Evolution of Web Bots and How They Are Detected

Robert J. Beane
beane031@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

Abstract

Web bots have become a major part of the internet over the past couple years. Though some have good intent, most bots are created with bad intentions. Because of these "bad" bots, detection techniques are needed to detect and deny access to those bots. However, some of these bad bots use advanced technology which makes them harder to detect than "simple" bots. This paper discusses the history of web bots and the impact they have on the internet. It also explores proposed frameworks for detecting advanced web bots efficiently using different techniques including web logs and mouse behavior. Both show promise in being effective tools for detecting web bots with mouse behavior being found to be extremely helpful.

Keywords: Web Bots, Neural Networks, CAPTCHA, Detection, Mouse Behavior, Web Logs

1 Introduction

Over the years, web bots have become somewhat of a necessity for the web to function properly. For example, bots can be used for indexing materials on the web, extracting specific data from sites for various means, and many more tasks. In 2019, web bots used 37.9% of all internet traffic [3]. The complexity of bot behavior varies, and in this paper, we categorize bots as being "simple" or "advanced." "Simple" bots are used for more mundane tasks like simply downloading data from the web. However, "advanced" bots actively aim to evade detection, which can be more problematic. In a 2021 study by Imperva (a cybersecurity company that sells security solutions), Imperva determined 16.7% of bots were defined as sophisticated, 40.4% as moderate, and 42.9% as simple. Combined, moderate and sophisticated account for 57.1% of bot traffic which shows that the majority of bots exhibit behavior that makes them challenging to detect [5]. In this paper, I begin with background in Section 2, explaining problems bots can/have caused, CAPTCHAs, and how bots have evolved. After that, Section 3 discusses a detection framework that uses web logs to detect bots. Section 4 builds on the previous framework with the addition of mouse behavior logs being used to detect bots. In Section 5, I discuss an "on-the-fly" detection framework for detecting bots. Section 6 briefly discusses the implications bots have on social media and how they can influence what we see. Finally, Section 7 brings forth the conclusion.

2 Background

2.1 Problems bots cause

Some web bots can be, and have been, used for malicious purposes. Some common malicious uses for web bots include vulnerability scanning, carding, spamming, and denial of service attacks (commonly referred to as DOS attacks) [4]. Bots can also influence certain markets via the use of scalping bots that buy an entire stock of something that is sought after (for example, graphics cards, game consoles, etc.) and re-sell those items for well above the Manufacturer Suggested Retail Price. All of these problems affect human users, making their internet experience more frustrating than it needs to be.

2.2 CAPTCHAs

One of the most common approaches to tackle web bots in recent times are Completely Automated Public Turing test to tell Computers and Humans Apart, more commonly referred to as CAPTCHA. CAPTCHAs come in many forms; probably the most well known is Google's version, which they call reCAPTCHA. Some examples include selecting images of buses or other objects or typing out obscured strings, but they all lead to the same end goal. Though effective, it is undeniable that CAPTCHAs interrupt and cause somewhat of an annoyance for the user. Some also lack certain accessibility options making it harder for some people to complete them. In addition, there have been some cases of speech-to-text generators being used to bypass the tests, therefore letting some bots through [4].

2.3 How have bots evolved?

With new bot detection techniques, bot developers have retaliated with more advanced web bots that have specifically been developed to avoid detection. These new advanced bots avoid detection by imitating human behavior [4]. Some have been found to use speech to text to solve CAPTCHAs that display randomized text and some use simulated mouse movement to make it look like a human is using a mouse. Without imitating mouse behavior, mouse movements from a bot would look like point-to-point movements or none at all if someone is using a script.

