An Overview of Web Crawlers Detection Techniques

Cyber Science 2020

Hanlin Chen
School of Aerospace, Transport and Manufacturing
Cranfield University
Bedford, United Kingdom
hanlin.chen@cranfield.ac.uk

Hongmei He
School of Aerospace, Transport and Manufacturing
Cranfield University
Bedford, United Kingdom
h.he@cranfield.ac.uk

Andrew Starr
School of Aerospace, Transport and Manufacturing
Cranfield University
Bedford, United Kingdom
a.starr@cranfield.ac.uk

***Abstract*— Web robots or web crawlers have become the major source of web traffic. Some robots are well-behaving such as search engines, while others can perform DDoS attacks, which put great threats on websites. To protect web content from malicious web robots, researchers have proposed various approaches based on machine-learning techniques, such as offline web log analysis, honeypots and online robot detection. This paper gives a comprehensive literature review for the years 2014-2019 to review and compare experimental results of these web robot detection methods, trying to find the research gap of web robot detection. We conclude that off-line web log analysis methods have quite high accuracy, but they are time-consuming compared to online detection methods. Honeypots, as a computer security mechanism, can be used to engage and deceive hackers and identify malicious activities performed over the Internet, but they may block legitimate robots. The review shows that a hybrid method is better than an individual classifier, and the performance of online web robot detection needs to be improved. Also, different types of features could play different roles in different machine learning models. Therefore, feature selection is important for crawler detection.**

Keywords*— *Literature review, Web robot detection, Machine-learning, Web log analysis, Honeypot, Online detection

I. Introduction

The World Wide Web provides a wealth of information about almost every type. The most popular way for people to grasp information from the web is by searching with a search engine [1]. Search engines like Google, Yahoo, MSN, and Bing use web crawlers to index web pages to be used in their page ranking process [2]. Apart from search engines, crawlers can be used to collect data for analysis, monitoring public opinions or even booking tickets. Web crawlers-also known as spiders, robots, walkers, and wanderers – are computer programs that travel across the Internet and collect data from web pages [1][3]. Web robots are almost as old as the Internet itself. Over the years, web robots have achieved a huge increase in numbers as well as a dramatic evolution in technologies. A 2015 industry report suggests that robots constitute 49.5% of all HTTP requests to popular web sites on the internet, while our recent academic report suggests this number to be closer to 60% on academic web servers [4]. However, web robots can be misused if they are in the wrong hands. The illegitimate robots can behave unethically by performing some activities that violate the *robot.txt* file, collect sensitive information and consume large amounts of bandwidth by staging Distributed Deny of Service (DDoS) attacks. This could cause serious damage to the reputation of companies that rely on web traffic [5].

A great deal of time and money was devoted to figuring out how to detect malicious web robots and protecting web content from being attacked by those robots. Researchers have proposed many methods to detect or identify web crawlers. Although the literature covers a wide variety of methods, this review will focus on three major themes that are popularly investigated in the literature, such as offline web robot detection based on web log analysis with machine-learning techniques, honeypots, and online web robot detection. Although the literature presents these methods in a variety of contexts, this paper will primarily focus on the difference between them and try to find the research gap in the field of web robot detection.

II. Web Log Analysis

Web-data mining or machine learning techniques are widely used for the detection of web crawlers. In this section, we will examine the four machine-learning-based web robot detection methods proposed by different groups of researchers from 2015 to 2018 and compare the performance of these methods.

Rajabnia and Jahan [6] proposed a hybrid fuzzy inference system based on NNGE (non-nested generalized exemplar) algorithm. In terms of the feature weights obtained by the NNGE algorithm, the main features are used to train the classification models, thus the dimension of the problem space is reduced. The hybrid inference system was developed to identify web robots by inspecting web log files, and obtained acceptable performance.

Bayesian network is another popular method in web robot detection. Suchacka and Sobków [7] used a Bayesian approach to robot detection based on characteristics of user sessions. In this paper Weak Bayesian Approach (WBA) and Strong Bayesian Approach (SBA) were compared in two variants: (a) using known bots' sessions from the testing samples and (b) using known bots' sessions from the verifying samples. Generally, SBA gave better results in terms of practical accuracy and the best result is shown in Table I. However the choice between WBA and SBA depends on expected results: if it is acceptable to commit a small error of classifying a human as a robot, it would be better to use SBA. But if such errors are unacceptable, one should use WBA.

