].

B. Web Robot Detection Approaches

There are four main categories of web robot detection approaches: *syntactic log analysis*, where string processing techniques are used; *analytical learning*, which make use of machine learning algorithms and features that contain different properties deriving from the user sessions in the server; *traffic pattern analysis*, which search for statistical diversity between the features of human and robots; *Turing test systems*, which identify a robot in real-time by means of a Turing test. Recent work on web robot detection mainly focuses on analytical learning approaches, which achieve considerably better results than the other categories of approaches, since the latter rely on procedures or algorithms that a properly engineered bot can evade [5].

C. Analytical Learning

An important first step in analytical learning approaches is *session identification*, which is concerned with breaking the click stream into sessions. Various timeout thresholds have been investigated in the past, such as 10 minutes [1], 30 minutes [20] and using dynamically adaptive thresholds ranging from 30 minutes to 60 minutes [16].

An important second step concerns feature extraction from the identified sessions, based on the variety of information found in the entries of web server access logs, such as: 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 (e.g GET, HEAD, POST), the HTTP response code sent back to the client (200, 404 etc.), the *size* of the returned object, the *Referer HTTP request header*, which is the page that links to the resource requested and the *User-Agent String* that identifies the client's browser. Fig. 1 shows a sample entry in a server access log. Some of the typical features extracted from identified sessions are:

- *Total Requests*. The total number of requests.
- *Session Duration*. Total time, in seconds, elapsed between the first and the last request.
- *Average Time*. Average time, in seconds, between two consecutive requests.
- *Standard Deviation Time*. The standard deviation of the time between two consecutive requests.
- *Repeated Requests*. Percentage of repeated requests. A repeated request is a request for an already visited page using the same HTTP method as previously.
- *HTTP requests*. Four features, each containing the percentage of requests associated with one of the following HTTP response codes: Successful (2xx), Redirection (3xx), Client Errors (4xx) and Server Errors (5xx).
- *Specific Type Requests*. The percentage of requests of a particular type over the number of all requests. This feature is application dependent.

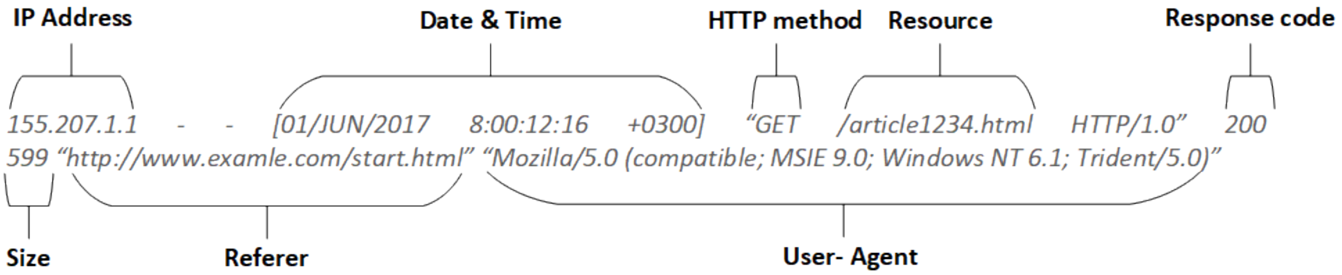
Many websites are based on content management systems or specialized web applications that log additional information about the identity of the visitors of a website, beyond that found in web server access logs. Such information may be the country that the request is coming from, by checking the user's IP using a geolocation service; the username of a logged in user or an indicator if the request originates from a web service. This kind of information can be a great source of valuable features for a web robot detector, but unfortunately it normally is application dependent and not always available.

