Bot recognition in a Web store: An approach based on unsupervised learning

Stefano Rovetta, Grażyna Suchacka, Francesco Masulli

PII: S1084-8045(20)30051-5

DOI: https://doi.org/10.1016/j.jnca.2020.102577

Reference: YJNCA 102577

To appear in: *Journal of Network and Computer Applications*

Received Date: 26 March 2019

Revised Date: 31 October 2019

Accepted Date: 10 February 2020

Please cite this article as: Rovetta, S., Suchacka, Graż., Masulli, F., Bot recognition in a Web store: An approach based on unsupervised learning, *Journal of Network and Computer Applications* (2020), doi: https://doi.org/10.1016/j.jnca.2020.102577.

Bot recognition in a Web store: an approach based on unsupervised learning

Stefano Rovetta[a], Grażyna Suchacka[b,*], Francesco Masulli[a]

[a] *Department of Informatics, Bioengineering, Robotics and Systems Engineering, University of Genoa, Italy*
[b] *Institute of Informatics, University of Opole, Opole, Poland*

Abstract

Web traffic on e-business sites is increasingly dominated by artificial agents (Web bots) which pose a threat to the website security, privacy, and performance. To develop efficient bot detection methods and discover reliable e-customer behavioural patterns, the accurate separation of traffic generated by legitimate users and Web bots is necessary. This paper proposes a machine learning solution to the problem of bot and human session classification, with a specific application to e-commerce. The approach studied in this work explores the use of unsupervised learning (k-means and Graded Possibilistic c-Means), followed by supervised labelling of clusters, a generative learning strategy that decouples modelling the data from labelling them. Its efficiency is evaluated through experiments on real e-commerce data, in realistic conditions, and compared to that of supervised learning classifiers (a multi-layer perceptron neural network and a support vector machine). Results demonstrate that the classification based on unsupervised learning is very efficient, achieving a similar performance level as the fully supervised classification. This is an experimental indication that the bot recognition problem can be successfully dealt with using method that are less sensitive to mislabelled data or missing labels. A very small fraction of sessions remain misclassified in both cases, so an in-depth analysis of misclassified samples was also performed. This analysis exposed the superiority of the proposed approach which was able to correctly recognize more bots, in fact, and identified more camouflaged agents, that had been erroneously labelled as humans.

Keywords: Web bot, Internet robot, Web bot detection, supervised classification, unsupervised classification, machine learning, Web server

1. Introduction

The work presented in this paper addresses the issue of recognizing artificial agents or *bots* from human visitors to a Web shop under realistic conditions. It includes a study of machine learning techniques for classification that can be employed even with missing labels, being built around unsupervised analysis, that compare favourably with fully supervised classifiers, as well as an analysis of the role of features and the type of errors incurred in by the best-performing classifiers.

This introduction describes the context, the motivation, and the approach proposed, highlighting the scientific contribution of this research.

1.1. Context

Contemporary Web services offer huge possibilities of running any kind of business on a global scale. Furthermore, social media, online marketing and Web analytics technologies make it possible to attract prospective e-customers, provide them with personalized service and systematically reinforce their loyalty to the brand. The efficient functioning of the electronic marketplace is largely possible due to Web bots, which explore the Web on a regular basis and automate the execution of many tedious, recurrent, and routine tasks.

A Web bot, also called a Web crawler, Internet robot or intelligent agent, is a software tool that performs specific actions on computers connected in a network without the intervention of human users, by following hyperlinks. Search engine indexers, monitoring bots, link checkers, feed fetchers are examples of "good bots" – they usually have legitimate goals and comply with directives placed by website mainteners in the *robots.txt* file to prevent or limit access to specific page subsets.

Along with collaborative agents, however, "bad bots" have been increasingly used in recent years. Such bots tend to adopt impersonation tactics, typically by changing the *user agent* field in HTTP/HTTPS headers of their requests to masquerade as either human users (by employing the user agent strings of human-operated Web browsers) or "benign" bots, like Google or Baidu crawlers (Bai et al., 2014). Moreover, advanced robots can operate at the application layer, being able to imitate the way in which legitimate users interact with online applications via their browsers, which makes them hard to detect. Such bots are often used to gain undue advantage in online business (see Subsection 2.1).

According to recent bot traffic reports (GlobalDots, 2018; Zeifman, 2017), about half of website visitors (42.2% – 51.8%) are actually robots, and as much as 51.7% – 55.8% of all robots are malicious ones. Among different

*Corresponding author
Email addresses: stefano.rovetta@unige.it (Stefano Rovetta), gsuchacka@uni.opole.pl (Grażyna Suchacka), francesco.masulli@unige.it (Francesco Masulli)

industries, e-commerce is the fifth in the ranking as regards bad bot traffic intensity and the first in terms of volumes of sophisticated bot traffic (generated with the use of browser automation software or malware installed within real browsers, thus perfectly imitating legitimate users) (GlobalDots, 2018).

The ever increasing proliferation and sophistication of bots, followed by real damages suffered by online companies, have been driving a lot of research on bot traffic analysis and detection. It has to be noted, however, that irrespective of malicious robot activities, all bot traffic on e-business websites should be identified. Despite the fact that all kinds of robots introduce additional server load, which should be controlled and limited in some cases, there is a strong need for separation of all bot and human visits on HTTP level.

1.2. Motivation and proposed approach

The main motivation for our study was the need for reliable identification of automatically generated visits in online stores. This problem has two main facets. First, the ability to identify HTTP bot traffic allows a website administrator to obtain accurate measurements of actual site popularity and other, business-related metrics. Second, this ability is fundamental for reliable and solid e-customer behaviour characterisation and pattern discovery. In practice, conclusions drawn from such analyses are greatly beneficial for optimization of the website design, implementation of more effective product recommendation methods, development of more adequate marketing strategies, etc. Since intelligent agents reveal different online navigational patterns than humans (Calzarossa and Massari, 2011; Doran et al., 2013; Suchacka, 2014), incapability of bot detection and elimination may skew outcomes of customer behavioural studies, leading to improper business decisions. Finally, since sophisticated software agents are able to efficiently mimic navigation of human visitors, this may incur real negative consequences for e-business, like price undercutting, click frauds, or credit card frauds, just to name a few. (A wider discussion on detrimental bot impact on e-business profitability is included in Section 2.1.)

The objective of our research is to explore a machine learning (ML) solution to the problem of bot and human session classification, with a specific application to e-commerce and under a realistic scenario. The main research question is whether it is possible to achieve good recognition rates in the task of distinguishing between sessions of legitimate, human users and Web robots using computational intelligence techniques rather than hand-engineered filtering criteria.

One issue that is commonly faced in real-world applications is the lack, or limited availability, of labelled data. Labelling is almost invariably an expensive task. Additionally, in the specific case of interest there is no solid criterion for labelling all possible bots, so, even among available labels, a fraction may contain unreliable information.

To gain insight into the problem and propose a realistic solution, this work explores the hypothesis that a structural characterization of bots may be possible. If this hypothesis is verified, a learning machine may be able to discriminate bots from regular traffic even without the supervision of class labels.

In this case, we can make the further hypothesis that clusters corresponding to legitimate users will be stable over time, while clusters corresponding to bots will change as a consequence of the evolution of bots themselves. This will allow future works to use the methods discussed here (Abdullatif et al., 2017), as well as possible variants in the same spirit (Abdullatif et al., 2018), in a change detection or novelty detection setting, to track the evolution of bots and keep the ability to identify them possibly even *before* novel types of malicious bot behaviour are identified.

1.3. Contribution

The major contributions of this paper are the following:

1. We explore an approach to session classification on a Web server, based on unsupervised learning (clustering) followed by supervised labelling of clusters ("nearest centroid" approach). This method can be described as a generative approach, and can be applied even when labels are present only for a subset of the data, making it viable for semi-supervised learning. Two clustering algorithms are applied: k-means and Graded Possibilistic c-Means (GPCM). To the best of our knowledge, this is the first study that applies the GPCM algorithm in the field of Web bot detection and one of the first works addressing the problem of bot and human session classification based on unsupervised learning. The efficiency of the approach is assessed by comparing it to that of supervised classification with the use of two methods: a multi-layer perceptron (MLP) neural network and a support vector machine (SVM).
2. The approach is dedicated to dynamic, e-commerce websites, implemented in a multi-tiered architecture (consisting of Web server, application server, and database server layers). We propose to describe sessions not only with common features, representing statistical session characteristics, but also with novel semantic features, related to the process of purchasing goods online.
3. To evaluate the classification performance of the proposed approach, we conduct experiments on a real e-commerce traffic dataset. To the best of our knowledge, all previous studies on bot detection with ML methods have been evaluated only for non e-business sites, mainly the ones from university domains. Such websites are usually characterized by a simpler structure, other functions, and other resource types. The corresponding studies for real dynamic, e-business

sites are missing and our study aims to partially fill this research gap concerning an economically significant area.

4. Experimental results show that the proposed clustering-based classifiers achieve a similar level of performance as the fully supervised ones, providing experimental evidence that the problem could be successfully addressed even in the presence of incomplete labelling.
5. An in-depth analysis of misclassified samples disclosed that the classifier based on unsupervised analysis recognized more robots and was able to correctly identify more camouflaged artificial agents, that were erroneously labelled as humans. Our methodology may be useful for more accurate session labelling for the use in classification models. It contributes to the area of reliable benchmarking for Web bot detection studies and developing novel bot detection methods.

1.4. Paper organization

The remainder of the paper is organized as follows. Section 2 presents the background on Web bot types and reviews related studies on methods for their detection offline. Section 3 presents basics on server log data and formulates the problem. The proposed research methodology is discussed in Section 4; Section 5 describes the experimental setup, followed by discussion of experimental results in Section 6. Section 7 concludes the paper.

2. Background and related work

2.1. Types of Web bots

Historically the first and primary bot application is Web crawling, aimed at indexing Web content on behalf of search engines. Nowadays a multitude of various types of supportive bots exist. Among these one can mention, e.g., link checkers detecting broken links, feed fetchers ferrying Web content to mobile apps, or shopping bots acting for product search engines or price comparers, which a given online retailer collaborates with. These types of robots usually inform a Web server about their identity via the bot name contained in the user agent field, so they may be easily recognized at the server and processed in a special way if needed. The only potential risk of their presence is the increase in network and server traffic, which though is usually kept under control by the bots themselves by limiting their own rate of activity.

Some concerns about privacy and ethics on the Web are raised by functioning of agents that collect sensitive data and reuse them, e.g., e-mail harvesters or resource archivers. Even more bothersome are activities of advanced application-level agents like social bots (Clark et al., 2016; Sadiq et al., 2017), blog bots (Chu et al., 2013), or spambots (Hayati et al., 2010; Kaur et al., 2018). A particularly severe problem for server administrators and site maintainers is camouflaged bad bots, whose behaviour can be really detrimental for target websites. A flagship example is Distributed Denial of Service (DDoS) attacks with the goal of blocking or gaining access to a particular website or service. A DDoS attack is launched by a botnet, created by many network-interconnected computers, each of which is infected with malware giving control of the infected machine to a central bot controller. Botnets can be organized in peer-to-peer or client-server structures. They can use several network protocols; however, due to the overwhelming diffusion of Web-based services, HTTP-based bots are the majority (Acarali et al., 2016). As opposed to traditional, network layer DDoS attacks, which are relatively easy detectable, HTTP-based application layer attacks are extremely hard to cope with (Adi et al., 2017; Behal and Kumar, 2017; Jazi et al., 2017; Singh et al., 2018).