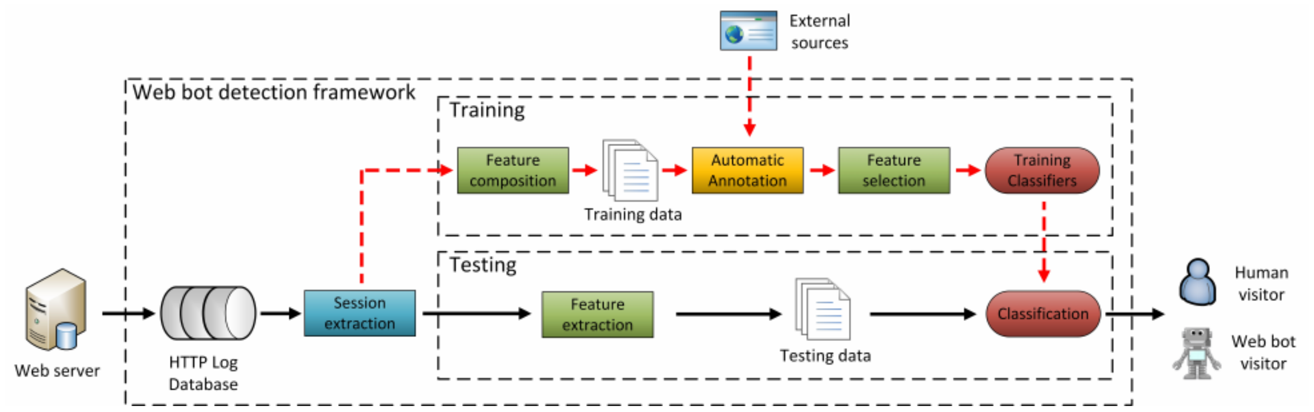

Figure 1. Proposed framework by C. Iliou et al. [4]

3 Utilizing Web Logs to Detect Bots

3.1 Machine Learning Framework

The machine learning framework proposed by C. Iliou et al. for detecting bots via web logs (see Figure 1) starts off by using regular expressions to comb through a given HTTP log database. Using a regular expression allows the researchers to choose what data they want to use [4]. When extracting the web logs, there are four steps the framework follows: Session extraction, feature extraction, feature selection, and classification. During the session extraction step, the framework takes the session's IP and browser agent name to create a session id that is then stored. This is important because it helps distinguish sessions that occur on the same IP. Feature extraction then takes the data from the session and converts it into more measurable values that are then used for the training and testing portion of the framework [4]. After testing features with the training data, the authors then selected the best performing features to the framework. The final step, classification, classifies the sessions as human or bot. This is done by using a classification algorithm. The authors used four of methods; Support Vector Machine (SVM), Random Forest, Adaboost, and Multi Layer Perceptron (MLP). Though the authors used these four, they say that the "framework is built to allow for the effortless incorporation of any machine learning algorithm" [4].

The framework takes the browser/user agent and the IP address of the session and compares them to external servers that contain known web bots. A browser/user agent is data that is sent to the server you are trying to access letting the server know what type of system you are on and what browser you are using to access the server; some bots lack this element. The framework then automatically annotates each session as one of the following: human, simple bot, or advanced bot. Figure 2 shows the process by which the sessions are categorized. This framework uses supervised learning instead of unsupervised learning. Supervised learning is a machine learning method that has a dataset that

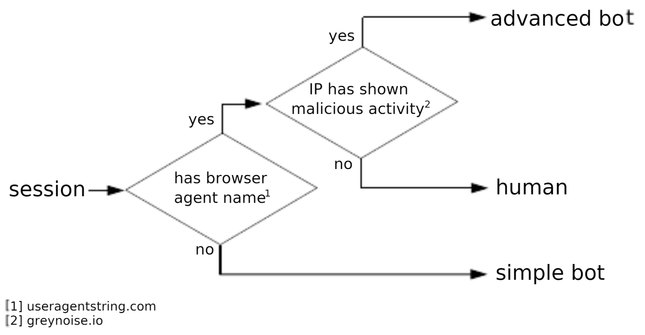

Figure 2. Automatic Annotation Process (decision tree) by C. Iliou et al. [4]