D. Related Work

Several supervised analytical learning approaches have been developed in the past based on a variety of learning algorithms and features. Tan and Kumar [20] used decision trees (C4.5 algorithm) to train a model using 25 different features that were extracted from each user session. The feature vector included percentages of the different content (images, multimedia, HTML etc.), time characteristics (average time, total time etc.), request types (GET, POST, HEAD etc.) and other (IP, user-agent etc.). Bomhardt et al. [3] used neural networks and included features like total number of bytes and percentage of response codes (200, 2xx, 404 etc.). Stassopoulou and Dikaiakos [17] used a heuristic semi-automatic method to label the training data and introduced a Bayesian approach to classify the sessions. Stevanovic et al. [18] experimented with a variety of classifiers (C4.5, RIPPER, k-nearest, Naive Bayesian, Bayesian Network, SVM and Neural Networks) and introduced two novel features considering the page depth of a session's requests and the sequentiality of HTTP requests. Finally, Doran and Gokhale [6] introduced a novel approach that can be used for real-time detection of web robots. Their approach is based on a first-order discrete time Markov chain model and the request patterns of the visitors. To the best of our knowledge, our work is the first to consider semantic features in a supervised learning approach for web robot detection, though a very recent work mentions it among its future work agenda [23].

In contrast with the above supervised approaches, Stevanonic et al. [19] used unsupervised neural networks to detect humans and robots to further analyze the behavior of malicious and non-malicious web robots, while Zabihu et al. [22] used the DBSCAN clustering algorithm with just four different features.

Few studies have addressed web robot detection in the domain of academic publishing, where our empirical study is focused. The first work to examine this domain [11], compared the activity of robots in open access and restricted full text articles of a biology journal. Robots were identified using different heuristic methods and behavioral pattern techniques without using any machine learning. A second recent study benchmarked existing web robot detection approaches in Open Access institutional repositories [10]. By performing a close



```
155.207.1.1   -   -   [01/JUN/2017   8:00:12:16   +0300]   “GET   /article1234.html   HTTP/1.0”   200
599 “http://www.examle.com/start.html” “Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)”
```

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

review of the literature, system documentation and open source code, the study concludes that web robot detection is most successful when a variety of data and techniques are combined and pinpoints that none of the examined methods leads to usage statistics that are completely free of robot activity.

III. Extracting Semantics from Sessions

As already mentioned in the introduction, this work assumes that typically humans look for specific information on a particular topic, while web bots go through the content of a website in a uniform fashion, 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 session. To achieve this, we start with topic modeling of the content of a website using latent Dirichlet allocation (LDA) [2]. 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. LDA was chosen for its modularity and interpretability.

Consider a session, S, comprising n requests for web pages (or other textual resources, such as PDF files). 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 p_i be a vector containing the distribution over the k topics for the web page associated with request i. We propose the extraction of the following *semantic features* from S:

- *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.

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

  where $\overline{p} = \frac{1}{n}\sum_{i=1}^{n} p_i$ is the mean of the vectors p_i 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.

IV. Real World Case Study

Our real world case study is concerned with web robot detection in the digital library of a commercial academic publisher. First, we present the dataset and the procedures we followed to pre-process it. Then, we discuss our session identification approach. Next, follows a description of the features that were extracted from each session. Finally, we discuss the procedures that we followed in order to label a sample of the sessions as robot and human.

A. Dataset Preparation

Our data come from the web portal of a large commercial academic publisher, which hosts articles from a variety of scientific domains, and span an entire month (January 2014). We removed from the original server access log those entries that didn’t contain sufficient length of text or any semantic value, such as search results, home page, log in page, about page, contact page, etc. The remaining 25,318,451 requests are associated with the available scientific articles in the library and belong to one of the following types:

- Abstract (HTML): Web page of an article's abstract.
- Full-text (HTML): Web page of an article's full-text.
- Full-text (PDF): PDF file of an article's full-text.
- References (HTML): Web page containing the references of an article.
- Supplementary Material (HTML): Web page containing supplementary material (tables, data, etc.) of an article.

Besides the log files we were also provided with the full corpus of the available articles, written in English, in the digital library. In total, there were 2,253,533 different articles in XML format.

B. Session Identification

This step of our pipeline groups together requests into sessions. We first group together requests that share the same IP address and user-agent string. These groups are then broken into sessions by applying a timeout threshold of 30 minutes, which appears to be the literature standard. This process identified 10,039,241 sessions. Furthermore, we ignore sessions containing less than 3 requests, as no meaningful behavioral patterns can be extracted from such cases. This led to 1,727,568 sessions.