Sisodia et al. [8] believed the result of web server log analyzers are not very reliable because the input log files are highly inflated by sessions of web robots. They proposed an agglomerative approach combining web logs with actual visitors' knowledge extraction, applied ensemble classifiers (boosting, bagging, and voting), and evaluated the performance of these ensemble learners with some performance indicators, such as recall, precision, and F1

measure. The precision for the web robots sessions with ensemble classifiers is more than 80% in first experiment and 98% in the second experiment.

Haidar and Elbassuoni [9] developed a two-class Boosted Decision Tree (BDT) for web robot detection based on website navigation behavior analysis. The dataset used for their experiments was extracted from two popular websites: wheelers and wherever. The advantage of the system lies in that it can be re-trained as web robots evolve. Experiment results are shown in Table 1. The items in the head of Table 1 are described as below:

- Resource of data: indicate where the data was collected.
- No. of Selected Features: number of features means how many features are selected for the detection.
- TP: true positive, is the percentage of positive cases correctly classified as belonging to the positive class.
- FP: false positive, is the percentage of negative cases misclassified as belonging to the positive class.
- Precision: (also called positive predictive value) is the fraction of relevant instances among the retrieved instances: $Precision = TP / (TP + FP)$
- Recall: (also known as sensitivity) is the fraction of the total amount of relevant instances that were actually retrieved: $Recall = TP / (TP + FN)$
- F1-score is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score, and it is the harmonic mean of the precision and recall: $F1 = 2 \times Precision \times Recall / (Precision + Recall)$

Lagopoulos et al. [10] presented a semantic approach for web robot detection. This method intended to detect web crawlers within content-rich websites. They designed this approach based on the assumption that human web users are interested in specific topics, while robots craw the web randomly.

Typical features extracted from sessions includes:

- *Total Requests*: The number of requests.
- *Session Duration*: Total time elapsed between the first and the last requests.
- *Average Time*: Average time between two consecutive requests.
- *Standard Deviation Time*: the standard deviation of the time between two consecutive requests.
- *Repeated Requests*: 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.

The used semantic features, extracted from a session, are:

- *Total Topics* (TT): The number of topics with non-zero probability.
- *Unique Topics* (UT). The number of unique topics with non-zero probability.
- *Page Similarity* (PS). The ratio of unique topics with non-zero probability over all the topics with non-zero probability.
- *Page Variance* (PV). The semantic variance of the pages of a session.
- *Boolean Page Variance*: It is a boolean version of PV.

The experiments were carried out with four different models: a SVM with an RBF kernel (RBF), a gradient boosting (GB) model, a multi-layer perceptron (MLP), and an eXtreme Gradient Boosting (XGB) model. From Table II, it can be seen that RBF can achieve the best performance on the semantic features only; Except for RBF, all other models, MLP, GB and XGB obtained the best performance on the combination of simple and semantic features; GB and XGB obtain better performance on simple features than on semantic features. From the experimental results on different feature sets, it can be seen that RBF achieved the best performance on semantic features, GB obtained the best performance on simple features, and GB also achieved the best performance on the combination of simple and semantic features [10].

TABLE I. COMPARISON BETWEEN DIFFERENT WEB ROBOT CLASSIFICATION METHODS

Classification Methods	Source of Data	No. of Selected Features	Precision	Recall	F1-score	Accuracy
NNGE-fuzzy Inference system	Pars Web Server	4	0.9931	0.9931	0.9931	0.993
SBA	Real E-commerce Site	20	N/A	N/A	N/A	0.931
Agglomerative Approach	Unclear	23	0.9800	0.9800	0.9800	
BDT	Wheelers	496	0.920	0.765	0.815	0.831
	Whereleb	472	0.916	0.502	0.601	0.731

TABLE II. PERFORMANCE OF THE FOUR DIFFERENT MODELS USING ONLY THE SIMPLE, ONLY THE SEMANTIC AND BOTH THE SIMPLE AND THE SEMANTIC FEATURES [10]