is labeled with correct outcomes whereas an unsupervised method has an unlabeled dataset. The dataset is split into two parts: training and testing. About 61% of data is used for training and the remaining 39% is used for testing. It is important to have more data used for training to avoid over-fitting. Over-fitting is a problem commonly faced in machine learning and occurs when the model is really good at classifying data that was in the training set. However, when it comes to classifying data that was not in the training set, it does not do so well. The final step of the framework is classification which uses the training phase which is supplied with known bot and human sessions. The training phase takes the chosen features from the session (feature composition). After that, it then annotates the session (automatic annotation). The framework then decides the most important features and selects them (feature selection). The training phase then concludes by using the classification algorithms. For the testing phase, the features that were chosen are extracted from the incoming session. This testing data is used to classify the session. During the authors testing, these sessions were labeled as human or bot by the researchers but the labels remained a secret to the framework so that they could evaluate the performance of the framework [4].

3.2 Framework Results

Through testing, the researchers discovered that the "web bot detection problem is a multifaceted one, characterised by the coexistence of simple bots that can be detected easily and advanced web bots that are considerably more difficult to detect" [4]. However, it was found that, "if the framework is applied on a false-positive intolerant Web server, its effectiveness regarding detecting advanced Web bots is significantly reduced" [4]. Besides false-positive intolerant servers, this framework was an effective way to detect web bots.

4 Utilizing Mouse Behavior to Detect Bots

More advanced web bots are able to mimic human mouse behaviors which in turn makes it easier for them to avoid certain levels of detection. For example, if a user is seen to be just going from one input to another in the most efficient way, that's probably a sign that the "user" is indeed a bot. However, when you add some variance (because human mouse behavior is not perfect) to the trajectory of that mouse, all of a sudden that "user" looks like a real person instead of a bot.

4.1 SapiAgent: Generating Mouse Behaviors

One such method for generating human-like mouse behaviors comes from a deep-learning framework called SapiAgent, developed by M. Antal et al. [1]. The main goal of SapiAgent is to be comparable to mouse movements performed by real humans so that the generated mouse movements are not detected by state-of-the-art detection software. SapiAgent is able to create its mouse movements with a dataset that contains mouse usage data from 120 subjects. The subject pool was quite diverse in terms of gender, age, and handedness [1]. The data was collected by having the subjects play a simple game where their mouse movements were sampled at a frequency of 60Hz. Different actions were performed such as left and right clicks, double clicks, and drag and drop all being included in the samples. This data is then graphed and examined.

The authors used this gathered data to create two different Bézier curves, quadratic and cubic. Bézier curves are curves which have starting and ending points that are, "always bounded by the convex hull of its control points" [1]. In this case, the control points were the start and end points of the user's mouse movements and they also made sure the length of the curve was the same in regards to the time of the mouse movement. The authors decided to use the cubic Bézier curves over the quadratic ones because they found the quadratic ones to be more of a "baseline" whereas the cubic curves were more "Human Like" [1]. This data is then plugged into an autoencoder. An autoencoder's purpose is to try to find the best way to represent a certain dataset in the lowest dimension possible while also staying close to the original dataset. The autoencoder the authors used can

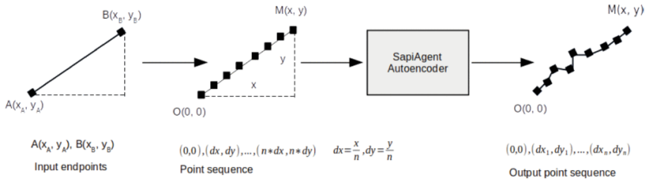
B(x_B, y_B)
A(x_A, y_A)
M(x, y)
y
x
O(0, 0)
SapiAgent Autoencoder
M(x, y)
O(0, 0)
A(x_A, y_A), B(x_B, y_B)
Input endpoints
$(0,0),(dx,dy),\dots,(n*dx,n*dy)$
Point sequence
$dx=\frac{x}{n}, dy=\frac{y}{n}$
$(0,0),(dx_1,dy_1),\dots,(dx_n,dy_n)$
Output point sequence