Autonomous agents are also used to interfere with commercial activity on the Internet. For instance, click bots are a specific type of network programs specialized in simulating clicks on sensitive links, such as advertising banners (Haider et al., 2018; Walgampaya and Kantardzic, 2011). They are typically employed to perform so-called "click frauds" in online advertising, whose reach is measured by metrics like impressions (the number of times a given ad is visualized) and conversions (the number of times a visualized ad is actually clicked). A click fraud consists in falsifying the actual number of genuine impressions and conversions by inflating them with artificially-generated ones. The advertiser may be attacked by exhausting their visualization budget early in the day so that their ads do not appear any more after the number of displays per day is reached. Alternatively, sites that host the advertising can increase their own revenue by simulating a higher traffic.

Malicious bots can harm the e-commerce competitiveness and profitability in a number of ways. Scraping and duplicating the Web content may damage the website SEO (Search Engine Optimization) ranking due to duplicate content on the Web. Price harvesting enables undercutting prices offered by competitors. Creation or takeover of customer accounts may be used to spam the site comment sections, exploit account promotion credits (discounts, loyalty points, etc.), or hold out items in shopping carts without purchasing them, thus lowering the inventory availability for real customers. Bots on e-commerce sites are also employed to commit gift card frauds, credit card frauds, and purchase tickets in bulk for illegal resale. Negative long-term consequences for e-business include the undermined company's reputation and lower conversion rates.

In this paper we are interested in filtering all bots that are trying to access an e-shop. These bots can serve multiple different purposes, ranging from page indexing to complete transactions (some bots actually buy items).

2.2. Approaches to Web bot detection

A limited subset of bots may be identified by syntactical analysis of HTTP fields extracted from log entries, i.e. by

examining access to the *robots.txt* file in sessions, inspecting specific keywords in user agent strings or comparing IP addresses against a blacklist. This simple approach allows one to detect only well-known and cooperative robots, being blind for new or evolving ones. Due to these limitations, more sophisticated solutions to the problem of bot detection offline have been proposed, including traffic pattern analysis and analytical learning.

Traffic pattern analysis looks for known differences in interaction styles between bots and legitimate users (Guo et al., 2005; Lin et al., 2008). Analytical learning instead does not search for known patterns but use statistical or ML techniques to learn rules from navigational data and incorporate them into a formal probabilistic or ML model. Example probabilistic models include Bayesian approaches (Stassopoulou and Dikaiakos, 2009; Suchacka and Sobków, 2015), as well as Markov models based on request arrival patterns (Lu and Yu, 2006) and requested resource types (Doran and Gokhale, 2016; Suchacka and Motyka, 2018).

Most ML approaches to robot detection apply *supervised* learning. It consists in training a classifier, i.e. a function mapping an input (usually feature vectors describing sessions) to an output (session class labels) based on a training dataset, which includes *labeled* training samples. The ability of the inferred function to determine correct class labels for new, unseen samples is assessed on a test dataset. Many supervised learning techniques demonstrated their efficiency in classification of bots and humans, e.g., decision trees (Gržinić et al., 2015; Kwon et al., 2012; Tan and Kumar, 2002) support vector machine (Gržinić et al., 2015; Jacob et al., 2012; Rovetta et al., 2019), neural networks (Bomhardt et al., 2005; Rovetta et al., 2019), and k-Nearest Neighbours (Stevanovic et al., 2012; Saputra et al., 2013). All supervised learning approaches, however, share a common disadvantage, related to a difficulty with preparation of a reliable training dataset, in particular with assigning accurate class labels to sessions of camouflaged robots.

This drawback does not affect *unsupervised* learning, which is to learn intrinsic data properties from *unlabeled* training samples. In the field of Web bot traffic analysis there have been some works on session clustering to isolate bots. In Alam et al. (2014) Particle Swarm Optimization (PSO) was applied to distinguish robots among genuine Web users based on three session features: total transfer volume, number of pages, and session duration. The underlying assumption was that bots are outliers. However, as previously noted, the percentage of bot traffic has dramatically increased, therefore this assumption is not valid any more. As a consequence, only some kinds of bots might be detected with this method. For the experimental scenario in Alam et al. (2014), these are those bots requesting many pages and downloading large amounts of data in relatively short time periods.

In Zabihi et al. (2014) the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm was employed to separate bots and humans into two clusters. Fourteen session features were taken into account, four of which were finally selected for classification by using T-test. DBSCAN achieved very good results (the mean cluster purity was 0.97), however some sessions were left unclustered and treated as a noise, thus not being included in performance metrics. The authors concluded that some known bots imitating human behavior are very difficult to identify.

Two unsupervised neural network learning algorithms, the Self-Organizing Map (SOM) and Modified Adaptive Resonance Theory 2 (Modified ART2), were applied in (Stevanovic et al., 2013) with two goals: to obtain a better insight into the types and distribution of Web clients, and to investigate the relative differences and similarities between malicious bots and other non-malicious clients. Four client groups were considered: humans, well-behaved bots, malicious bots, and unknown. As a result of clustering, a pretty clear separation between robots and humans was achieved. However, out of all visitor groups, malicious bots exhibited the greatest variability – they were spread over all five generated clusters. A conclusion was that bots, in particular the malicious ones, display a range of browsing strategies; moreover, as much as 52% of malicious bots exhibit very "human-like" behavior.

In Hamidzadeh et al. (2018) and Zabihimayvan et al. (2017) session clustering was combined with feature selection based on Fuzzy Rough Set (FRS) theory. The first method, called FRS-WRD (FRS - Web Robot Detection) (Hamidzadeh et al., 2018), applied the SOM-based clustering. Its efficiency was evaluated in terms of the ability to separate bot and human sessions and turned out to be very high (the mean cluster purity was about 96%). The second method, called SMART (Soft computing for MAlicious RoboT detection) (Zabihimayvan et al., 2017), exploited a Markov clustering (MCL) algorithm to separate bots from humans. Clustering-based classification resulted in the mean accuracy of 0.92. A common conclusion from the both studies was that the most relevant attributes selected by FRS differ on a dataset so preceding the session classification with the feature selection stage can improve the classifier efficiency.

The analysis of literature shows that Web traffic reveals properties discriminating bots from humans to a large extent, though some sophisticated bots can actually impersonate legitimate users, thus remaining hard to detect. Very few studies classified sessions based on unsupervised learning; instead, most of approaches focused on investigating the ability to partition bots and humans into separate clusters and explored session properties depending on a cluster. Clustering-based models were used to classify new sessions only in Zabihimayvan et al. (2017), as well as in Rovetta et al. (2019) which discusses preliminary results of our approach. Furthermore, ML approaches to the problem of bot detection offline were evaluated for non e-business websites only and thus, the up-to-date literature lacks studies performed on real e-commerce datasets.

To address the aforementioned deficiencies, we develop

a novel method to classify bot and non-bot sessions in a Web store. The proposed classifier is based on unsupervised classification and subsequent labelling of generated clusters. Since some related works, e.g. (Stevanovic et al., 2013), reported a considerable diversification of bad bot navigation strategies, we examine a "microclustering" approach, similar to that used in DBSCAN, using a wide range of the number of centroids to be able to map possibly complex class boundaries using smaller convex components. Classification efficiency is evaluated on real e-commerce log data: the classifier is developed on a training dataset and its ability to generalize results to new observations is verified on a test dataset. Achieved performance metrics are compared with those obtained for the supervised learning classifiers. Moreover, features of individual sessions that were misclassified by unsupervised and/or supervised approaches are analysed thoroughly to inspect possible reasons for the residual errors.

3. Problem formulation

Web servers process traffic coming from Web clients according to HTTP protocol (Berners-Lee et al., 1996; Fielding et al., 1999; Belshe et al., 2015). Basic data on every HTTP request seen on a server is recorded in an access log, a standardized text file complying with a predefined format, set up in the server configuration. Common choices for e-business servers include the de facto standard *NCSA Common* log format or formats derived from it by adding more fields, like Apache's *NCSA Combined*.

The input for our approach is one or more logs, containing traffic data for an e-commerce website over some period of time. We assume to have access to at least the information contained in the *NCSA Combined* log format. The following fields are used:

- `host`, corresponding to the IP address of the Web client;
- `date:time`, being the time stamp of the request;
- `request` – information on the *HTTP method* (e.g., GET for downloading the server resource, HEAD for downloading the resource header) and the URI (Uniform Resource Identifier) specifying the requested server resource;
- `statuscode`, the numeric code informing about the result of request processing at the server (e.g., codes 2xx for the success, 4xx for client errors);
- `bytes` – volume of data transferred to the client;
- `referrer`, the URL (Uniform Resource Locator) which linked the client to the website (for the first request in session) or the URL of the recent page (for consecutive requests);
- `user_agent`, being the string specifying the client Web browser and platform.

Interactions of Web clients with the website may be represented as sessions. A *Web session* is defined as a sequence of HTTP requests coming from a client during a single visit. The HTTP protocol is stateless and Web-application-level session information is not stored in access logs, so session identification at this level remains uncertain and heuristics have to be used. As common practice, HTTP requests are grouped into the same session if they have the same IP address, the same user agent string and the time between consecutive requests does not exceed a threshold, set at 30 minutes (Bomhardt et al., 2005; Doran and Gokhale, 2016; Sisodia et al., 2015; Stassopoulou and Dikaiakos, 2009; Stevanovic et al., 2012).

Based on HTTP request fields available in log, various session features may be further determined, e.g. the total number of requests, session duration, mean time per page and many others.

A problem of the session-based bot recognition offline can be formalized as follows: given a set of HTTP request records from a Web session, *label* the session as performed by a bot or by a human. The information about all requests in session is entirely available at the time of decision making.

The research question addressed by this work is whether the task of recognizing bots offline is (1) learnable with standard ML methods, and (2) characterized by intrinsic differences between the behaviour of legitimate users and that of automatic software agents, such that the *unsupervised* analysis is able to reveal significant, interesting information.

4. Research methodology

4.1. Methodological framework

The general framework of our approach is presented in Fig. 1. It involves the following stages: (1) Session preparation based on HTTP log data (data pre-processing, session identification, extraction of session features, assigning ground truth labels to sessions); (2) developing and training a classification model with one of four ML methods applied to sessions in a training set; and (3) exploiting the model to classify sessions in a test set.

In this section our research methodology is discussed in detail. The data used is briefly characterized and four ML methods, used to develop session classifiers, are presented.

4.2. Preparation of sessions

4.2.1. Data description

Source data are HTTP access logs of an online bookstore[1]. The store is an osCommerce-based application, hosted on Linux Apache server supported with PHP and MySQL. The application allows a customer to perform

[1]The identity of the store is not revealed in the paper due to a non-disclosure agreement with the online retailer.