Performance Indicator	Feature sets	RBF	MLP	GB	XGB
F1-score	Simple	0.655	0.784	**0.907**	0.905
	Semantic	**0.848**	0.749	0.848	0.846
	Both	0.648	0.816	**0.918**	0.917
Accuracy	Simple	0.651	0.768	**0.900**	0.898
	Semantic	**0.848**	0.771	0.845	0.841
	Both	0.651	0.801	**0.913**	0.912
G-mean	Simple	0.583	0.743	**0.898**	0.896
	Semantic	**0.847**	0.767	0.843	0.839
	Both	0.565	0.781	**0.912**	0.911

III. HONEYPOT

Honeypot is represented as an invisible link or a vulnerable web page that intentionally designed by developers in order to trap web crawlers. A human, who is using a browser to read a web page, cannot see the links or other resources that have been hidden from the view window of the website. However, a web crawler, looking at the source code, does not check the format before requesting it. Based on this assumption, McKenna [11] proposed a strategy to detect and classify web robots with honeypots. In this case, a methodology based on honeypot techniques for hiding content that were explicitly defined would be more likely to be ignored by screen readers than the ones which were not. In contrast, "constructed" formatting techniques such as hiding content through indentation or clipping are more likely be read by screen readers. They used a CSS rule named *display:none* to construct hidden contents. *pdf, doc* and *html* files were added to the hidden contents. Besides, they built a sandtrap, which implements a server-side PHP script to catch crawlers. Also, they employed a "one-strike" rule to classify bots, which accessed the honeypots. If a bot failed to check our site's robots.txt file or failed to comply with its directives, it was classified as a "bad" bot. If a bot accessed the honeypot, checked the robox.txt file, and followed the directives, it was classified as a "good" bot. Experimental results showed naive honeypots did not produce an effective confirmation of malicious web crawlers. It may be helpful, to match the honeypot resource with the focus of the crawler [11].

In 2015, a specialized data collection system for crawler detection, called Lino, was developed. It simulates a vulnerable web page and traps web crawlers. From collected features, they select features which mostly contribute to the classification of visitor behavior, and used machine learning techniques, such as Support Vector Machine (SVM) and decision tree C 4.5, for detecting web crawlers. With Lino they selected top 5 features that dominate the dataset including:

- *Post data*, which shows us whether the client has filled/not filled the fake form in the Lino system.
- *Session change*, which shows us if user, during the session, has changed the session identifier or not.
- *Session duration*, duration of the session in seconds.
- *Robots*, which shows us whether the user accessed /not accessed to the robots.txt file, which defines the rules of robot conduct.

TABLE III. PERFORMANCE OF C 4.5 AND SVM, EXPERIMENT 1 USES ONLY SELECTED FEATURES. EXPERIMENT 2 USES SELECTED FEATURES PLUS COUNTRY AND ASN OF A CLIENT.

	Class	TP	FP	F1	AUC
C 4.5 Experiment #1	Human	0.177	0	0.301	0.773
	Robot	1	0.823	0.972	0.773
C 4.5 Experiment #2	Human	0.793	0.002	0.872	0.985
	Robot	0.998	0.207	0.992	0.985
SVM Experiment #1	Human	0.625	0	0.419	0.801
	Robot	1	0.735	0.979	0.801
SVM Experiment #2	Human	0.962	0.006	0.942	0.976
	Robot	0.998	0.042	0.997	0.978

The main drawback of the crawler detector with selected features lies in that the crawler detector on the collected data gave a high false positive rate for robot detection. This could drop the performance of the web browser, as users may not see what they want [12].

Priyanka et al. [13] proposed a similar solution to detect malicious web crawler. In their work, honeypot was utilized but in a different way. The flow chart of the system is shown in Fig. 1.