Figure 3. Autoencoder for generating human-like trajectory from equidistant points. [1]

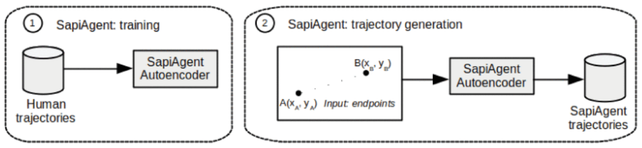

Figure 4. 1. Training the bot agent. 2. Generating mouse trajectories. [1]

be seen in Figure 3 where it initially takes in the endpoints and generates a "human like" mouse trajectory. The overall approach of SapiAgent is outlined in Figure 4 which also showcases where the autoencoder is used.

In the end, the authors of SapiAgent concluded that their, "experimental results show that SapiAgent trained in a novel way generates more realistic mouse trajectories compared with conventional autoencoders and Bézier curves" [1]. They also believe that SapiAgent could be used to generate more data similar to mouse trajectories.

4.2 Detecting Advanced Bots with Mouse Behavior

Because these bots are more "human-like," they are harder to detect than more simple web bots. Christos Iliou, et al. have proposed a framework that utilizes web logs and mouse behavior to detect such advanced bots [3]. Currently, the most used approach to detect advanced bots is to use machine learning while focusing on classification and/or clustering algorithms [3]. The framework proposed by Christos Iliou, et al. uses two separate detection modules. One module detects bots based on web logs and the other detects bots based on mouse movements. The framework can be seen in Figure 5.

The module that uses web logs is the framework discussed in Section 3. While the user is using the site, the second module is collecting mouse data and logging it. This data includes the x and y coordinates of the mouse and the time t the action took place. The data is collected in such a form $\{(x_1, y_1, t_1), (x_2, y_2, t_2), ..., (x_n, y_n, t_n)\}$. All of this mouse collection is done with a JavaScript file that is embedded in the page [3]. Finally, the prior steps are combined using a fusion method. This fusion method takes the scores from both modules ($score_{mv}$ for mouse movements and $score_{wl}$ for web logs) and combines them to get a total score ($score_{tot}$). However, if $score_{mv}$ is either very high or very low, Iliou et al. determined that just using the score from mouse movements

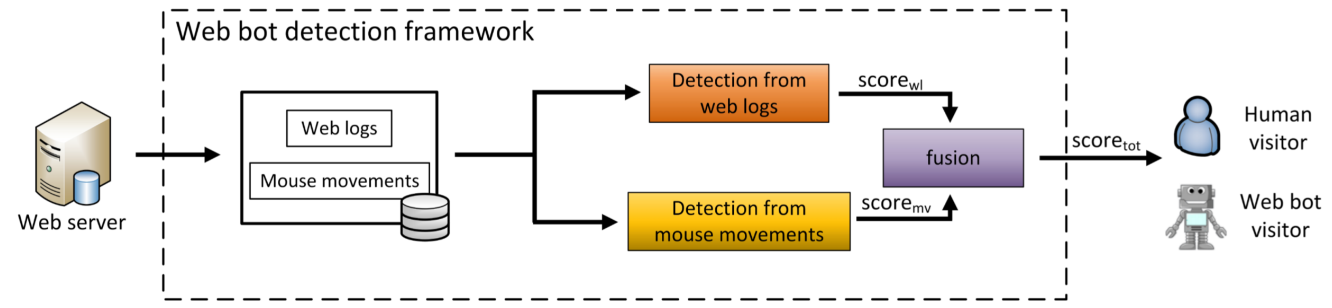

Figure 5. Christos Iliou, et al proposed this framework for categorizing users as being a bot or a human [3]

was enough to determine a session as human or bot [3]. Figure 6 displays the process in which the mouse behavior data is collected.