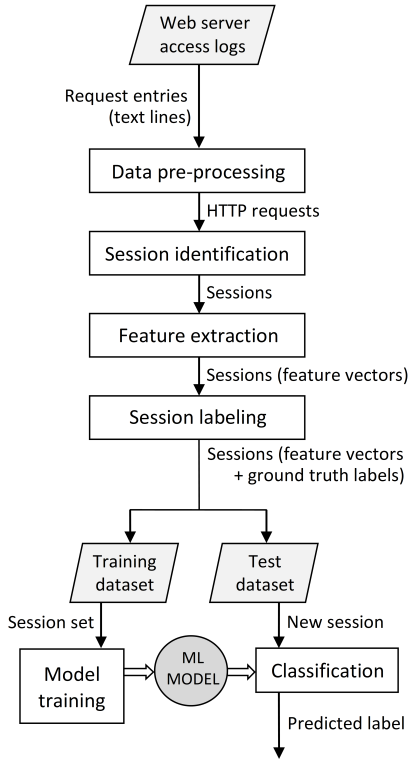

Figure 1: Flowchart of the proposed approach

typical e-commerce operations, like browsing and searching for products, reading detailed product information, adding products to the shopping cart, and finalizing the checkout process. The e-commerce module is integrated with some entertainment pages containing interactive puzzle, quizzes, mini games and short movies.

Logs were recorded in April 2014 according to the NCSA Combined format with a 1-second resolution and contained 1 397 838 entries.

A log analyser was implemented in C++ to pre-process Web server access logs, extract sessions and label them. Standard pre-processing steps were performed: reading data from files, merging them, and removing erroneous entries. Request records were assembled and sorted by their time stamps.

Based on the request dataset, sessions were reconstructed according to the standard approach (see Subsection 3). One-request sessions (generated by autonomous agents, in fact), one-page sessions (being typically human sessions making up the so-called bounce rate), as well as sessions generated by the website administrator and administrative software were eliminated.

4.2.2. Feature extraction

Twenty-one summary features descriptive of whole sessions were extracted from HTTP data. The features, listed in Table 1, may be divided into two categories.

The first category groups features no. 1 – 10, that have been commonly used as various subsets in previous bot detection studies (Bomhardt et al., 2005; Sisodia et al., 2015; Stassopoulou and Dikaiakos, 2009; Stevanovic et al., 2012; Tan and Kumar, 2002; Jacob et al., 2012; Alam et al., 2014; Zabihi et al., 2014; Stevanovic et al., 2013; Hamidzadeh et al., 2018; Zabihimayvan et al., 2017; Balla et al., 2011; Lagopoulos et al., 2017). They represent some aggregated or averaged session statistics, related to request types, request referrers, response codes, HTTP methods, time frequencies, and volumes of data sent to the client.

The second category includes e-commerce oriented features that have been distinguished taking into account the specificity of e-commerce websites. Features no. 11 – 20 are related to some typical operations performed by potential customers in an online store. Feature no. 21 reflects knowledge of the referrer-based source of the user visit, corresponding to the way in which the user reached the bookstore site (e.g., via a reference from an organic or paid search engine result) or lack of this knowledge in session.

In the feature extraction phase, boolean values were encoded as 0/1 while numeric features were individually normalized, i.e., scaled into $[0, 1]$. As a result, each session was represented as a 21-element feature vector.

4.2.3. Session labeling

For the purpose of development and/or verification of classification models, each session was assigned a ground truth label ("bot" or "human"). The log analyser maintains a table of user agent fields and IPs of known bots, as well as a table of regular expressions related to known Web browsers (including mobile browsers). These tables were built with the use of two online databases:

- *Udger* database (Udger, 2017), containing 43 user agent strings of legitimate Web browsers (categories "browser" or "mobile browser"), as well as 2832 user agent strings and 996 657 IPs of bots (categories "crawler", "fake crawler", "e-mail client", "validator", "offline browser", "multimedia player", "library", "known attack source – http", "known attack source – mail", "known attack source – ssh"); most browsers and bots have a client category, name, and version assigned.

- *User-agents* database (Staeding, 2017), containing 2459 user agent strings of known Web browsers and bots.

The table was augmented with additional user agents, semi-automatically discovered to represent bots based on some keywords occurring in them ("robot", "crawler", "spider", "worm", "search", "track", "harvest", "hack", "trap", "archive", "scrap", etc.).

Moreover, some heuristic rules were applied to label sessions based on well-known differences in accessing the Web content by humans and bots. Humans navigate the website via the browser software, following available hyperlinks; for each page the browser requests a page description file, followed by a batch of requests for embedded objects

Table 1: List of session features used.

Category	No.	Name	Type	Description
Common session features	1	*pag*	int	Total number of page views ("clicks") in session
	2	*req*	int	Total number of requests in session (session length)
	3	*vol*	double	Total volume of data sent to the client [KB]
	4	*dur*	int	Session duration [s]
	5	*timPP*	double	Mean time per page [s]
	6	*eRefR*	[0, 100]	Percentage of requests with empty referrer
	7	*eRefP*	[0, 100]	Percentage of page requests with empty referrer
	8	*4xx*	[0, 100]	Percentage of erroneous requests (4xx)
	9	*imgPP*	double	Image-to-page ratio
	10	*head*	[0, 100]	Percentage of requests of type HEAD
E-commerce -oriented features	11	*purch*	bool	Whether the session ended with a purchase
	12	*noH*	int	Number of views of the website's home page
	13	*noL*	int	Number of login operations (including "Register success" and "Login success")
	14	*noSh*	int	Number of views of the page with shipping terms and conditions
	15	*noS*	int	Number of searches using the internal search engine
	16	*noD*	int	Number of views of product description pages
	17	*noA*	int	Number of operations of adding a product to the shopping cart
	18	*noI*	int	Number of views of pages informing about the store and the trading company
	19	*noE*	int	Number of views of pages with entertainment contents
	20	*noB*	int	Number of other page views
	21	*src*	bool	Whether a "source" of the session is specified

(mostly images). This navigation is reflected by the information recorded in request referrers. On the contrary, robots may crawl the site according to their own strategies, not limited by the logical link structure. Thus, session properties indicative of a bot are the following: no image request in session, no page request in session, all requests with empty referrers, all page requests with empty referrers, all requests with HEAD method, all requests with 4xx status codes, or a request for *robots.txt* file occurred in session.

A session labeling procedure was the following (Suchacka and Motyka, 2018). If an IP address or a user agent was found in the table of known bots or one of the heuristic conditions indicative of a bot was met, a session was labeled as a bot. Else if a user agent was found in the table of known browsers, the session was labeled as a human. Otherwise, the session remained unlabeled.

As a result of session extraction and labeling, the session dataset included 13 397 sessions: 6195 bots and 7202 humans. Two unlabeled sessions were removed.

4.3. Classification methods

Four standard methods were applied to explore the feasibility of bot detection offline and assess the quality of results. Two strategies have been experimented with: (1) supervised learning and (2) unsupervised learning followed by supervised labeling (calibration).

4.3.1. Supervised classification

With the supervised classification we address the following research question:

Question I (problem solvability): By analyzing some summary information about a given Web session, already labelled by experts, is it possible to identify whether the client is a Web bot or a human-operated user agent?

The answer to this question is given by the use of two popular supervised classifiers: a multi-layer perceptron (MLP) neural network (NN) and a support vector machine (SVM).

These two supervised classification methods were selected as the most popular representative of two different classifier paradigms. An MLP neural network is essentially a generative classifier (Rumelhart et al., 1986) that learns an internal representation of class distributions, while an SVM is the most successful discriminative classifier, based on directly representing the decision boundaries (Duda et al., 2012). Both have been state-of-the-art methods for years, both were the default classifiers in different periods, and very effective training algorithms are available for both, due to extensive research efforts by the machine learning, statistics, and operations research communities.

A **multi-layer perceptron** (Goodfellow et al., 2016) is the basic structure of which deep neural networks are built. It is composed of two or more layers of computational neurons with nonlinear input-output mapping capabilities and can be used to "learn" complex classification tasks. MLPs have been successfully applied to many problems involving Internet traffic data (Bomhardt et al., 2005; Grzonka et al., 2018; Suchacka and Stemplewski, 2017; Zatwarnicki, 2012).

The data submitted at the NN input layer are feature vectors whereas each of the following network layers may receive the whole set of outputs produced by the preceding layers or some subset only. Each computational unit receives an array of inputs weighted by a set of corresponding parameters and computes a scalar, non-linear input-output mapping from it. Network learning consists in the weight optimization and may be performed with different algorithms. Approximation capabilities of a NN depend on a network size (number of layers, number of units in each layer) and structure (type of non-linear function, connec-

tivity pattern).

A **support vector machine** (Cortes and Vapnik, 1995) is a default choice for solving binary classification problems. It was the first successful "kernel method", build upon statistical learning theory to provide high generalization ability and performance guarantees. It proved successful in solving many Web session classification tasks, e.g., in Gržinić et al. (2015); Jacob et al. (2012); Suchacka et al. (2015).

To develop an SVM classifier, input samples (i.e., feature vectors from the training dataset) are represented as points in a high-dimensional space H, being transformed by some kernel function. The goal is to find a separating hyperplane that creates the largest possible margin between the points of two classes in the new space (this margin is determined by support vectors of two classes). Since in most cases the data is not linearly separable, the method imposes an additional penalty for each misclassified point, proportional to the distance from the margin boundary. Thus, the objective function maximizes the margin around the linear discriminant in H with the constraint on the total error penalty.

Various kernel functions may be applied, e.g., linear, polynomial, sigmoid, or RBF (Gaussian).

4.3.2. Unsupervised classification

Unsupervised classification allows us to answer the following research question:

Question II (presence of "natural" classes): If the answer to *Question I*, given by supervised classification, is largely positive, is it obtained because of intrinsic, structural differences between interaction profiles of humans and robots? Or does it stem from apparent correlation forced by the optimization procedures used?

The rationale of this research question is that classification is led by an optimization process. As such, it is subject to finding false relations that are not due to the underlying input-to-output mechanism, but nevertheless reduce the objective function on the available data. This happens even when using the cross-validation or other forms of generalization control because the intrinsic inductive bias in supervised methods makes them prone to overfitting.

To investigate if possible input-output correlations are real, we apply an unsupervised analysis followed by labelling to obtain the final classification. Two centroid-based clustering techniques are used: k-means and Graded Possibilistic c-Means (GPCM), with the subsequent majority labelling of the obtained centroids.

The k-means method is a standard first choice for clustering in metric spaces. Since it is a centroid-based algorithm, it has also been extensively used for approximate probability density estimation or vector quantization. Practical application shows that despite its simplicity and some drawbacks it is a remarkably effective method. The GPCM method is a more recent proposal by the authors. It was developed to allow for dealing with uncertainty, noisy data, and outliers, and for this reason it is designed as a fuzzy clustering method of the *possibilistic* type. It was selected because it has originally been introduced specifically for tasks similar to the one at hand.

K-means (MacQueen et al., 1967) is the most common and simplest, yet very efficient, central clustering technique. The algorithm is hard partitioning so each sample is attributed to only one cluster. Clusters are represented by their centres (centroids), whose elements are averages of the corresponding features' values for all instances in the cluster. The aim is to find a partition that minimizes the sum of distances between the cluster centroid and members over all clusters. Various distance metrics may be applied, including the most common Euclidean distance.