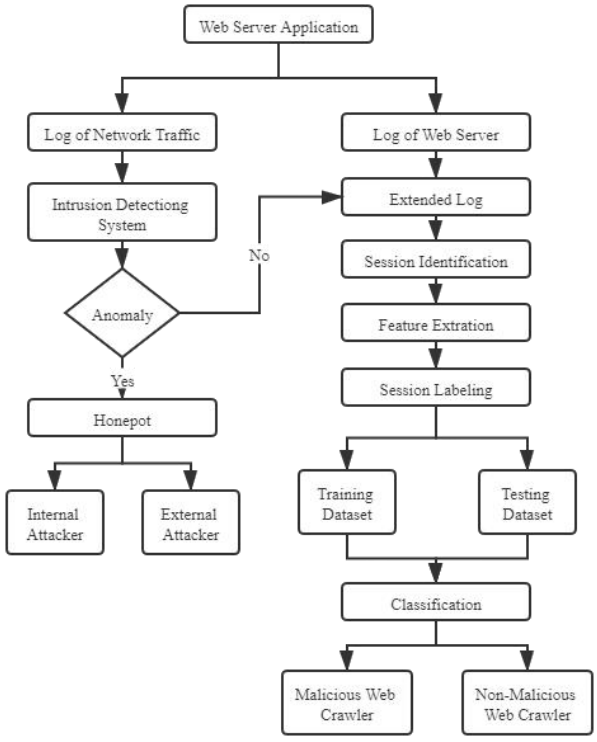

Fig. 1. Data flow Diagram (DFD) of the crawler detection system [11]

They combined honeypot with intrusion detection system to identify web crawler sending DDoS attacks. The system will divert the malicious crawlers to a honeypot so that the server will be safe.

IV. ONLINE WEB CRAWLER DETECTION

Most of existing research focused on offline web robot recognition, based on information on user sessions in web logs. Very few approaches aimed at detecting web robot online [14]. In 2019, Wan et al. [15] proposed a new anti-crawler, called PathMarker, to detect and restrain distributed

crawlers. They can trace the page that leads to the access to an URL by adding a marker to the URL and identify the user, who accesses to this URL. A Support Vector Machine was utilized to distinguish malicious web crawler from normal users, in terms of different patterns of URL visiting paths and visiting timings. Experimental results showed the proposed system could successfully identify 96.74% crawlers' long sessions and 96.43% normal users' long sessions. The architecture of PathMarker is shown in Fig. 2.

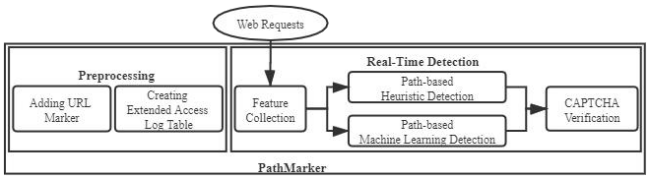

Fig. 2. PathMarker Architecture

The PathMarker system has two processes: Preprocessing and Real-time detection. In preprocessing period, the system will dynamically add a URL marker at the end of each hyperlink on every web page. Then the system will create an extended access log table containing typical informations including IP address, the pages URL being visited, timestamp, and extended informations, such as visitors' ID and corresponding marker's information. In real-time detection period, first the system will collect features from web requests, then the path-based heuristic detection will perform basic analysis on the traffic, trying to find crawlers based on basic features such as User-Agent. At the same time , six new features will be input to the Machine learning detection model to detect web crawlers. At last, Completely Automated Public Turing test to tell Computer and Humans Apart (CAPTCHA), will fainally identify web robots.

The classification results are shown in the Table IV. Type 0 represents normal users. Type 1, 2, and 3 represents crawlers that expose extrusive breadth-first, depth-first, and random-like crawling features, respectively. We can see that the system could successfully identify 96.74% crawlers' long sessions and 96.43% normal users' long sessions.

TABLE IV. CONFUSION MATRIX OF PATHMARKER

Original type	Classify as 0	Classify as 1	Classify as 2	Classify as 3
0	96.43%	0%	3.57%	0%
1	0%	100%	0%	0%
2	0%	6.25%	93.75%	0%
3	1.51%	1.77%	0%	96.72%

Apart from web robot identification, Wan et al. [16] also conducted two sets of simulation experiments to study the impacts of PathMarker on distributed crawlers. First, they assumed the crawling efficiency of each distributed worker is the same. Therefore, theoretically a distributed crawler, consisting of 10 workers, will spend $10t$ for visiting a page, assume a single crawler uses time t to visit a page. But we can see that the efficiency drops rapidly as the number of workers increases. When the number of workers reaches 100, the crawling efficiency of single worker is less than 50% than it works alone.