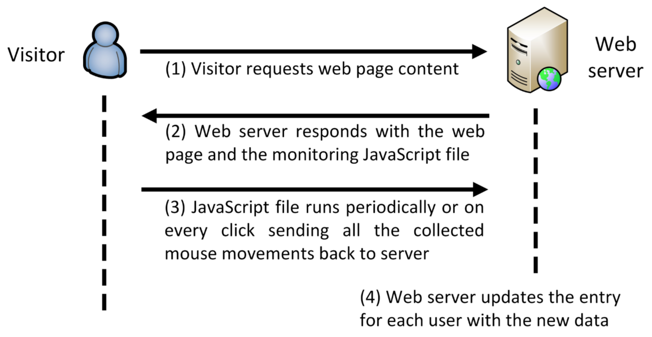
Visitor
Web server
(1) Visitor requests web page content
(2) Web server responds with the web page and the monitoring JavaScript file
(3) JavaScript file runs periodically or on every click sending all the collected mouse movements back to server
(4) Web server updates the entry for each user with the new data

Figure 6. Mouse data collection process [3]

In C. Iliou et al. final findings, they discovered that the mouse behavior module was more effective at detecting bots in general than the module that used web logs [3]. They also determined that, "simulating humanlike mouse movements is more difficult than simulating humanlike browsing behaviour and simulating both simultaneously requires a higher complexity than the ones tested in this work" [3]. Finally, C. Iliou et al. concluded that their, "proposed approach is suitable for the online detection of web bots, as it achieves high effectiveness with very few requests" [3]. They found that some sessions were identified in as few as two to three requests.

4.3 Is Mouse data helpful?

From the work of C. Iliou et al., it does seem that the usage of mouse behavioral data can be an effective tool to detect bots compared to other methods. They found it more useful than using web logs for detecting bots. However, bots like SapiAgent have been able to create realistic mouse behavior [1]. Because bots like SapiAgent have shown that it is possible to simulate realistic human mouse movements, being able to detect bots by mouse movement might not be as effective in the future as it is now.

5 On-the-fly detection

Being able to identify web bots in real time brings forth many advantages, one being the ability to prevent threats sooner. One of the most common current "on-the-fly" techniques for detecting bots is a CAPTCHA. Though CAPTCHAs are useful, they add a layer of annoyance for the regular user. G. Suchacka et al. have acknowledged this fact and have proposed a framework that uses machine learning to detect ongoing visits as human or bot [6]. Their framework is able to classify a user as soon as it receives enough information from the user's session.

5.1 Proposed Framework

The framework proposed by G. Suchacka et al. can be seen in Figure 7. The back end portion of the framework is triggered periodically and handles the processing of historical HTTP data, training the neural network, and monitoring and storing the performance scores which uses the recall ("fraction of positive sessions that are correctly classified") [6], precision ("fraction of positive decisions that are correct" [6]), and accuracy ("fraction of correct classifications" [6]) of the network. If it detects a decrease in performance, the network will re-train the neural network estimator [6]. The front end portion of the framework handles the real time activities of whenever there is a new incoming request. Similarly to the previously mentioned framework that used web logs, this framework also labels each session as human or bot by using third party sites by using session user agent strings and IP addresses. Each request is given a value that is used to estimate if the session is 0 (human), 1 (bot), or None (same as "undecided"). Sessions that end too quickly to decide are given an "undecided" classifier; in this case, G. Suchacka et al. suggest the idea of presenting a CAPTCHA to that user the next time they connect [6]. The framework uses ten different features for recognizing the sessions. Seven features come from the HTTP request headers, two come from the HTTP response headers, and the last feature comes from the HTTP server's timestamp to log the time at which the session begins [6]. These ten features can be categorized in