For a given number of clusters (k), the algorithm starts with generating initial centroid values, usually either randomly or with informed heuristics like k-means++ (Arthur and Vassilvitskii, 2007). Clusters are then built by determining the closest centroid for each sample. Then centroids are recalculated for the formed clusters and a new partition is created. This procedure keeps iterating until cluster membership stabilizes or some other stopping criterion is met.

A disadvantage of k-means is that a single run of the algorithm can only converge to a local optimum, depending upon the initial centroid values. As a result, for a given k and dataset, different initializations can lead to different final partitions. For this reason the algorithm is usually run many times with multiple different initial centroids each time, and the best version is finally selected or mean/median clustering results are reported, depending on a problem.

GPCM (Masulli and Rovetta, 2006) is a fuzzy clustering method derived from Possibilistic c-Means (Krishnapuram and Keller, 1993). It allows detection of outliers and makes learning more robust with respect to location identification (placing centroids) but gives the user more control than its original version. The method provides a soft transition between two types of membership: a relative (probabilistic) membership – indicating to what proportion a given sample should be attributed to each cluster, and an absolute (possibilistic) membership – indicating the strength of the sample attribution to any cluster independent from the rest.

This is a *fuzzy* central clustering method, implying that the cluster membership can be partial. This is represented by means of cluster indicators (or membership functions) which are real-valued rather than binary. A parametric uncertainty model is exploited to bound the possible combinations of membership values, leading to an idea of a graded possibility.

While centroids are defined as in all k-means-type algorithms, the specific graded possibilistic membership model employed in GPCM for a particular input pattern x with

respect to the centroid y_j of cluster j is computed as

$$u_i = \frac{e^{-\|x-y_i\|^2\beta_i}}{\left(\sum_{j=1}^{k} e^{-\|x-y_j\|^2\beta_j}\right)^{\alpha}}$$

where β_i is a width parameter for cluster i and α controls the possibility level, from a totally probabilistic ($\alpha = 1$) to a totally possibilistic ($\alpha = 0$) model, with a continuum of all intermediate cases for $\alpha \in (0, 1)$. Cluster learning is implemented by the Picard iteration for a given α, which iterates evaluation of the partial membership and determination of the cluster centres until convergence.

A description of the method, containing more details than those that fit within the scope of the present work, can be found in Masulli and Rovetta (2006).

4.3.3. Using unsupervised methods for building classifiers

Clustering was performed with a "vector-quantization" (Gray and Olshen, 1997) or "microclustering" (Aggarwal, 2007) approach which does not attempt to associate a meaningful group to each centroid, but uses (possibly many) centroids to approximate the support and density of data. This was obtained by allowing the number of clusters to be larger (possibly much larger) than the number of classes, two in the case at hand.

After the unsupervised density estimation step, it is necessary to assign class labels to generated clusters. A non-parametric classifier was obtained by labelling clusters with the majority class label, the one most represented among points in the cluster. In the crisp case labelling corresponds to the classic "nearest centroid" classification method. Thus, with k-means, each cluster is simply assigned the majority label, the one most represented among points in the cluster. In the fuzzy case, labels to one point are voted by all centroids proportionally to the membership of that point to each, so that the decision is collective, with the nearest centroids retaining the biggest influence. If clusters $y_1 \dots y_k$ are labelled with classes $C_1 \dots C_k \in \{\text{'bot'}, \text{'human'}\}$ and data point x belongs to each cluster with membership degrees $u_1 \dots u_k$ respectively, x is labelled with the class obtained as:

$$C(x) = \underset{\{\text{'bot'},\text{'human'}\}}{\operatorname{argmax}} \left\{ \sum_{C_i=\text{'bot'}} u_i, \sum_{C_i=\text{'human'}} u_i \right\}.$$

5. Experimental setup

This section describes the design decisions, including model hyperparameters values and the testing strategy adopted in the experiments.

5.1. Hyperparameters for the multi-layer perceptron classifier

Regarding the MLP classifier, while the choice of the activation function for the hidden units has an effect on the observed convergence speed and quality of training, it does not affect much the representation learning capacity of the classifier (Stinchcombe and White, 1989) (studies that "prove" the superiority of hyperbolic tangent over the logistic sigmoid, for instance, disregard the crucial role of a proper initialization (Drago and Ridella, 1992)). On the contrary, the number of hidden units is widely acknowledged as the most important model parameter.

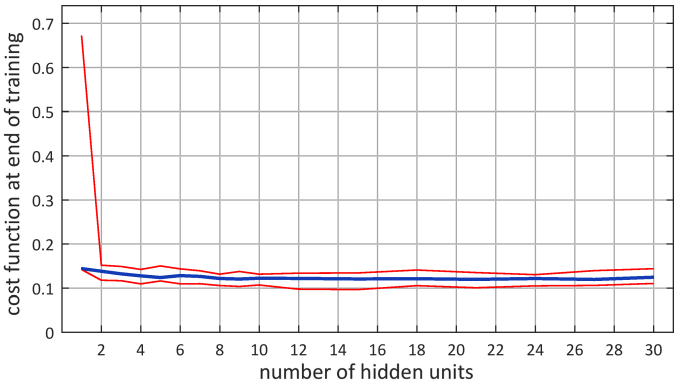

Figure 2: Classification performance of MLP vs. number of hidden units (median, minimum and maximum values over 20 trainings with different random initializations and random training/test splits).

In the present research we aimed at generality of the results, so we elected not to work under conditions of overfitting as is common in deep learning in the presence of huge, almost exhaustive training sets. The minimum number of hidden units was therefore searched.

An experimentally-driven model selection step for our data revealed that the performance does not vary much with the number of units in a hidden layer above a minimum of two, being very stable up to tens of hidden units (the results of this preliminary search step are plotted in Fig. 2). Thus, the number of hidden units was selected as two, the minimum value for which performance is reasonably stable over different trainings. The model was trained by the scaled conjugate gradient with the cross-entropy objective function

$$-\sum_{l} \left(t_l \log y_l - (1 - t_l) \log(1 - y_l)\right)$$

which implies a single sigmoid output unit for binary classification problems.

5.2. Hyperparameters for the support vector machine classifier

An SVM classifier does not have many user-selectable hyperparameters. The main design decision is related to the choice of the kernel function. Our SVM classifier turned out to be successful with the RBF kernel, with the width parameter equal to 0.1.

As to the number of support vectors, this is an output of the algorithm rather than a user input. For the case selected to illustrate the results, 761 support vectors were automatically determined by the optimization.

5.3. Hyperparameters in unsupervised analysis

Preliminary experiments showed that for both unsupervised classifiers the higher the number k of clusters is, the higher accuracy is achieved (Fig. 3). Note that a similar effect is trivially to be expected for the approximation error (computed as the mean square distance from the nearest prototype) evaluated on the training set. However, it is not obvious for classification error on a test set, since it does not only depend on the quality of the approximation, but also on the fact that smaller clusters retain a high out-of-sample purity.

The increase in performance with the increase in the number of centroids used is much less noticeable for GPCM than for k-means. Therefore, the GPCM method allows the use of a substantially lower number of centroids with a modest performance decrease. This makes the method particularly suitable for reverse-engineering the learned classification rule, since the readability of a simple partition is higher compared to that of a complex one. However, for larger k higher accuracy rates are achieved by the k-means-based classifier.

For the experiments, the following criteria were applied for selecting representative values for k: it was chosen larger than the number of classes to allow flexibility in density approximation (so that classes are not assumed to be linearly separable); three values differing by an order of magnitude each were chosen, to sample notably different conditions; the largest number was chosen to be equal to the number of support vectors in the supervised case, to allow a comparison between classifiers that use the same quantity of resources.

Following these criteria, the three values of 5, 50 and 761 were chosen for the number of centroids k in both the k-means and GPCM cases.

Some model parameters, α and β_i ($i = 1 \dots k$), are specific to the GPCM method (see Subsec. 4.3.2). In the experiments presented here, α was set at the value of 0.9, while the widths β_i were obtained dynamically for each centroid during the learning process and adapted at each iteration. Criteria and strategies for selecting these parameters are described in (Rovetta and Masulli, 2019).

5.4. Classifier performance evaluation

The session dataset was randomly divided into two class-proportion-preserving subsets of equal size (50%): a training set, used to learn classification models, and a test set, used to verify the classification efficiency on unseen data. Although systematic k-fold cross-validation was not used, 30 random splits were performed to increase the confidence in the obtained results.

With the aim of bot detection, class "bot" is considered *positive* and class "human" *negative*. For each of the four classifiers, results are described by means of a confusion matrix reporting the percentage of true and false positives (TP and FP, respectively) and true and false negatives (TN and FN, respectively). Many classification performance indices can be computed starting from the confusion matrix. Among those, a standard set was chosen to analyse the experimental results: recall (fraction of TP among all bot sessions), precision (fraction of TP among all sessions classified as positives), F-measure (harmonic mean of recall and precision) and accuracy (fraction of all correct classifications, both positive and negative).

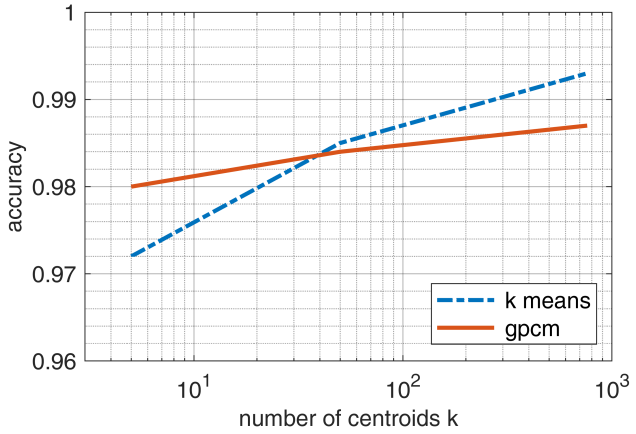

Figure 3: Accuracy vs. number of centroids for k-means and GPCM.

A Receiver Operating Characteristic (ROC) curve analysis was not performed because in all cases F-measure and accuracy are strongly correlated, which is an indication that type I and type II errors are effectively balanced.

Besides assessing the overall classifier efficiency, we analyse individual sessions misclassified by unsupervised and supervised methods for most representative cases, to address the following research question:

Question III (error analysis): Regarding the set of sessions that were not correctly recognized, is it possible to characterize them?

We investigate possible reasons for these misclassifications. Are they the same in supervised and unsupervised cases? Are the misclassified sessions deeply different from the rest (e.g., outliers)? Or were they mislabelled by the session labelling procedure?

6. Results and discussion

From all results of multi-trial learning runs, we selected those corresponding to the median objective function value so as to exclude the best and worst results which are statistically not representative.

Confusion matrices for supervised classification are presented in Fig. 4 and for the unsupervised one in Fig. 5 – 6. Table 2 summarizes performance indicators for all the classifiers: MLP with two hidden units, SVM with 761 support vectors found, k-means with $k = 5, 50, 761$, and GPCM with $k = 5, 50, 761$. Fig. 7 visualizes the performance scores.