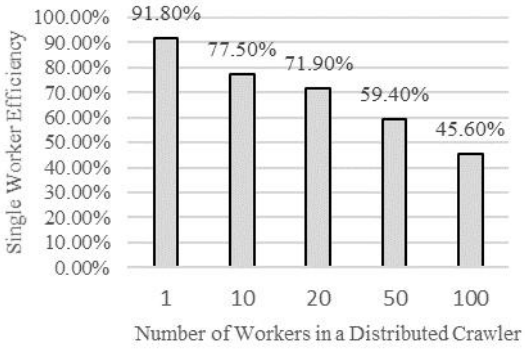

Fig. 3. PathMarker Suppressing Distributed Crawlers

In the same year, Cabri et al. [14] proposed a novel approach to binary classification of a multivariate data stream incoming on a web server, in order to recognize ongoing user sessions, generated by robots or humans. Deep neural networks and Wald's Sequential Probability Ratio Test were utilized to represent the relation between subsequent HTTP requests in an ongoing session and further to assess the possibility of each session being generated by a robot or human visitors. Experimental results showed the proposed approach could detect robots with a high accuracy and efficiency, and had slight impact on human visitors. From Fig. 4, we can see that the developed system could achieve stable Precision (0.979), Accuracy (0.963), F1 (0.959) and Recall (0.940) after the number of requests reached 8.

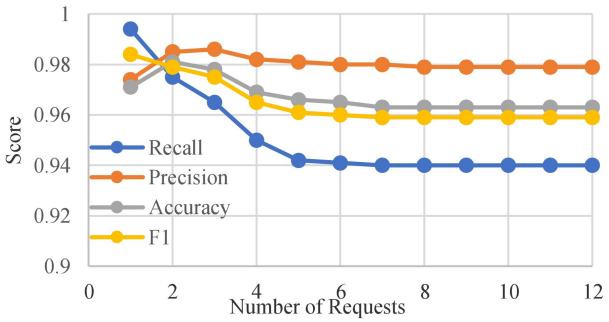

Fig. 4. Evaluation scores of the Sequential Classification Approach [14]

V. THE IDENTIFIED GAP AND CONCLUSIONS

To get insight into the various methods or techniques for web robot detection over the period of 2014-2019, a literature review was conducted. The goal was to appraise what kind of methods or techniques have been utilized for web robot detection, what strategies can be used to achieve high performance and efficiency for web robot identification, and what kinds of features are available for crawler detection.

We categorized the web robot detection methods to three types: offline web-log mining with machine learning techniques, honeypot, and online web robot detection. As shown by the results of the literature review, the existing studies about crawler detection mainly focus on off-line web log mining, which identify web crawlers by extracting features from sessions, and building up binary classifiers, through training with the extracted features. Only few of studies managed to identify web robots online, and even less

studies investigated the detection of distributed web robots. Experimental results show that offline analysis of web logs using machine-learning methods for web robot detection achieved considerable accuracy in some scenarios. Honeypot plays an important role for collecting samples from website, as it could collect invisible contents. However, one setback is that legitimate robots can be fooled by honeypot too. Therefore, honeypots could be a data collector rather than a classifier.

From the review, it can be seen that not only the detection techniques or methods are important, but also the features, being used for training the machine learning models, are important. Assessment approach is important as well. Different performance indicators could be used. It should be noticed that the online detection performance should not affect the performance of web browser, where users navigate.

In the latest research, there is a new solution of online web robot detection by inspecting the relationship between web pages or requests, and it obtained quite good performance.

For future study, it is demanded to design and develop high performance and high efficient online web techniques or methods for detecting web robots and/or distributed web crawlers. This is required by "Security by Design" for Industry 4.0.

ACKNOWLEDGMENT

This research is partially funded by research scholarship in the Department of Manufacturing, Cranfield University.