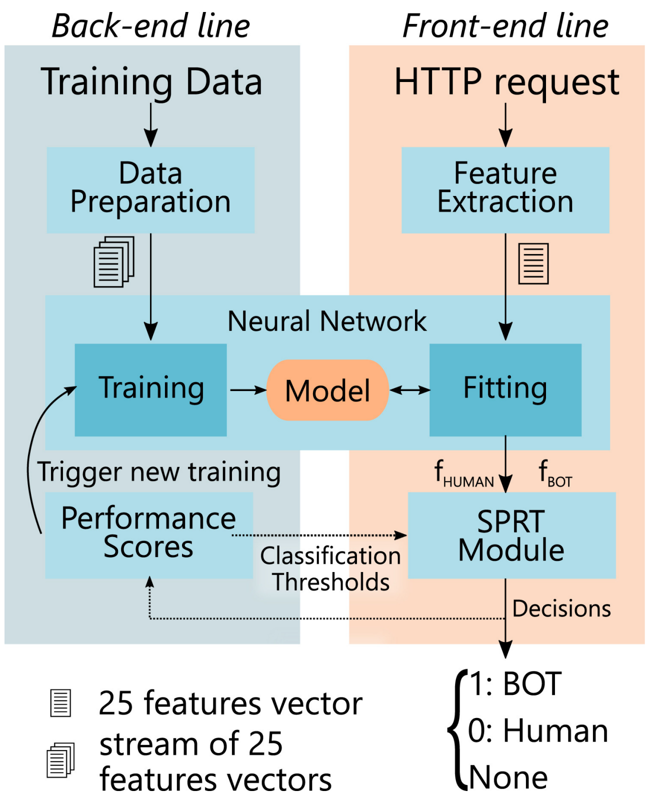

Figure 7. G. Suchacka et al.'s proposed detection framework [6]

three types of data: numerical features, categorical features, and boolean features.

5.2 On-The-Fly Detection Results

Through their testing, G. Suchacka et al. discovered that their framework, "is able to determine a decision for nearly all active sessions, leaving only 0.69% of them undecided" [6]. They go on to claim that about 99% of the sessions tested with their framework were able to be classified within only six requests. Not only that, the max number of requests throughout the whole dataset was 19. The accuracy of the framework was also pretty impressive, they claim that 96% of sessions where classified correctly. Furthermore, G. Suchacka et al. claim that, "the classifier can be easily implemented as a simple extension to a real Web server software and integrated with other fallback bot detection mechanisms, like a CAPTCHA test" [6]. This should allow easy implementation for anyone that wants to use it.

6 Bot Impact on Social Media

Social networks have become a major part of many people's lives over the past decade. Social media can be used to connect with friends over long distances as well as being able to seeing up to date news from all over the globe. Social networks also carry a huge influence on their users which, if controlled, could push certain agendas. According to a study done in 2017 by O. Varol et al., it was estimated that about 15% of accounts on Twitter were bots [7]. It is likely that over the last five years that that number has only increased due to major political events.

One thing Twitter bots have been found to do is increase certain URL traffic according to Z. Gilani et al. [2]. Z. Gilani et al.'s Twitter bot took popular "job" related tweets that included URLs. The bot then took the URLs and shortened them by having them go through their own web server so they could log the data from users that clicked the link. After roughly two years of data logging, they discovered that more than 44% of clicks were performed by bots; this can be seen in Figure 8. They also discovered that roughly 4% of the users were recurring bots [2].