An important note on the numerical precision: The results in confusion matrices are reported as percentages

Target class \ Output class	Non-bot	Bot	
Non-bot	54.00%	0.06%	*99.89%*
Bot	1.34%	44.60%	*97.08%*
	97.58%	*99.87%*	**98.60%**

(a) MLP classification.

Target class \ Output class	Non-bot	Bot	
Non-bot	53.62%	0.00%	*100.00%*
Bot	0.84%	45.54%	*98.19%*
	98.46%	*100.00%*	**99.16%**

(b) SVM classification.

Figure 4: Confusion matrices for supervised classification.

with two decimal digits; however, this is just to reduce inconsistencies due to rounding effects. Such a high numerical precision *should not* be interpreted as *measurement* precision, since the observed statistical variability in different trials was of the order of percent units from worst to best experiments.

6.1. Results of supervised classification

The confusion matrix for MLP (Fig. 4a) shows very good classification results: 44.60% of all sessions were correctly classified robots, 54.00% were correctly classified humans and only 1.34% and 0.06% were misclassified robots and humans, respectively. As much as 99.89% of all humans were correctly identified compared to 97.08% of recognized bots. There were 97.58% correct negatives among all negatives and 99.87% correct positives among all positive classifications. In total, 98.60% of all sessions were correctly identified.

The SVM classifier was even more efficient (Fig. 4b), achieving an overall accuracy of 99.16%; all humans and 98.19% of all bots were correctly classified.

Both supervised classifiers achieved very small false positives rates which means that only very few legitimate users were mistakenly classified as robots.

6.2. Results of unsupervised classification

Visualizations of the data do not reveal any clusters. Nevertheless, it is clear that the data have some structure, even if it does not take the form of clear groups. We present results for 5, 50 and 761 clusters for both clustering methods (as previously noted, the maximum investigated

Target class \ Output class	Non-bot	Bot	
Non-bot	54.00%	0.03%	*99.94%*
Bot	2.80%	43.17%	*93.91%*
	95.07%	*99.93%*	**97.17%**

(a) k-means, $k = 5$.

Target class \ Output class	Non-bot	Bot	
Non-bot	53.00%	0.24%	*99.55%*
Bot	1.30%	45.46%	*97.22%*
	97.61%	*99.47%*	**98.46%**

(b) k-means, $k = 50$.

Target class \ Output class	Non-bot	Bot	
Non-bot	53.14%	0.36%	*99.33%*
Bot	0.46%	46.04%	*99.01%*
	99.14%	*99.22%*	**99.18%**

(c) k-means, $k = 761$.

Figure 5: Confusion matrices for k-means + majority-labelling classification.

number of groups is equal to the number of support vectors determined for the best SVM classifier).

It can be noticed that for both k-means (Fig. 5) and GPCM (Fig. 6) the increase in the number of groups leads to the increase in accuracy, as well as in F-measure and recall (Table 2). After the calibration the microclustering ($k = 761$) makes it possible to find a reasonable structural difference between bots and non-bots. In general, for a given number of clusters k-means achieves better performance scores than GPCM (except precision, which decreases with k), although with a higher dependency on k (Fig. 3). Thus, further on in this paper we analyze the results of unsupervised classification (*UNSUP*) for the case of k-means with 761 clusters and juxtapose them with the results of SVM-based supervised classification (*SUP*).

Comparing the classification results in the fully supervised case with those in the unsupervised + labelling case, we can observe that the performance level is surprisingly

Target class	Output class: Non-bot	Output class: Bot	
Non-bot	53.53%	0.04%	*99.93%*
Bot	3.00%	43.43%	*93.54%*
	94.69%	*99.91%*	**96.96%**

(a) GPCM, $k = 5$.

Target class	Output class: Non-bot	Output class: Bot	
Non-bot	53.36%	0.28%	*99.48%*
Bot	1.42%	44.94%	*96.94%*
	97.41%	*99.38%*	**98.30%**

(b) GPCM, $k = 50$.

Target class	Output class: Non-bot	Output class: Bot	
Non-bot	53.91%	0.12%	*99.80%*
Bot	1.30%	44.68%	*97.17%*
	97.65%	*99.74%*	**98.59%**

(c) GPCM, $k = 761$.

Figure 6: Confusion matrices for GPCM + fuzzy-labelling classification.

similar. The quality of errors, however, is more favourable in *SUP* case, with the percentage of false positives that is (1) very small in absolute terms (close to 0), (2) smaller than the corresponding percentage in the unsupervised case for high number of clusters and (3) much smaller than the false negative rate. In *UNSUP* case the false positive rate is also smaller than the false negative rate, although the difference is not so apparent.

6.3. Investigation of misclassified sessions

Although the performance scores are very high, a fraction of sessions remains misclassified by both supervised and unsupervised learning classifiers. We examine if subsets of misclassified sessions overlap for *SUP* (SVM) and *UNSUP* (k-means, $k = 761$) cases. We also thoroughly analyse these sessions' features to find out whether they are atypical or whether the errors result from shortcomings of the session labelling procedure, e.g. caused by the fact that some camouflaging bot sessions could not have been properly labelled before the classification.

Table 3 summarizes quantities of misclassified sessions. The supervised classifier was unable to identify 70 sessions (including 10 sessions labelled as bots and 60 ones labelled as humans) while the unsupervised one was better, with only 53 unrecognised sessions (20 bots and 33 humans). As much as 43 sessions (10 bots and 33 humans) are in the intersection of misclassified subsets of both approaches. It is interesting to note that samples misclassified only by *SUP* are all humans (27 sessions) while samples misclassified only by *UNSUP* are all bots (10 sessions).

It is worth noting that some session features, used in previous bot detection studies for static websites, turned out not to be good discriminants of bots and humans for the analysed e-commerce website. In particular, the percentage of requests of type HEAD was extremely low for both classes.

Table 4 details values of most significant features for samples that were not correctly identified. Two first sections of the table present human sessions misclassified by both approaches (*H1-H10*) and only by the unsupervised classifier (*H11-H20*). Investigation of Web client categories and names revealed that most of these sessions were marked as generated by Web browsers (Chrome, Firefox, and Internet Explorer), and one by a mobile (Android) browser. The analysis of the number of pages in session (*pag*) and session durations (*dur*) does not allow us to unambiguously conclude about the actual client types; however, some other feature values (marked in boldface) are clearly atypical for legitimate users.

First of all, all the human sessions except *H1* have extremely low mean image-to-page ratios (*imgPP*). In reality, on the e-commerce website there is a lot of graphics and even on the home page and product description pages many additional images are presented (for novelties, bestsellers, items related with the currently displayed product, etc.). Thus, human visits are typically characterized by high image-to page rates (the mean for humans is 29.9, compared to 1.5 for bots). Very low *imgPP* values for the misclassified human samples suggest that these sessions might have been actually generated by bots. This applies especially to sessions *H2-H8* and *H13-H20*, which contain non-zero vales of *noB* and *noD* attributes, confirming views of some Web pages with the product information. This supposition is additionally confirmed by non-zero percentages of erroneous requests (*4xx*) and empty referrers (*eRefR* and *eRefP*) for some sessions (*H16-H20*). More ambiguous are sessions *H9-H12*, which do not contain typical operations performed in the Web store but only accesses to the entertainment pages ($noE > 0$), some of which may be based on server-side scripts and contain less embedded files.

Divergent characteristics may be noticed for sample *H1*. This session has *imgPP* and *4xx* typical for humans, but some other features suggest a bot client: many empty referrers, especially for page requests and very short mean

Table 2: Summary of performance indicators for various methods.

Method		Accuracy	Precision	Recall	F-measure
MLP	#h = 2	0.986	0.999	0.971	0.985
SVM	#sv= 761	**0.992**	**1.000**	**0.982**	**0.991**
k-means	$k = 5$	0.972	0.999	0.939	0.968
	$k = 50$	0.985	0.995	0.972	0.983
	$k = 761$	**0.992**	**0.992**	**0.990**	**0.991**
GPCM	$k = 5$	0.970	0.999	0.935	0.966
	$k = 50$	0.983	0.994	0.969	0.981
	$k = 761$	0.986	0.998	0.972	0.985

Table 3: Summary of misclassified sessions for supervised (*SUP*) and unsupervised (*UNSUP*) learning.

Metric	#sessions (#bots/#humans)	% of all sessions
Misclassified by *SUP*	70 (**10** bots/60 humans)	1.04%
Misclassified by *UNSUP*	53 (20 bots/**33** humans)	0.79%
Intersection (misclassified by both *SUP* and *UNSUP*)	43 (**10** bots/**33** humans)	0.64%
Misclassified by *SUP* but not *UNSUP*	27 (0 bots/27 humans)	0.40%
Misclassified by *UNSUP* but not *SUP*	10 (10 bots/0 humans)	0.15%

time per page (*timPP* equal to 5.5 seconds); the session lasted only 66 seconds and contained 13 page views.

To sum up the analysis of misclassified human sessions, we can conclude that these sessions are very likely to have been accomplished by intelligent agents, in fact. This means that our approach, in particular the k-means-based classification, is able to identify some bots hidden behind legitimate user agents.

Two last sections of Table 4 summarize selected features of robot sessions misclassified by both approaches (*R1-R33*) and only by the supervised one (*R34-R60*). The former group is dominated by visits of mail attacking tools, Google bots of various types (a search engine indexer, Web Preview screenshot creators, Web Light tools), and "probable bots" – sessions labelled as bot-generated as a result of applying heuristic rules. There is also one session of a Firefox fake crawler. Inability of the classifiers to detect these robots indicates that they are very advanced application-layer agents, successfully imitating online behaviour of human users.

Robots misclassified by both *SUP* and *UNSUP* methods seem to reveal two distinct kinds of traffic patterns. The first group, including sessions *R2-R18*, is characterised by a number of human-like characteristics: higher image-to-page ratios, no page requests with empty referrers, and relatively low (although non-zero) percentages of errors and requests with empty referrers (they have exactly the same values of *4xx* and *eRefR*). Mean time per page of the order of tens is also typical for humans (bots, in contrast, tend to either reveal *timPP* of one order of magnitude higher or exhibit extremely short times per page). Most of sessions in the second group, generated by "probable bots" (*R19-R32*), emulate legitimate traffic in terms of zero *eRefP*, *eRefR*, and *4xx*. On the other hand, they do not contain images and have significantly lower durations and times per page than humans. It has to be noted, however, that most of these sessions are extremely short and consist of only a few (page) requests. Such short sessions are known to be hard to classify and they are often eliminated from bot detection studies.

It is interesting to observe that one undetected robot (*R33*) performed a purchase operation in the Web store (*purch* = True), although no action connected with reading product details or adding products to the shopping cart was made ($noD = 0$ and $noA = 0$). This session lasted only 23 seconds and included two page views.

We were amazed to discover that the conventional, supervised learning approach to bot detection misclassified more bots than *UNSUP*. These sessions (*R34-R60*) were mostly due to search engine indexers and "probable bots"; there were also two marketing bots (MJ12bot and BLEXBot), a Java library tool, and two uncategorised agents. These sessions exhibit multiple typical bot traffic features, like extremely low image-to-page ratio, high percentage of erroneous requests, and especially enormous shares of requests and page requests with empty referrers. These sessions, again, consist of only a few pages or, on the contrary, are very long-lasting visits aiming at the systematic exploration of the website content.