These sessions vary a lot, both in number of requests and in duration. They have an average (median) of 7.8 (4) requests, while there are sessions with more than 10,000 requests. Their average (median) duration is 639 (209) seconds and the average time between two consecutive requests is 132 seconds. 1,653,999 unique articles are accessed in total within these sessions.

C. Feature Extraction

We extracted two sets of features from the identified sessions: *simple features*, based on past approaches of the literature, and *semantic features*, based on the approach that we presented in this paper.

Simple features include the features discussed in Section II-C. We customize the feature *Specific Type Requests* to *PDF requests*, measuring the percentage of PDF requests over the number of all requests, so as to match our case study. In addition, simple features include the following features that come from the particular web application of the publisher:

- *Unique Content*. Number of distinct scientific articles requested in a session regardless their format (e.g. HTML, PDF, full-text, abstract, etc.).
- *Web Service*. Indicates whether the session comes from a web service or an application programming interface (API) of the web application or not.

To extract the semantic features we first applied the LDA algorithm on the full corpus of the 2,253,533 articles, since every request is associated with an article as specified in Section IV-A. The title, the abstract and the text of the article were given as input. After initial testing, the number of topics, k, was set to $5,000$, but for each article only the top 10 topics with the highest probabilities were considered.

D. Session Labeling

The robot labeling procedure consists of three stages. During the first stage we label each session using the API of useragentstring.com[2]. This API takes as input a user-agent and returns, among other information, one of the following agent types: Cloud Client, Console, Offline Browser, Link Checker, Crawler, Feed Fetcher, Library, Mobile Browser, Validator, Browser, Unknown and Other. All sessions whose user-agent was identified as Crawler were considered robots.

In the second stage, we use two lists containing regular expressions that match with the user-agent string of known bots. The first one[3] is the official list of user agents that are regarded as robots/spiders by project COUNTER[4], 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[5] is a regularly updated list that is used by the open source web analytics software Matomo[6]. All sessions whose user-agent string matched one of the regular expressions were considered robots. After the manual inspection of a sample of these user-agent strings, we decided to remove the following questionable regular expressions of the first list: *^Mozilla$, ^Mozilla.4\.0$* and *^Mozilla.5\.0$*. Besides the user-agent strings identified as Crawlers by the API, the two lists considered as robots all user-agent strings identified by the API as Cloud Client, Offline Browser, Link Checker, Feed Fetcher, Library, Validator and Other, and some of the sessions identified as Unknown.

In the third stage, in an effort to label more sessions, we manually labeled the user agents marked as Unknown from the API, which were not identified as robots by the two lists. 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[7]). Furthermore, all user agents associated with a programming library (e.g. HttpClient[8]) or with custom names and uncommon format (e.g. dummy) were also considered bots. From the total of 2,562 user-agent strings, 1,946 were identified as robots. These user agents are mostly WordPress plugins, reference and citation management tools and custom applications.

To identify humans, we use information from the logs of the publisher's web application. In particular, we label as human all the sessions that come from a logged in user. Sessions identified as Browser by the API, cannot be considered human, because robots can mask their user-agent string with one of a known browser [19].

Following the above labeling strategy we managed to label 67,484 out of the 1,727,568 sessions ($\approx 4\%$). Of these labeled sessions, 37,922 ($\approx 56\%$) are labeled as robot and 29,562

[2] www.useragentstring.com/pages/api.php - Accessed 20-Sept-2018

[3] github.com/atmire/COUNTER-Robots - Accessed 20-Mar-2018

[4] www.projectcounter.org - Accessed 20-Sept-2018

[5] goo.gl/5WQ6ds - Accessed 20-Sept-2018

[6] matomo.org - Accessed 20-Sept-2018

[7] www.readcube.com/papers/ - Accessed 20-Sept-2018

[8] hc.apache.org - Accessed 20-Sept-2018

TABLE I

THE SCORES OF THE FEATURES IN INCREASING AVERAGE RANK. THE SEMANTIC FEATURES ARE MARKED IN BOLD.