REFERENCES

[1] T. V. Udapure, R. D. Kale, and R. C. Dharmik, "Study of Web Crawler and its Different Types," *IOSR J. Comput. Eng.*, vol. 16, no. 1, pp. 01–05, 2014.

[2] N. Algiriyage, "Offline analysis of web logs to identify offensive web crawlers," International Research Symposium on Pure and Applied Sciences (IRSPAS 2017), Kelaniya, Sri Lanka, 20 Oct. 2017.

[3] M. Schmidt, "Web crawler," MSc thesis, Governors State University, 2015.

[4] M. Zabihimayvan, R. Sadeghi, H. N. Rude, and D. Doran, "A soft computing approach for benign and malicious web robot detection," *Expert Syst. Appl.*, vol. 87, pp. 129–140, 2017.

[5] M. Catalin and A. Cristian, "An efficient method in pre-processing phase of mining suspicious web crawlers," *2017 21st Int. Conf. Syst. Theory, Control Comput* (*ICSTCC*), Sinaia, Romania, 19-21 Oct. 2017, pp. 272–277.

[6] J. Rajabnia and M. V. Jahan, "Web robot detection with fuzzy inference system based on NNGE," available on https://www.academia.edu/8753500/Web_Robot_Detection_With_Fuzzy_Inference_System_Based_On_NNGE, accessed on 20/01/2020.

[7] G. Suchacka and M. Sobkow, "Detection of Internet robots using a Bayesian approach," *Proc. - 2015 IEEE 2nd Int. Conf. Cybern. (CYBCONF), Gdynia, Poland,* , 24-26 Jun. 2015. pp. 365–370, 2015. DOI: 10.1109/CYBConf.2015.7175961

[8] D. S. Sisodia, S. Verma, and O. P. Vyas, "Agglomerative Approach for Identification and Elimination of Web Robots from Web Server Logs to Extract Knowledge about Actual Visitors," *J. Data Anal. Inf. Process.*, vol. 03, no. 01, pp. 1–10, 2015.

[9] R. Haidar and S. Elbassuoni, "Website navigation behavior analysis for bot detection," *Proc. - 2017 Int. Conf. Data Sci. Adv. Anal. DSAA 2017*, Tokyo, Japan, 19-21 Oct 2017, pp. 60–68, DOI: 10.1109/DSAA.2017.13 .

[10] A. Lagopoulos, G. Tsoumakas, and G. Papadopoulos, "Web robot detection: A semantic approach," *Proc. - Int. Conf. Tools with Artif. Intell. ICTAI*, Volos, Greece, 5-7 Nov. 2018 pp. 968–974. DoI: 10.1109/ICTAI.2018.00150.

[11] S. F. McKenna, "Detection and classification of web robots with honeypots," MSc thesis, Naval Postgraduate School, 2016.

[12] T. Gržinić, L. Mršić, and J. Šaban, "Lino - An Intelligent System for Detecting Malicious Web-Robots," in *Nguyen N., Trawiński B., Kosala R. (eds) Intelligent Information and Database Systems. ACIIDS 2015. Lecture Notes in Computer Science, vol 9012. Springer, Cham*, pp. 559–568.

[13] Priyanka, V. Patankar, and S. Jangale, "Malicious Web Crawler Detection using Intrusion Detection System," *IJMTER*, vol. 3, no. 3(04-2016), pp. 364–371, 2016.

[14] A. Cabri, G. Suchacka, S. Rovetta, and F. Masulli, "Online Web Bot Detection Using a Sequential Classification Approach," Proc. – IEEE 20th Int. Conf. High Perform. Comput. Commun. IEEE 16th Int. Conf. Smart City, IEEE 4th Int. Conference Data Science and Systems (HPCC/SmartCity/DSS),, Exeter, UK. 28-30 Jun 2018. pp. 1536–1540.

[15] S. Wan, Y. Li, and K. Sun, "PathMarker: protecting web contents against inside crawlers," *Cybersecurity*, vol. 2, no. 1, pp. 1–17, 2019.

[16] S. Wan, Y. Li, and K. Sun, "Protecting web contents against persistent distributed crawlers," *IEEE Int. Conf. Commun.*, Paris, France, 21-25 May 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7996685.