Summarizing the discussion of results, the in-depth analysis of misclassified sessions allows us to conclude that (1) most of misclassified humans are bots, in fact and (2) misclassified robots are indeed robots. In the light of this finding we can clarify outcomes of the classification performance experimental evaluation and state that:

1. The unsupervised learning classifier recognised more robots than the supervised one (the number of misclassified robot sessions was lower for *UNSUP* than that for *SUP*).
2. Both approaches identified some camouflaged bots that had been improperly labelled as humans and the unsupervised approach was better in this respect (the number of misclassified human sessions was higher for *UNSUP* than that for *SUP*).

Table 4: Selected features of human (*H1-H20*) and robot (*R1-R60*) sessions that were misclassified by supervised (*SUP*) and/or unsupervised (*UNSUP*) learning classifiers.

Miscl.by	Id	*pag*	*req*	*vol*	*dur*	*timPP*	*eRefP*	*eRefR*	*4xx*	*imgPP*	*purch*	*noL*	*noS*	*noB*	*noD*	*noA*	*noE*	Category (Name)
SUP & UNSUP	*H1*	13	589	4881	66	5.5	**92.3**	**2.0**	0	43.2	False	0	0	2	10	0	0	Browser (Firefox)
	H2	2	7	31	3	3.0	0	0	0	**2.5**	False	0	0	0	2	0	0	Browser (Chrome)
	H3	3	7	92	28	14.0	0	0	0	**1.3**	False	1	0	2	0	0	0	Browser (IE)
	H4	18	37	332	246	14.5	0	0	0	**1.1**	False	1	0	9	3	4	0	Browser (Chrome)
	H5	2	3	33	25	25.0	0	0	0	**0.5**	False	0	0	1	1	0	0	Browser (IE)
	H6	4	8	67	**1615**	**538.3**	0	0	0	**1.0**	False	0	0	0	4	0	0	Mobile browser (Android browser)
	H7	10	28	257	618	68.7	0	0	0	**1.8**	False	0	0	5	5	0	0	Browser (Firefox)
	H8	2	5	37	12	12.0	0	0	0	**1.5**	False	0	0	0	2	0	0	Browser (Firefox)
	H9	11	24	363	**1825**	182.5	0	0	0	**1.2**	False	0	0	0	0	0	11	Browser (IE)
	H10	3	8	95	305	152.5	0	0	0	**1.7**	False	0	0	0	0	0	3	Browser (Chrome)
UNSUP	*H11*	2	7	120	305	**305.0**	0	0	0	**2.5**	False	0	0	0	0	0	2	Browser (Chrome)
	H12	4	12	144	**1767**	**589.0**	0	0	0	**2.0**	False	0	0	0	0	0	4	Browser (Chrome)
	H13	9	29	495	1007	125.9	0	0	0	**2.2**	False	1	0	5	2	0	0	Browser (IE)
	H14	3	7	149	904	**452.0**	0	0	0	**1.3**	False	0	0	3	0	0	0	Browser (Chrome)
	H15	26	86	500	680	27.2	0	0	0	**2.3**	False	2	0	11	4	1	0	Browser (Safari)
	H16	14	45	409	**1856**	142.8	0	0	**2.2**	**2.2**	False	3	0	1	6	0	0	Browser (Chrome)
	H17	28	31	32	1214	45.0	0	**6.4**	**87.1**	**0.1**	False	0	0	25	1	0	0	Browser (Firefox)
	H18	16	53	225	433	28.9	**87.5**	**30.2**	**3.8**	**2.3**	False	1	0	0	15	0	0	Browser (Chrome)
	H19	16	46	357	102	6.8	**87.5**	**37.0**	**6.5**	**1.9**	False	0	0	1	15	0	0	Browser (Opera)
	H20	46	88	804	1026	22.8	**2.2**	**1.1**	**1.1**	**0.9**	False	0	0	18	23	5	0	Browser (IE)
SUP & UNSUP	*R1*	2	2	2	**1037**	1037	50	50	**0**	0	False	0	0	0	0	0	2	Search engine bot (Googlebot)
	R2	3	101	1218	11	5.5	**0**	4.0	4.0	**32.0**	False	0	0	0	0	0	2	Attack source/MAIL (Firefox)
	R3	37	492	4666	1710	**47.5**	**0**	0.4	0.4	**12.2**	False	0	0	16	21	0	0	Attack source/MAIL (Firefox)
	R4	3	135	1223	85	**42.5**	**0**	1.5	1.5	**43.3**	False	0	0	2	1	0	0	Attack source/MAIL (Firefox)
	R5	5	266	2656	151	**37.8**	**0**	**0**	**0**	**51.8**	False	0	0	3	0	0	0	Attack source/MAIL (IE)
	R6	3	244	2385	**434**	217.0	**0**	0.8	0.8	**79.7**	False	0	0	2	1	0	0	Attack source/MAIL (Firefox)
	R7	23	161	1416	**308**	**14.0**	**0**	1.9	1.9	**5.9**	False	2	8	3	1	5	0	Attack source/MAIL (Firefox)
	R8	2	84	957	28	**28.0**	50	1.2	1.2	**40.0**	False	0	0	0	0	0	1	Attack source/MAIL (Android browser)
	R9	2	76	705	23	**23.0**	**0**	1.3	1.3	**35.5**	False	0	0	0	0	0	0	Tool (Googlebot)
	R10	2	77	690	15	**15.0**	**0**	1.3	1.3	**36.5**	False	0	0	0	0	0	0	Tool (Googlebot)
	R11	2	75	642	20	**20.0**	**0**	1.3	1.3	**35.5**	False	0	0	0	0	0	0	Tool (Googlebot)
	R12	2	77	703	13	**13.0**	**0**	1.3	1.3	**36.5**	False	0	0	0	0	0	0	Tool (Googlebot)
	R13	2	76	721	26	**26.0**	**0**	1.3	1.3	**36.0**	False	0	0	0	0	0	0	Tool (Googlebot)
	R14	2	76	732	13	**13.0**	**0**	1.3	1.3	**36.0**	False	0	0	0	0	0	0	Tool (Googlebot)
	R15	2	78	661	16	**16.0**	**0**	1.3	1.3	**36.5**	False	0	0	0	0	0	0	Tool (Googlebot)
	R16	2	166	1379	67	**67.0**	**0**	0.6	0.6	**81.0**	False	0	0	2	0	0	0	Fake crawler (Firefox)
	R17	13	213	1688	**301**	**25.1**	**0**	**0**	**0**	**15.2**	False	0	0	0	13	0	0	Screenshot creator (Googlebot)
	R18	2	58	754	**272**	272.0	**0**	**0**	**0**	**28.0**	False	0	0	0	0	0	1	Screenshot creator (Googlebot)
	R19	3	3	5.3	9	4.5	**0**	**0**	**0**	0	False	0	0	0	0	0	3	Probable bot - no image (-)
	R20	3	3	5.3	31	**15.5**	**0**	**0**	**0**	0	False	0	0	0	0	0	3	Probable bot - no image (-)
	R21	4	4	7	43	**14.3**	**0**	**0**	**0**	0	False	0	0	0	0	0	4	Probable bot - no image (-)
	R22	3	3	5.3	57	**28.5**	**0**	**0**	**0**	0	False	0	0	0	0	0	3	Probable bot - no image (-)
	R23	2	2	16	3	3.0	**0**	**0**	**0**	0	False	0	0	0	2	0	0	Probable bot - no image (-)
	R24	2	2	34	19	**19.0**	**0**	**0**	**0**	0	False	0	0	2	0	0	0	Probable bot - no image (-)
	R25	2	2	15	4	4.0	**0**	**0**	**0**	0	False	0	0	0	2	0	0	Probable bot - no image (-)
	R26	2	2	15	19	**19.0**	**0**	**0**	**0**	0	False	0	0	2	0	0	0	Probable bot - no image (-)
	R27	2	2	16	4	4.0	**0**	**0**	**0**	0	False	0	0	0	2	0	0	Probable bot - no image (-)
	R28	3	3	8	80	**40.0**	**0**	**0**	66.7	0	False	2	0	0	0	0	1	Probable bot - no image (-)
	R29	7	7	54	55	**9.2**	14.3	14.3	57.1	0	False	0	0	0	0	0	0	Probable bot - no image (-)
	R30	2	2	0.4	**1121**	1121	**0**	**0**	100	0	False	0	0	2	0	0	0	Probable bot - all 4xx (-)
	R31	3	3	0.6	11	5.5	**0**	**0**	100	0	False	0	0	0	0	0	0	Probable bot - all 4xx (-)
	R32	8	8	1.6	45	6.4	**0**	**0**	100	0	False	0	0	0	0	0	0	Probable bot - all 4xx (-)
	R33	2	13	62	23	**23.0**	100	15.4	7.7	5	True	0	0	1	0	0	0	Probable bot - no page ref (-)
SUP	*R34*	331	371	8143	86226	261.3	100	100	0.5	0.1	False	1	245	3	71	0	2	Search engine bot (Googlebot)
	R35	2936	3106	28512	81526	**27.8**	100	99.9	1.8	0.1	False	3	2129	38	583	0	85	Search engine bot (Googlebot)
	R36	39	72	1005	18454	485.6	100	58.3	**0**	0.8	False	0	29	2	7	0	1	Search engine bot (Googlebot)
	R37	2	4	10	3017	3017	100	50	**0**	0	False	0	0	0	0	0	2	Search engine bot (Googlebot)
	R38	2	18	1045	17679	17679	100	100	**0**	8.0	False	0	0	1	0	0	1	Search engine bot (YandexBot)
	R39	4	104	5353	46633	15544	100	100	**0**	**24.3**	False	0	0	1	0	0	1	Search engine bot (YandexBot)
	R40	54	63	82	36645	691.4	100	100	52.4	0	False	0	0	0	1	0	53	Search engine bot (YandexBot)
	R41	3	5	11	5730	2865	100	100	60.0	0.7	False	0	0	0	1	0	2	Search engine bot (YandexBot)
	R42	45713	47624	322770	10135	0.2	100	100	2.7	0	False	1198	15	14618	22953	2041	182	Search engine bot (Yahoo!)
	R43	553	564	3250	23844	**43.2**	100	100	**0**	0	False	13	0	127	235	71	68	Search engine bot (Yahoo!)
	R44	288	290	161	**659**	2.3	100	100	**0**	0	False	0	0	0	4	0	271	Marketing (MJ12bot)
	R45	784	1006	1923	3113	4.0	100	100	5.5	0.3	False	0	0	3	12	0	288	Marketing (BLEXBot)
	R46	201	324	5941	120	0.6	100	100	9.3	0.1	False	5	3	10	3	0	123	Library (Java)
	R47	16	1552	13951	3171	211.4	100	100	**0**	96.0	False	0	0	10	6	0	0	BOT - known bot (-)
	R48	935	1593	44196	7019	7.5	100	100	**0**	0.7	False	1	1	178	538	0	108	BOT - robots.txt req. (-)
	R49	3069	3956	2115	30302	**9.9**	100	100	98.5	0.3	False	0	0	757	2191	0	23	Probable bot - no page ref (-)
	R50	93	93	707	84505	918.5	100	100	**0**	0	False	0	0	0	0	0	93	Probable bot - no page ref (-)
	R51	124	124	943	48345	393.1	100	100	**0**	0	False	0	0	0	0	0	124	Probable bot - no page ref (-)
	R52	2	6	18	**1152**	1152	100	33.3	33.3	2.0	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R53	2	4	78	2	2.0	100	50	**0**	0.5	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R54	2	110	1034	103	103.0	100	16.4	1.8	**52.5**	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R55	2	9	69	**1220**	1220	100	22.2	**0**	3.5	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R56	2	6	18	198	198.0	100	33.3	33.3	2.0	False	0	0	0	0	0	2	Probable bot - no page ref (-)
	R57	2	72	489	12	**12.0**	100	25.0	2.8	**34.5**	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R58	2	19	41	**675**	675.0	100	84.2	73.7	8.5	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R59	2	20	37	**804**	804.0	100	90.0	80.0	9.0	False	0	0	0	2	0	0	Probable bot - no page ref (-)
	R60	7	7	12	8	1.3	**0**	**0**	**0**	0	False	0	0	0	0	0	7	Probable bot - no image (-)