Feature	F-Test	χ^2 test	Rank
Unique Topics	9696	2188150	4
Session Duration	3736	43153244	4
Boolean Page Variance	55347	72612	4.5
Page Similarity	21269	2803	5.5
Unique Content	5627	310769	5.5
Total Topics	1585	6265438	6
Total Requests	1593	628570	6.5
Repeated Requests	16868	678	7
Page Variance	13679	2208	7
Average Time	909	259440	8.5
Standard Deviation Time	574	153835	9.5
HTTP Client Error	2221	268	10
PDF Requests	458	84	13
HTTP Successful	101	25	14
HTTP Server Error	36	18	15.5
Webservice	23	23	16
HTTP Redirection	25	10	16.5

($\approx 44\%$) as human. Interestingly, the distribution of the target variable in our dataset is similar to the reported distribution of human and bot traffic on the Web.

V. RESULTS

We contribute empirical results concerning the utility of the proposed semantic features for web robot detection in our real-world case study. We first present and discuss measures of the dependency between each feature (both simple and semantic) and the class variable (human vs bot). Then we compare and contrast simple, semantic and both sets of features in conjunction with a variety of machine learning algorithms for building web robot detection models.

A. Feature Evaluation

We discuss the dependency between each feature and the class, as measured by two univariate statistical tests: the F-test in ANOVA and the χ^2 test. Table I presents the scores of all features according to these two tests, sorted by average rank.

We first notice that all semantic features are highly ranked by at least one of the two tests. In particular, four out of the five semantic features (*Boolean Page Variance*, *Page Similarity*, *Page Variance* and *Unique Topics*) are among the top-5 features according to the F-test, while two out the five semantic features (*Total Topics* and *Unique Topics*) are among the top-3 features according to the χ^2 test. These findings are in line with our initial hypothesis that semantic features make a useful representation of sessions for web robot detection in content-rich websites.

We also notice that simple features like *Repeated Requests*, *Session Duration* and *Total Requests* are also ranked high by both tests. This is expected, since long sessions with many and repeated requests are typical of the behavior of web robots.

The *Unique Topics* semantic feature is the only feature to be found among the top-5 features according to both tests. Fig. 2 contrasts the distribution of *Unique Topics* in human sessions with that of robot sessions. It is evident that robot sessions exhibit a much higher number of *Unique Topics* compared to human sessions.

B. Predictive Modeling

We split the original training data in two parts: a training set containing 70% and a test set containing the rest 30%. The split is done in a time-ordered way, so that the training set contains only sessions that occurred before the test set, in accordance with a real-world deployment.

We experiment with four different models: a support vector machine with an RBF kernel (RBF), a gradient boosting (GB) model and a multi-layer perceptron (MLP) using scikit-learn [14], as well as an eXtreme Gradient Boosting (XGB) model[9].

Tables II, III and IV present, respectively, the F-measure, Balanced Accuracy and G-mean of each model using only the simple features, only the semantic features and, finally, both the simple and the semantic features.

We first see that the best results in all three evaluation measures are achieved by RBF when the semantic features are used by themselves, and by GB when the simple features are used either by themselves or in tandem with the semantic features (in bold typeface). Considering these best results per feature space used, we notice that a decent level of web robot detection accuracy can be achieved using semantic features alone. Simple features lead to better results compared to semantic features when these two types of features are used by themselves. However, the best results in all three evaluation measures are achieved when using both the simple and the

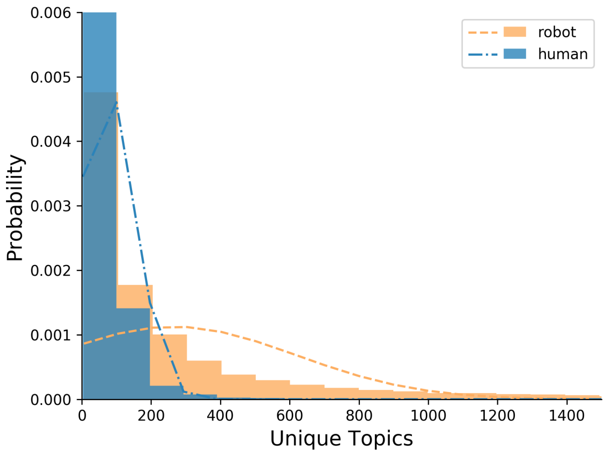


Fig. 2. Probability histograms and corresponding empirical probability density functions (PDF) of the distribution of human and robots sessions' unique topics. Clipped graph (max probability:0.008, max unique topics:2000).