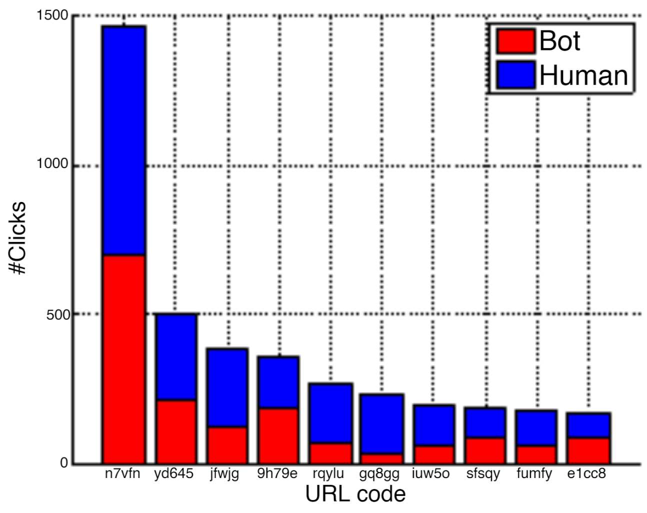

Figure 8. Number of Clicks per URL over two years from Z. Gilani et al. [2]

6.1 Social Media Companies' Response

Social media companies like Twitter have recently implemented a label that is displayed on tweets that originate from an automated source. This was added to inform users if a tweet is from a real human or from a bot. However, this label is not given to bots pretending to be humans; the label is reserved for accounts that utilize the Twitter API to post tweets.

7 Conclusion

The issue with advanced bots will only increase, which is why frameworks like the ones mentioned in this paper are important. Each discussed framework has displayed its effectiveness at detecting advanced web bots from varying techniques. The use of web logs has shown to be an effective technique for detecting bots as well as the more complicated technique of using session mouse behavior. Some of the authors of some of these frameworks have stated that they have future plans and improvements for their proposed frameworks to implement new features and to improve speed and efficiency of the framework. Within the near future, hopefully we can see some of these frameworks being implemented into live servers (instead of test servers) providing effective bot detection that does not rely on the interruptive common CAPTCHA.

Acknowledgments

Special thanks to Kristin Lamberty and Elena Machkasova for their feedback throughout this process. Thank you to Melissa Helgeson for taking the time to read and review my paper.

References

[1] Margit Antal, Krisztian Buza, and Norbert Fejer. 2021. SapiAgent: A Bot Based on Deep Learning to Generate Human-Like Mouse Trajectories. *IEEE Access* 9 (2021), 124396–124408. https://doi.org/10.1109/ACCESS.2021.3111098

[2] Zafar Gilani, Reza Farahbakhsh, and Jon Crowcroft. 2017. Do Bots Impact Twitter Activity?. In *Proceedings of the 26th International Conference on World Wide Web Companion* (Perth, Australia) *(WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 781–782. https://doi.org/10.1145/3041021.3054255

[3] Christos Iliou, Theodoros Kostoulas, Theodora Tsikrika, Vasilis Katos, Stefanos Vrochidis, and Ioannis Kompatsiaris. 2021. Detection of Advanced Web Bots by Combining Web Logs with Mouse Behavioural Biometrics. *Digital Threats: Research and Practice* 2, 3, Article 24 (jun 2021), 26 pages. https://doi.org/10.1145/3447815

[4] Christos Iliou, Theodoros Kostoulas, Theodora Tsikrika, Vasilis Katos, Stefanos Vrochidis, and Yiannis Kompatsiaris. 2019. Towards a Framework for Detecting Advanced Web Bots. In *Proceedings of the 14th International Conference on Availability, Reliability and Security* (Canterbury, CA, United Kingdom) *(ARES '19)*. Association for Computing Machinery, New York, NY, USA, Article 18, 10 pages. https://doi.org/10.1145/3339252.3339267

[5] Imperva. 2021. Bad Bot Report 2021: The Pandemic of the Internet. https://www.imperva.com/resources/resource-library/reports/bad-bot-report/

[6] Grażyna Suchacka, Alberto Cabri, Stefano Rovetta, and Francesco Masulli. 2021. Efficient on-the-fly Web bot detection. *Knowledge-Based Systems* 223 (2021), 107074. https://doi.org/10.1016/j.knosys.2021.107074

[7] Onur Varol, Emilio Ferrara, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2017. Online Human-Bot Interactions: Detection, Estimation, and Characterization. https://aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15587/14817