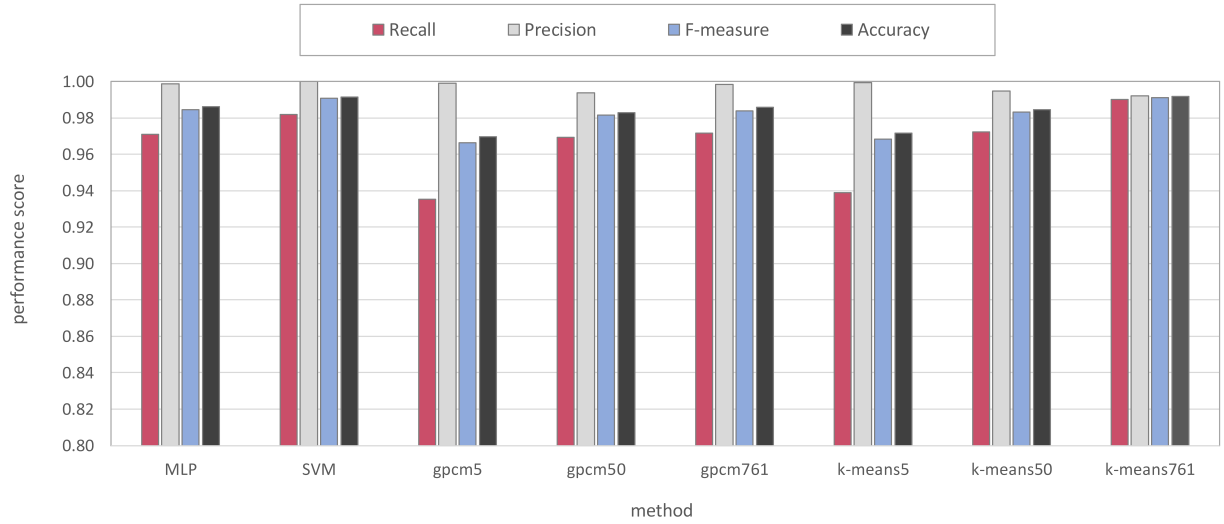

Figure 7: Graph summarizing experimental results.

7. Conclusion

In this paper we applied unsupervised and supervised learning methods to classify bot and human sessions in a Web store. Results of the experimental study demonstrate that the efficiency of the unsupervised classification is strikingly similar to that accomplished by the supervised one. Moreover, the outcomes expose an ability of the unsupervised approach to identify sessions of hidden robots by classifying them as humans based on their feature vectors. However, a very small fraction of bot sessions (most of which are extremely short) still remain undetected.

The obtained performance scores confirm that the task of the offline bot recognition is learnable with standard ML approaches to a very high extent. Furthermore, very high efficiency of *unsupervised* classification, approaching the analogous results for supervised learning, shows that intelligent agents and legitimate users interacting with an e-commerce site are characterized by intrinsic differences in their online behaviour. The results confirm the hypothesis that a structural difference is present in navigational patterns of bots and humans on an e-commerce site, which allows a learning machine to discriminate bots from regular traffic even without the supervision of class labels.

From the standpoint of generalisation ability, it can be proved that in terms of Vapnik-Chervonenkis dimension the unsupervised + labelling approach is advantageous with respect to the fully supervised one, because it depends on the number of centroids, not the total number of parameters (Ridella et al., 2001). This is an indication of a probable better out-of-sample performance, within the limits of the Vapnik-Chervonenkis theory (Vapnik, 1998), which is a worst-case approach. It also means that the sample complexity of learning under this paradigm is independent on input size, so results can be more safely extended to higher-dimensionality cases, i.e. using more observed features.

Our methodology may be used to improve labelling of bot and non-bot sessions for the use in experimental verification of ML approaches to Web bot detection, both in the "offline" and "online" scenario. Thus, our work is a step toward reliable benchmarking for bot detections studies. It is also a contribution to the area of developing novel bot detection methods. After adjusting session features the proposed approach may be applied to websites of any kind, not necessarily related to e-business. In future work we are going to extend our approach with a dynamic session feature selection stage and verify its performance for multiple Web traffic datasets.

Acknowledgment

This paper is based upon work from COST Action IC1406 High-Performance Modelling and Simulation for Big Data Applications (cHiPSet), supported by COST (European Cooperation in Science and Technology).

References

Abdullatif, A., Masulli, F., Rovetta, S., 2018. Clustering of non-stationary data streams: A survey of fuzzy partitional methods. Wiley Int. Rev. Data Min. and Knowl. Disc. 8, e1258. doi:10.1002/widm.1258.

Abdullatif, A., Masulli, F., Rovetta, S., Cabri, A., 2017. Graded possibilistic clustering of non-stationary data streams, in: Petrosino, A., Loia, V., Pedrycz, W. (Eds.), Fuzzy Logic and Soft Computing Applications: 11th International Workshop, WILF 2016, Naples, Italy, December 19-21, 2016, Revised Selected Papers. Springer International Publishing, pp. 139–150. doi:10.1007/978-3-319-52962-2_12.

Acarali, D., Rajarajan, M., Komninos, N., Herwono, I., 2016. Survey of approaches and features for the identification of HTTP-based botnet traffic. J. Netw. Comput. Appl. 76, 1–15. doi:10.1016/j.jnca.2016.10.007.

Adi, E., Baig, Z., Hingston, P., 2017. Stealthy Denial of Service (DoS) attack modelling and detection for HTTP/2 services. J. Netw. Comput. Appl. 91, 1–13. doi:10.1016/j.jnca.2017.04.015.

Aggarwal, C.C., 2007. Data streams: models and algorithms. Springer Science & Business Media.

Alam, S., Dobbie, G., Koh, Y.S., Riddle, P., 2014. Web bots detection using particle swarm optimization based clustering, in: Proceedings of IEEE Congress on Evolutionary Computation (CEC'14), IEEE. pp. 2955–2962.

Arthur, D., Vassilvitskii, S., 2007. k-means++: The advantages of careful seeding, in: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics. pp. 1027–1035.

Bai, Q., Xiong, G., Zhao, Y., He, L., 2014. Analysis and detection of bogus behavior in Web crawler measurement. Procedia Comput. Sci. 31, 1084–1091. doi:10.1016/j.procs.2014.05.363. 2nd International Conference on Information Technology and Quantitative Management, ITQM 2014.

Balla, A., Stassopoulou, A., Dikaiakos, M.D., 2011. Real-time Web crawler detection, in: Proc. Int. Conf. Telecommunications, pp. 428–432. doi:10.1109/CTS.2011.5898963.

Behal, S., Kumar, K., 2017. Detection of DDoS attacks and flash events using novel information theory metrics. Comput. Netw. 116, 96–110. doi:10.1016/j.comnet.2017.02.015.

Belshe, M., Peon, R., Thomson, M., 2015. Hypertext Transfer Protocol Version 2 (HTTP/2). IETF RFC 7540.

Berners-Lee, T., Fielding, R., Frystyk, H., 1996. Hypertext Transfer Protocol – HTTP/1.0. IETF RFC 1945.

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, Berlin, Heidelberg. pp. 113–124.

Calzarossa, M.C., Massari, L., 2011. Analysis of Web logs: Challenges and findings, in: Performance Evaluation of Computer and Communication Systems. Milestones and Future Challenges: IFIP WG 6.3/7.3 International Workshop, PERFORM 2010. Springer, Berlin, Heidelberg, pp. 227–239. doi:10.1007/978-3-642-25575-5_19.

Chu, Z., Gianvecchio, S., Koehl, A., Wang, H., Jajodia, S., 2013. Blog or block: Detecting blog bots through behavioral biometrics. Comput. Netw. 57, 634–646. doi:10.1016/j.comnet.2012.10.005.

Clark, E.M., Williams, J.R., Jones, C.A., Galbraith, R.A., Danforth, C.M., Dodds, P.S., 2016. Sifting robotic from organic text: A natural language approach for detecting automation on Twitter. J. Comput. Sci. 16, 1–7.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20, 273–297. doi:10.1023/A:1022627411411.

Doran, D., Gokhale, S.S., 2016. An integrated method for real time and offline Web robot detection. Expert Syst. 33, 592–606. doi:10.1111/exsy.12184.

Doran, D., Morillo, K., Gokhale, S.S., 2013. A comparison of Web robot and human requests, in: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'13), pp. 1374–1380.

Drago, G.P., Ridella, S., 1992. Statistically controlled activation weight initialization (SCAWI). IEEE Trans. Neur. Netw. 3, 627–631. doi:10.1109/72.143378.

Duda, R.O., Hart, P.E., Stork, D.G., 2012. Pattern classification. John Wiley & Sons.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T., 1999. Hypertext Transfer Protocol – HTTP/1.1. IETF RFC 2616.

GlobalDots, 2018. 2018 Bad Bot Report. Technical Report. GlobalDots. URL: `www.globaldots.com`.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Gray, R.M., Olshen, R.A., 1997. Vector quantization and density estimation, in: Proceedings of Compression and Complexity of SEQUENCES 1997, pp. 172–193. doi:10.1109/SEQUEN.1997.666914.

Gržinić, T., Mršić, L., Šaban, J., 2015. Lino - an intelligent system for detecting malicious Web-robots, in: Intelligent Information and Database Systems. ACIIDS'15, Springer International Publishing, Cham. pp. 559–568. doi:10.1007/978-3-319-15705-4_54.

Grzonka, D., Jakóbik, A., Kołodziej, J., Pllana, S., 2018. Using a multi-agent system and artificial intelligence for monitoring and improving the cloud performance and security. Future Gener. Comput. Syst. 86, 1106–1117. doi:10.1016/j.future.2017.05.046.

Guo, W., Ju, S., Gu, Y., 2005. Web robot detection techniques based on statistics of their requested URL resources, in: Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design, pp. 302–306. doi:10.1109/CSCWD.2005.194187.