[9] github.com/dmlc/xgboost - Accessed 20-Mar-2018

TABLE II
F-MEASURE OF THE FOUR DIFFERENT MODELS USING ONLY THE SIMPLE, ONLY THE SEMANTIC AND BOTH THE SIMPLE AND THE SEMANTIC FEATURES.

Features \ Classifier	RBF	MLP	GB	XGB
Simple	0.6552	0.7844	**0.9075**	0.905
Semantic	**0.8489**	0.7497	0.8482	0.846
Simple & Semantic	0.6484	0.8166	**0.9181**	0.9177

TABLE III
BALANCED ACCURACY OF THE FOUR DIFFERENT MODELS USING ONLY THE SIMPLE, ONLY THE SEMANTIC AND BOTH THE SIMPLE AND THE SEMANTIC FEATURES.

Features \ Classifier	RBF	MLP	GB	XGB
Simple	0.6551	0.7685	**0.9007**	0.898
Semantic	**0.8484**	0.7712	0.845	0.8418
Simple & Semantic	0.6518	0.801	**0.9133**	0.9127

TABLE IV
G-MEAN OF THE FOUR DIFFERENT MODELS USING ONLY THE SIMPLE, ONLY THE SEMANTIC AND BOTH THE SIMPLE AND THE SEMANTIC FEATURES.

Features \ Classifier	RBF	MLP	GB	XGB
Simple	0.5835	0.7432	**0.8989**	0.8961
Semantic	**0.8475**	0.7673	0.8432	0.8395
Simple & Semantic	0.5656	0.7815	**0.9123**	0.9116

semantic features (in underlined typeface). In particular, the F-measure is increased by 1.16%, Balanced Accuracy by 1.39% and G-mean by 1.49% compared to using the simple features alone. These findings offer evidence that semantic features can lead to improved web robot detection accuracy in content-rich websites. Finally, we notice that the RBF classifier's performance is degraded when using both the simple and the semantic features compared to when the simple or the semantic features are used by themselves. Increasing the complexity of some algorithms (i.e Support Vector Machine) by using more features, can sometimes decrease the model's discriminative capability between classes and lead to lower scores.

We further validate our assumption with a paired t-test. First, we split our dataset in 10 equally sized consecutive folds and we set the size ratio of the training set to the test set to 2:1 folds. We create training-test set pairs by defining the first two folds as training set and the third one as test test. Then, we slide both the training and the test set window by one fold until the last fold is used as test set. In total, 8 training-test set pairs are created with a fixed size ratio and ordered in time (i.e. TRAIN_1:[1,2] - TEST_1:[3], TRAIN_2:[2,3] - TEST_2:[4] etc.). Finally, we compute the f-measure score for each training-test set pair using the simple features by themselves and in tandem with the semantic features. Our best classifier, the GB algorithm, is used. Our results show that higher scores are always achieved when using both feature spaces. The mean difference of the two is 0.066 and the t-value is 4.31. The difference is strongly statistically significant with a p-value of $0.0035 < 0.01$ which further strengthens our initial assumption.

We conclude the discussion of the results with a learning curve plotting the Balanced Accuracy of the best learning algorithm, Gradient Boosting, using the best representation of the sessions, both simple and semantic features, for a varying number of training examples (Fig. 3). We see that the training and testing curves converge when almost half of the available data is used. We therefore conclude that our model has low variance and there is no need for additional training data to improve the current prediction accuracy. Instead, the complexity of our model should be increased, for example by getting additional features or by using polynomial features. This finding highlights the importance of novel classes of features, such as the semantic features that we propose in this work.

VI. CONCLUSION & FUTURE WORK

We introduced a novel class of features for supervised web robot detection. These features assess the semantic coherence of the content visited within a session, inspired from a simple observation: typically, humans look for specific information on a particular subject, while on the other hand, robots go through the content of a website in a uniform fashion.

We performed an empirical study on real world data originating from the web portal of a commercial academic publisher. Statistical tests verify the correlation of the proposed features with the target variable (bot vs human). The predictive accuracy of a variety of classifiers, evaluated with a variety of measures, improves when semantic features are appended to traditional non-semantic features.

In the future, we aim at reaping more benefits out of the concept of semantic content analysis by constructing addi-

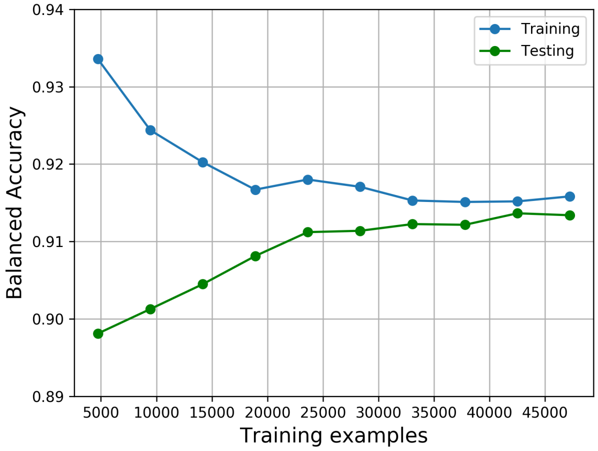


Fig. 3. Learning curve of the Gradient Boosting algorithm using both the simple and the semantic features.

tional features that could better characterize the (in)coherence of a session. Toward this, we plan to fine-tune the parameters of LDA, as well as explore other algorithms for extracting semantic representations of the content visited in a session, such as doc2vec and word2vec.

Our future plans further include the investigation of a multi-class modeling of the web robot detection task, since web robots come with different characteristics and functions. An analysis of the significance of semantic features on the different types of web robots may help us understand their behavior better, while the use of unsupervised learning techniques may reveal new uncategorized robots.

Acknowledgments

This work is partially funded by Atypon Systems LLC [10].

References

[1] AlNoamany, Y.A., Weigle, M.C., Nelson, M.L.: Access patterns for robots and humans in web archives. In: Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries. pp. 339–348. ACM (2013)

[2] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan), 993–1022 (2003)

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

[4] Diligenti, M., Coetzee, F., Lawrence, S., Giles, C.L., Gori, M., et al.: Focused crawling using context graphs. In: VLDB. pp. 527–534 (2000)

[5] Doran, D., Gokhale, S.S.: Web robot detection techniques: overview and limitations. Data Mining and Knowledge Discovery **22**(1), 183–210 (2011)

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

[7] Doran, D., Morillo, K., Gokhale, S.S.: 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. pp. 1374–1380. ACM (2013)

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

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

[10] Greene, J.W.: Web robot detection in scholarly open access institutional repositories. Library Hi Tech **34**(3), 500–520 (2016)

[11] Huntington, P., Nicholas, D., Jamali, H.R.: Web robot detection in the scholarly information environment. Journal of Information Science **34**(5), 726–741 (2008)

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

[13] Networks, D.: Bad bot report (2017), https://resources.distilnetworks.com/whitepapers/2017-bad-bot-report, (Last accessed 20-September-2018)

[14] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

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

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

[17] Stassopoulou, A., Dikaiakos, M.D.: Web robot detection: A probabilistic reasoning approach. Computer Networks **53**(3), 265–278 (2009)

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

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

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

[21] Wang, B., Zheng, Y., Lou, W., Hou, Y.T.: DDoS attack protection in the era of cloud computing and software-defined networking. Computer Networks **81**, 308–319 (2015)

[22] Zabihi, M., Vafaei Jahan, M., Hamidzadeh, J.: A density based clustering approach to distinguish between web robot and human requests to a web server. The ISC International Journal of Information Security **6**(1) (2014)

[23] Zabihimayvan, M., Sadeghi, R., Rude, H.N., Doran, D.: A soft computing approach for benign and malicious web robot detection. Expert Systems with Applications (2017)

[24] Zeifman, I.: Bot Traffic Report (2016), https://www.incapsula.com/blog/bot-traffic-report-2016.html, (Last accessed 20-September-2018)

[10] www.atypon.com/