Haider, C.M.R., Iqbal, A., Rahman, A.H., Rahman, M.S., 2018. An ensemble learning based approach for impression fraud detection in mobile advertising. J. Netw. Comput. Appl. 112, 126 – 141. doi:10.1016/j.jnca.2018.02.021.

Hamidzadeh, J., Zabihimayvan, M., Sadeghi, R., 2018. Detection of Web site visitors based on fuzzy rough sets. Soft Comput. 22, 2175–2188. doi:10.1007/s00500-016-2476-4.

Hayati, P., Potdar, V., Chai, K., Talevski, A., 2010. Web spambot detection based on Web navigation behaviour, in: Proc. Int. Con. Adv. Info. Net. (AINA'10), pp. 797–803. doi:10.1109/AINA.2010.92.

Jacob, G., Kirda, E., Kruegel, C., Vigna, G., 2012. PUBCRAWL: Protecting users and businesses from CRAWLers, in: Proceedings of the 21st USENIX Security Symposium, USENIX Association, Berkeley, CA, USA. pp. 25–25.

Jazi, H.H., Gonzalez, H., Stakhanova, N., Ghorbani, A.A., 2017. Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling. Comput. Netw. 121, 25–36. doi:10.1016/j.comnet.2017.03.018.

Kaur, R., Singh, S., Kumar, H., 2018. Rise of spam and compromised accounts in online social networks: A state-of-the-art review of different combating approaches. J. Netw. Comput. Appl. 112, 53–88. doi:10.1016/j.jnca.2018.03.015.

Krishnapuram, R., Keller, J.M., 1993. A possibilistic approach to clustering. IEEE Trans. Fuz. Sys. 1, 98–110.

Kwon, S., Oh, M., Kim, D., Lee, J., Kim, Y.G., Cha, S., 2012. Web robot detection based on monotonous behavior, in: Proceedings of the Information Science and Industrial Applications, p. 43–48. doi:10.1109/CSCWD.2005.194187.

Lagopoulos, A., Tsoumakas, G., Papadopoulos, G., 2017. Web robot detection in academic publishing. CoRR abs/1711.05098. URL: `http://arxiv.org/abs/1711.05098`, `arXiv:1711.05098`.

Lin, X., Quan, L., Wu, H., 2008. An automatic scheme to categorize user sessions in modern HTTP traffic, in: Proc. IEEE GLOBECOM'08, pp. 1–6. doi:10.1109/GLOCOM.2008.ECP.290.

Lu, W.Z., Yu, S.Z., 2006. Web robot detection based on Hidden Markov Model, in: Proceedings of International Conference on Communications, Circuits and Systems, pp. 1806–1810. doi:10.1109/ICCCAS.2006.285024.

MacQueen, J., et al., 1967. Some methods for classification and analysis of multivariate observations, in: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Oakland, CA, USA. pp. 281–297.

Masulli, F., Rovetta, S., 2006. Soft transition from probabilistic to possibilistic fuzzy clustering. IEEE Trans. Fuz. Sys. 14, 516–527.

Ridella, S., Rovetta, S., Zunino, R., 2001. K-winner machines for pattern classification. IEEE Trans. Neur. Netw. 12, 371–385. doi:10.1109/72.914531.

Rovetta, S., Cabri, A., Masulli, F., Suchacka, G., 2019. Bot or not? A case study on bot recognition from Web session logs, in: Quantifying and Processing Biomedical and Behavioral Signals. Springer. volume 103 of *Smart Innovation, Systems and Technologies*. chapter 19, pp. 197–206.

Rovetta, S., Masulli, F., 2019. Soft Clustering: Why and How-To, in: Fullér, R., Giove, S., F., M. (Eds.), Fuzzy Logic and Applications. WILF 2018. Springer, Cham. Lecture Notes in Computer Science. doi:10.1007/978-3-030-12544-8_6.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536. doi:10.1038/323533a0.

Sadiq, S., Yan, Y., Taylor, A., Shyu, M.L., Chen, S.C., Feaster, D., 2017. AAFA: Associative affinity factor analysis for bot detec-

tion and stance classification in Twitter, in: Proceedings of IEEE International Conference on Information Reuse and Integration (IRI'17), pp. 356–365. doi:10.1109/IRI.2017.25.

Saputra, C.H., Adi, E., Revina, S., 2013. Comparison of classification algorithms to tell bots and humans apart. Journal of Next Generation Information Technology 4, 23–32.

Singh, K., Singh, P., Kumar, K., 2018. User behavior analytics-based classification of application layer HTTP-GET flood attacks. J. Netw. Comput. Appl. 112, 97–114. doi:10.1016/j.jnca.2018.03.030.

Sisodia, D.S., Verma, S., Vyas, O.P., 2015. Agglomerative approach for identification and elimination of Web robots from Web server logs to extract knowledge about actual visitors. Journal of Data Analysis and Information Processing 03, 1–10. doi:10.4236/jdaip.2015.31001.

Staeding, A., 2017. User-agents.org. URL: `http://www.user-agents.org (access date: September 4, 2017)`.

Stassopoulou, A., Dikaiakos, M.D., 2009. Web robot detection: a probabilistic reasoning approach. Comput. Netw. 53, 265–278. doi:10.1016/j.comnet.2008.09.021.

Stevanovic, D., An, A., Vlajic, N., 2012. Feature evaluation for Web crawler detection with data mining techniques. Expert Syst. Appl. 39, 8707–8717. doi:10.1016/j.eswa.2012.01.210.

Stevanovic, D., Vlajic, N., An, A., 2013. Detection of malicious and non-malicious website visitors using unsupervised neural network learning. Appl. Soft Comput. 13, 698–708. doi:10.1016/j.asoc.2012.08.028.

Stinchcombe, M., White, H., 1989. Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions, in: IJCNN International Joint Conference on Neural Networks. doi:10.1109/IJCNN.1989.118640.

Suchacka, G., 2014. Analysis of aggregated bot and human traffic on e-commerce site, in: Proc. FedCSIS'14, pp. 1123–1130. doi:10.15439/2014F346.

Suchacka, G., Motyka, I., 2018. Efficiency analysis of resource request patterns in classification of Web robots and humans, in: Proc. Eur. Conf. Modelling and Simulation, pp. 475–481. doi:10.1109/CYBConf.2015.7175961.

Suchacka, G., Skolimowska-Kulig, M., Potempa, A., 2015. Classification of e-customer sessions based on Support Vector Machine, in: Proc. Eur. Conf. Modelling and Simulation, pp. 594–600. doi:10.7148/2015-0594.

Suchacka, G., Sobków, M., 2015. Detection of Internet robots using a Bayesian approach, in: Proceedings of IEEE 2nd International Conference on Cybernetics, pp. 365–370. doi:10.1109/CYBConf.2015.7175961.

Suchacka, G., Stemplewski, S., 2017. Application of neural network to predict purchases in online store, in: Information Systems Architecture and Technology: Proc. of ISAT'16 – Part IV, Springer. pp. 221–231.

Tan, P.N., Kumar, V., 2002. Discovery of Web robot sessions based on their navigational patterns. Data Min. Knowl. Discov. 6, 9–35. doi:10.1023/A:1013228602957.

Udger, 2017. Udger. URL: `https://udger.com (access date: September 4, 2017)`.

Vapnik, V.N., 1998. Statistical learning theory. Wiley, New York.

Walgampaya, C., Kantardzic, M., 2011. Cracking the Smart ClickBot, in: Proceedings of the 13th IEEE International Symposium on Web Systems Evolution (WSE'11), pp. 125–134. doi:10.1109/WSE.2011.6081830.

Zabihi, M., Jahan, M.V., Hamidzadeh, J., 2014. 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, 77–89.

Zabihimayvan, M., Sadeghi, R., Rude, H.N., Doran, D., 2017. A soft computing approach for benign and malicious Web robot detection. Expert Syst. Appl. 87, 129–140. doi:10.1016/j.eswa.2017.06.004.

Zatwarnicki, K., 2012. Adaptive control of cluster-based Web systems using neuro-fuzzy models. Int. J. Appl. Math. Comput. Sci. 22, 365–377. doi:10.2478/v10006-012-0027-4.

Zeifman, I., 2017. Bot Traffic Report 2016. Technical Report. Imperva Incapsula. URL: `https://www.incapsula.com/blog/bot-traffic-report-2016.html`.

Stefano Rovetta is Associate Professor of Computer Science at the University of Genova (Italy). He authored more than 170 scientific papers in machine learning, neural networks, clustering, fuzzy systems, and bioinformatics. He received the 2008 Pattern Recognition Society Award, and was the chair of international conferences. He is a member of the Italian Neural Network Society, the European Neural Network Society, and the European Society for Fuzzy Logic And Technology.

Grażyna Suchacka received the M.Sc. degrees in Computer Science and in Management, as well as the Ph.D. degree in Computer Science (with distinction), from Wroclaw University of Science and Technology (Poland). Now she is an assistant professor in the Institute of Informatics at the University of Opole (Poland). Her research interests include data analysis and modeling, data mining, and Quality of Web Service with special regard to bot detection and electronic commerce support.

Francesco Masulli is the Chair of IEEE Italy Section Computational Intelligence Society Chapter, a Full Professor of Computer Science with the University of Genoa (Italy), and an Adjunct Professor at the Temple University in Philadelphia (PA, USA). He held also positions at the Ansaldo Automazione (Genoa, Italy), the Radboud University in Nijmegen (NL), the International Computer Science Institute (Berkeley CA, USA), the University of Pisa (Italy), and the Université Nice Sophia Antipolis (France). He received the 2008 Pattern Recognition Society Award, was chair of several international conferences and schools, and authored more than 250 scientific papers in clustering, machine learning, neural networks, fuzzy systems, and bioinformatics.

Author biographies:

Stefano Rovetta is Associate Professor of Computer Science at the University of Genova (Italy). He authored more than 170 scientific papers in machine learning, neural networks, clustering, fuzzy systems, and bioinformatics. He received the 2008 Pattern Recognition Society Award, and was the chair of international conferences. He is a member of the Italian Neural Network Society, the European Neural Network Society, and the European Society for Fuzzy Logic And Technology.

Grażyna Suchacka received the M.Sc. degrees in Computer Science and in Management, as well as the Ph.D. degree in Computer Science (with distinction), from Wroclaw University of Science and Technology (Poland). Now she is an assistant professor in the Institute of Informatics at the University of Opole (Poland). Her research interests include data analysis and modeling, data mining, and Quality of Web Service with special regard to bot detection and electronic commerce support.

Francesco Masulli is the Chair of IEEE Italy Section Computational Intelligence Society Chapter, a Full Professor of Computer Science with the University of Genoa (Italy), and an Adjunct Professor at the Temple University in Philadelphia (PA, USA). He held also positions at the Ansaldo Automazione (Genoa, Italy), the Radboud University in Nijmegen (NL), the International Computer Science Institute (Berkeley CA, USA), the University of Pisa (Italy), and the Universit´e Nice Sophia Antipolis (France). He received the 2008 Pattern Recognition Society Award, was chair of several international conferences and schools, and authored more than 250 scientific papers in clustering, machine learning, neural networks, fuzzy systems, and bioinformatics.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: