Expert Systems With Applications 87 (2017) 129–140

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A soft computing approach for benign and malicious web robot detection

Mahdieh Zabihimayvan*, Reza Sadeghi, H. Nathan Rude, Derek Doran

Department of Computer Science and Engineering, Kno.e.sis Research Center, Wright State University, Dayton, OH, USA

ARTICLE INFO

Article history:
Received 12 October 2016
Revised 31 May 2017
Accepted 2 June 2017
Available online 7 June 2017

Keywords:
Markov clustering algorithm
Web Robot Detection
Web crawler
Malicious web agents
Fuzzy Rough Set Theory

ABSTRACT

The accurate detection of web robot sessions from a web server log is essential to take accurate traffic-level measurements and to protect the performance and privacy of information on a Web server. Moreover, the irrecoverable risks of visits from malicious robots that intentionally try to evade web server intrusion detection systems, covering-up their visits with fabricated fields in their http request packets, cannot be ignored. To separate both types of robots from humans in practice, analysts turn to heuristic methods or state-of-the-art soft computing approaches that have only been tuned to the specification of a kind of web server. Noting that the landscape of web robot agents is ever changing, and that behavioral patterns and characteristics vary across different web servers, both options are lacking. To overcome this challenge, this paper presents SMART, a soft computing system that simultaneously detects benign and malicious types of robot agents from web server logs and can automatically adapt to the session characteristics of a web server. The results of experiments over some access log file servers, each servicing different domains of the web, demonstrate outperformance of the proposed method on state-of-the-art ones for benign and malicious robot detection.

1. Introduction

Great numbers of modern Web-based technologies and services are required to study, analyze, and collect information from massive web repositories. Web robots (also called Web crawlers) are employed by such technologies to collect and scrutinize the dynamic content repositories contain. Perhaps surprisingly, robot agents are likely the dominant type of agent on the Web today: a 2015 industry report suggests that robots constitute 49.5% of all http requests to popular web sites on the internet (Incapsula, 2015), while our recent academic report suggests this number to be closer to 60% on academic web servers (Rude & Doran, 2015). The rise in robot traffic to these levels may be attributed to the widespread use of social media services that encourage users to share information with others in real-time. This dynamic information, which may be very valuable to organizations yet is time sensitive, need to be collected with robots carrying specialized functions and intentions. Web robots also carry out helpful tasks like web content archiving, link and HTML validation, search engine indexing, content change monitoring, and website mirroring.

Robots harvesting information is not an intrinsically bad thing. In fact, robots are crucial for powering the Web search engines society relies on to find information on the Web. They are also core to the "Internet of Things" concept (Atzori, Iera, & Morabito, 2010) whereby automated agents acting on behalf of physical devices will fetch data from Web sites, systems, and APIs. However, because their behaviors are completely unregulated and have been measured to be unethical (Sun, Councill, & Giles, 2010), some robots may pose a threat to the performance, privacy of information, and security of a Web server. Moreover, aggressive robots with specialized functions may find innovative ways to harvest e-mail addresses, perform click fraud, and access information behind 'paywalls' or login screens where only authorized human users should be allowed access.

It is difficult to assess whether a robot has malicious intentions when they visit a site. The only scenario where this task is easy is when a robot displays obvious signals of malice that can be detected via some heuristic. For example, robots requesting resources that have restricted access to registered users, and robots that submit a barrage of connection requests over a brief time period, likely have malicious intentions. But since these overtly dangerous behaviors are easy to identify, they may not be employed by intelligent malicious robots that try to circumvent detection. A strategy for subtler malicious robot visits is to have it *mask* or *hide* its requests by exhibiting behaviors that outwardly appear to be

* Corresponding author.
E-mail addresses: zabihimayvan.2@wright.edu, mahdieh.zabihi@gmail.com (M. Zabihimayvan), sadeghi.2@wright.edu (R. Sadeghi), howard.rude@wright.edu (H.N. Rude), derek.doran@wright.edu (D. Doran).

http://dx.doi.org/10.1016/j.eswa.2017.06.004
0957-4174/

human. For example, a robot could set the client IP address of a request packet to one it knows is a trusted human, or follow a navigational pattern that follows the structure of a site like a human. Such *malicious robot agents* are at least unethical in the sense that they attempt to circumvent detection and hence the differentiated treatment a web server administrator wants robot traffic to have, and at worst a threat to the information and the security of a web server as they intentionally seek out unauthorized resources, search for site vulnerabilities, and even launch attacks on the web server and the information it hosts. Previous studies on malicious robot traffic confirm their sophistication, exhibiting browsing behaviors not statistically significantly different compared to human traffic (Doran & Gokhale, 2011).

To improve the performance, security, and privacy of information on a web server, accurate detection of benign and malicious web robot agents is thus necessary. Detected benign robots may be given differentiated treatment, including limited access to resources or a lower priority for request servicing. Detected malicious agents could have their sessions flagged for subsequent security analysis, or given their suspicious nature, blocked from future access to a web server. A number of methods to accurately identify web robot sessions have been developed (Doran & Gokhale, 2011), but few attempt to tackle the problem of discovering malicious agents as well. Perhaps the leading, state-of-the-art solution for this problem combines soft computing methods (namely self-organizing maps coupled with adaptive resonance theory) to cluster visitors into groups of humans, well-behaved crawlers, malicious web robots, and unknown visitors (Stevanovic, Vlajic, & An, 2013). While the approach is very promising, the clusters are defined by a large suite of session features that may or may not be applicable depending on the nature of the web server and the robots that visit it (Tan & Kumar, 2002).

This article presents a new web robot detection algorithm able to discover human, benign, and malicious robot sessions simultaneously. It draws inspiration from the present art by employing soft computing concepts to classify web sessions, but innovatively integrates dynamic feature selection according to the patterns seen at a particular web server. In comparison to past systems for web robot detection, which use a limited number of features (Stevanovic et al., 2013) or utilize a feature selection strategy which is adequate just for certain domains (Zabihi, Jahan, & Hamidzadeh, 2014a, 2014b), our approach *dynamically* selects the most appropriate features from a large suite of candidates using a fuzzy rough set-based method. This automatic feature selection is a critical component since most web server administrators are unable to know, a priori, what session characteristics best distinguish web robots from humans. Experiments ran over traces of web traffic to two web servers, each servicing different domains of the internet, find that the proposed algorithm has higher accuracy compared to state-of-the-art methods, and hence, can detect robot agents that have benign and malicious intentions with higher reliability.

The rest of this paper is organized as follows: Section 2 gives a literature review in web robot detection. The proposed algorithm, called SMART (Soft computing for MALicious RoboT detection) is explained in Section 3. The results of our experiments and a comparative analysis against the state-of-the-art are presented in Section 4. Section 5 summarizes the article and discusses future work.

2. Related work

Discovering efficient and accurate ways to identify web robot traffic is an active area of research (Doran & Gokhale, 2011; Doran, Morillo, & Gokhale, 2013) with several existing studies utilizing supervised learning algorithms. For example, Tan and Kumar use a C4.5 decision tree to classify web visitors based on navigational patterns in click-stream data and a suite of 26 attributes (Tan & Kumar, 2002). Lourenço and Belo introduce a site-specific, updatable detection model called “ClickTips” for web robot detection (Lourenço & Belo, 2006) that also uses a C4.5 decision tree. Kwon et al. also use a decision tree to classify web sessions with some never before considered behavioral robot features (Kwon et al., 2012). They suggest the switching factor of the three attributes used in other literatures as the new attributes. Decision trees are not the only classification model considered for web robot detection, however. Recently, Gržinić et al. propose an intelligent system named “Lino” to diagnosis if a robot is malicious (Gržinić, Mršić, & Šaban, 2015). The authors exploit the support vector machine and C4.5 decision tree algorithms to compare their performance in implementing the Lino system. Bomhardt et al. present a web log pre-processing tool named “RDT” and apply Neural Networks and compare their results with those of the decision tree used by Tan and Kumar (2002) for web robot detection (Bomhardt, Gaul, & Schmidt-Thieme, 2005). What is more, the authors propose some new attributes in addition to the ones suggested by Tan and Kumar (2002). Stassopoulou and Dikaiakos introduce a probabilistic model based on the Bayesian networks and use a technique to adapt threshold for session extraction (Stassopoulou & Dikaiakos, 2009). They describe each session by six attributes known in Tan and Kumar (2002). Also, Suchacka and Sobkow utilize a Bayesian approach to detect web robots of a real e-commerce website while sessions are characterized by twenty-one attributes. Furthermore, they use cluster analysis to determine the parameters of the classification model (Suchacka & Sobkow, 2015).

Some other researches utilize several classification techniques and compare their performance in web robot detection. Stevanovic et al. examine the performance of seven different classifiers, including C4.5, RIPPER, k-Nearest, Naïve Bayesian, Bayesian Network, SVM, and Neural Networks, to categorize the web sessions into human and web robot groups. They propose two new attributes, the consecutive sequential request ratio and standard deviation of page request depth, and evaluate their effectiveness by the information gain and gain ratio metrics (Stevanovic, An, & Vlajic, 2012). In addition, Ensemble-based learners can be used to combine the advantages of several classifiers. Therefore, Sisodia et al. focus on bagging, boosting, and voting to identify robot sessions and also, perform multi-fold robot session labeling to enhance the classification performance (Sisodia, Verma, & Vyas, 2015). The authors extract twenty-three common attributes proposed in Tan and Kumar (2002) for each session.

In contrast to the previous researches, unsupervised learning to separate robots from humans have also been explored. Unsupervised methods are rooted in the hypothesis that there exist at least two classes of traffic (robot and human) with distinct behaviors and sessions that can be learned. For example, Zabihi et al. use DBSCAN (Density-Based Spatial Clustering of Applications with Noises) to identify web robots on two web servers (Zabihi et al., 2014a, 2014b). They propose two new attributes according to the behavioral patterns of web visitors suggested in Doran and Gokhale (2011) and Doran et al. (2013). In addition, they apply a *t*-test to check for attributes that are irrelevant to solve the curse of dimensionality problem. As another instance, Doran et al. employ the K-means clustering to classify 169 web robots of an academic website (Doran & Gokhale, 2009). In contrast to the current research, they just consider the classification of web robots and do not deal with general web visitor’s sessions. Also, they suggest three attributes based on workload metrics, volume of HTTP request sent, volume of bandwidth consumed, and average size of resources requested, which are also utilized in this paper.

Of the few studies that have tackled the detection of web robot and malicious web agents, Stevanovic et al. presented the

most promising approach by employing soft computing methods (Stevanovic et al., 2013. They employ unsupervised neural network models, namely a Self-Organizing Map (SOM) with Adaptive Resonance Theory (modified ART2), to cluster visitors into four groups of humans, well-behaved crawlers, malicious crawlers, and unknown visitors. The approach considers a set of distinguishing features between robots and humans often cited in the literature (Bomhardt et al., 2005; Lee, Cha, Lee, & Lee, 2009; Stassopoulou & Dikaiakos, 2009; Tan & Kumar, 2002) along with two new attributes, namely the standard deviation of requested page's depth and percentage of consecutive sequential HTTP requests. However, the dynamic behaviors of the many types of robots that crawl a web server (Tan & Kumar, 2002) may indicate that different subsets of features are relevant depending on the nature of the Web server under investigation.

To advance the state-of-the-art algorithms, our proposed algorithm not only dynamically selects the features per the real nature of Web servers by a novel fuzzy-based feature selection method, but also overcomes multiple weaknesses in the related work. For example, whereas some work assumes a statistical distribution over session characteristics (Zabihi et al., 2014a, 2014b), the proposed algorithm is agnostic to the probabilistic nature of the sessions. Furthermore, SMART employs a Markov clustering algorithm which has a simple and intuitive definition compared to other soft computing solutions involving neural systems with parameters set in an opaque fashion (Stevanovic et al., 2013). Finally, SMART differentiates user-agent strings into three categories of human, benign and malicious crawlers while most studies either only divide the agents into human and web robots or classify some of them in additional category called unknown visitors.

3. The SMART robot detection algorithm

It is natural to think that soft computing methods, which are machine learning models that embraces uncertainty and imprecision for tractable, robust, and low-cost solutions to problems, are a good fit for web robot detection. This is because:

- Benign and malicious agent detection must leverage features from the limited amount of data available in a server access log, which typically includes a client's IP address, a user defined user-agent field, the resource requested, response code, response size, and timestamp of the request at the coarse resolution of a second. These data limitations affect the quality and diversity of features that may be extracted.
- Finding malicious agents is a problem with an inherent amount of uncertainty. For example, inspection of a behavioral pattern may lead to malicious robots being labeled as a human (Doran & Gokhale, 2011). Thus, discovering malicious agents require an algorithm to trade-off statistical session characteristics against simpler behavioral patterns (e.g. requesting the file *robots.txt* or initiating a session from a non-root page).
- Evaluating recently seen sessions to decide if a robot agent should be marked for differentiated treatment must be done quickly to protect a web server. This requires the detection algorithm to be tractable so it can be implemented in real systems and low-cost to enable rapid detection with little overhead.

This section presents a novel soft computing approach, called SMART, to discover benign and malicious web robots. It differentiates itself from prior work with a novel feature selection mechanism that tunes the algorithm to a particular web system. SMART integrates a Markov Clustering (MCL) algorithm with Fuzzy Rough Set (FRS) theory to perform dynamic feature selection and detection. This section reviews the foundations of the MCL algorithm and FRS theory, and then explains how they are integrated into SMART.

3.1. Markov clustering

The MCL algorithm (Van Dongen, 2001) is a powerful method to cluster data points by simulating stochastic flows over an input graph. MCL has seen success in a variety of domains, such as social network analysis (Parthasarathy, Ruan, & Satuluri, 2011), knowledge base enrichment (Dutta, Meilicke, & Stuckenschmidt, 2015), community detection (Ruan, Fuhry, & Parthasarathy, 2013), and bioinformatics (Heinz, Selkrig, Belousoff, & Lithgow, 2015; Satuluri, Parthasarathy, & Ucar, 2010). The MCL algorithm is specified by an $M_{k\times k}$ column stochastic matrix, representing probabilities of transitions within a complete graph on k nodes. Nodes of this graph correspond to a data point (i.e. a web session) while transition probabilities (specified by the matrix element m_{ij}) specify the strength of a relationship or the degree of similarity among them. MCL finds a clustering of nodes in the graph by transforming M with iterative applications of three operations, namely *expand, inflate*, and *prune,* until a convergence criterion is reached. These operations are summarized below:

- *Expand*(M, e): This operation raises the input matrix M to an expansion parameter e. As M is stochastic, this exponentiation yields a new matrix whose ith row and jth column describe the probability that a random walk starting at state j will find itself at state i after e steps.
- *Inflate*(M, r): This operation computes the entry-wise Hadamard–Schur product (Huang, Xu, & Lu, 2016) of M. This product raises each element of M to its rth power and then divides elements by the sum of all elements in a corresponding column, i.e. $m_{ij} = m_{ij}/\sum_{p=1}^{k} m_{pj}$ to maintain column stochasticity. By enforcing the parameter r to be greater than 1, exponentiations of the elements of M forces large e-step transition probabilities to take higher values and low e-step transition probabilities to take lower ones. This enhances entries representing pairs of nodes participating in highly likely random walks over the graph. r also controls the granularity of clustering so that smaller values of r reveal smaller numbers of clusters.
- *Prune*(M): This operation sets all entries in M that are below a certain threshold to zero, thereby saving memory and reducing the number of iterations the MCL algorithm must perform. Following an inflation operation, the pruning threshold for the ith entry of column c of M is defined as: $a[\sum_{i=1}^{k} c_i^2]^b$, with $0 < a \leq 1$ and $b \in \{1, 2, \ldots, 8\}$ as tunable parameters with ranges specified in Van Dongen (2001).

When the elements of M do not roughly change value after an iteration of the MCL algorithm, an equilibrium state is reached and M becomes doubly idempotent. A clustering result is then derived according the strongly-connected components of a graph represented by the transformed M matrix where edges are defined by non-zero entries. Algorithm 1 summarizes the MCL algorithm.

3.2. Rough and Fuzzy Set Theory

Rough Set Theory (RST) describes how a collection of data may be separated based on a decision boundary and an indiscernibility relation (Pawlak, 1982). Defining each datum as a vector of discrete attributes, RST first defines a relation to check if two data vectors have identical feature values. Supposing a_k is the value of the kth attribute of datum s_i and N is the length of that vector (e.g. the number of features), the Indiscernibility Relation (IR) between two

Algorithm 1 Pseudo code of the MCL algorithm.

Input: column stochastic matrix ***M***, power parameter ***e***, and inflation parameter ***r***.
1. Normalize the ***M*** (column-wise normalization) to obtain a stochastic matrix.
2. **Repeat**
3. *Expand* by raising ***M*** to the e^{th} power.
4. *Inflate* all values of ***M*** with the parameter ***r***.
5. *Prune* any values of ***M*** that are small (which is defined based on the values of all elements in each column).
6. **Until** no elements of ***M*** change roughly between iterations.
7. Interpret resulting matrix to discover clusters.

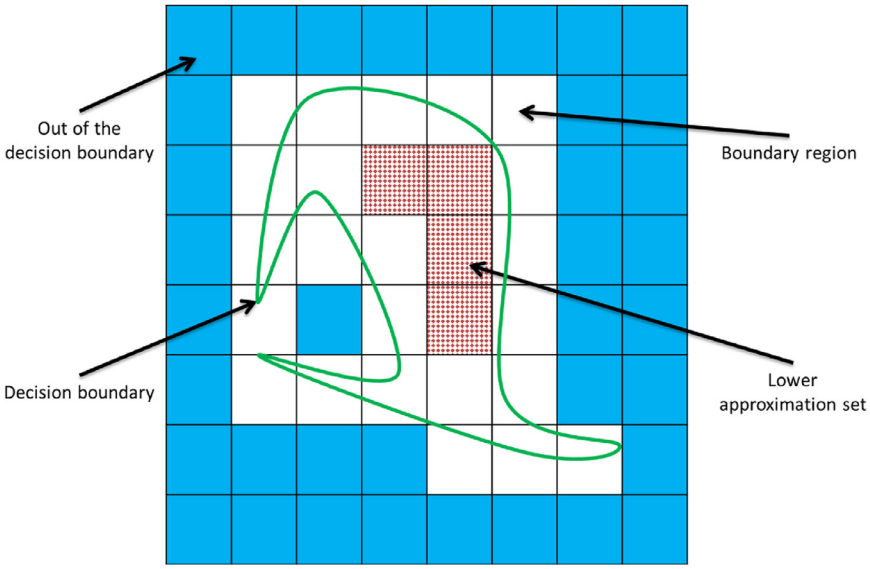

Fig. 1. An example of Rough Set Theory (RST) segregation. We imagine a data space partitioned into regions, called equivalence classes, where datums occupying the same region are equivalent. Data regions fully enclosed by the decision boundary are "described" by the boundary and is called its Lower Approximation Set (LAS). Regions spanning the decision boundary may or may not be "described" and is called its Upper Approximation Set (UAS). Regions outside of the decision boundary are "rejected".

datums is one such check for identical feature values:

$$\begin{cases} IR(s_i, s_j) = 1 & if\ \forall k = 1...N, a_k(s_i) = a_k(s_j) \\ IR(s_i, s_j) = 0 & otherwise \end{cases} \quad (1)$$

To better explain the theory, Fig. 1 illustrates a grid of regions modeling a *universe of discourse* partitioned by a green, closed, curved line that defines a *decision boundary*. The square regions represent areas of the data space occupied by datums that exhibit identical feature values, e.g., have $IR = 1$ for all pairs in the region. In other words, each square indicates an *equivalence class* in the universe of discourse. As the red dotted squares are lying fully within the decision boundary, they represent a group of equivalence classes that are fully described by this boundary. We call these classes the boundary's *lower approximation set (LAS)*. The white squares represent sets of equivalence classes that may lie inside or outside of the decision boundary. This boundary region and *LAS* form *upper approximation set (UAS)*. The blue squares that fully lie outside of the decision boundary are fully rejected by the decision boundary.

Define $[s_i]_{IR}$ as the set of all datums in the ith equivalence class of a universe and *DB* as the function that defines a decision boundary. Then the *LAS* of *DB* is given as:

$$\underline{DB_{IR}} = \{s_i \in Data\ Set | \forall s_j \in [s_i]_{IR} : DB(s_j) \leq 0\} \quad (2)$$

While the *UAS* of *DB* is given by:

$$\overline{DB_{IR}} = \{s_i \in Data\ Set | \exists s_j \in [s_i]_{IR} : DB(s_j) \leq 0\} \quad (3)$$

Rough Set Theory can be extended to consider Fuzzy Rough Sets (FRS) (Dubois & Prade, 1990) where datums can exhibit both continuous and discrete attributes. Under FRS, data attributes are one of two types: a *label* or an *attribute*. Labels define the attributes relevant to determine the class of a datum, while attributes define the remaining features. In this setting, the indiscernibility relation takes on a similarity measure to assess the equivalence of two datums. Eq. (4) indicates this similarity for a pair of datums s_i and s_j with continuous features normalized to the range [0, 1]:

$$\forall a \in Attributes, Label : IR_{Attributes,Label}(s_i, s_j) = \underset{a}{\tau}(\max(0, 1 - \left|s_i(a) - s_j(a)\right|^2)) \quad (4)$$

In the above equation, $s_i(a)$ and $s_j(a)$ are the value of attribute a in the samples and $|s_i(a) - s_j(a)|^2$ defines their squared difference. τ is called a Triangular-norm (t-norm) operator that aggregates the similarity of the whole attributes together and returns a continuous number between 0 and 1. We utilize the Lukasiewicz form of τ (Radzikowska & Kerre, 2002):

$$\forall x, y \in [0, 1] : \tau(x, y) = \max(0, x + y - 1) \quad (5)$$

FRS then defines a value that reflects if a datum s_i is in the LAS of a decision boundary called the *Lower Approximation Membership* $\underline{\mu_{DB}(s_i)}$. In essence, the larger the value of $\underline{\mu_{DB}(s_i)}$, the larger amount of faith we have that s_i is completely described by a decision boundary. It is defined as:

$$\underline{\mu_{DB}(s_i)} = \inf_{s_j \in DataSet, s_i \neq s_j} \mathrm{I}(IR_{Attributes}(s_i, s_j), IR_{Label}(s_i, s_j)) \quad (6)$$

where $IR_{Attributes}(s_i, s_j)$ and $IR_{Label}(s_i, s_j)$ is the *IR* between samples s_i and s_j based on their *attributes* and *label sets*, respectively. In Eq. (6), the I is defined as an *Implicator*, which is another fuzzy operator to aggregate continuous values. We again consider the Lukasiewicz form of I (Radzikowska & Kerre, 2002):

$$\forall x, y \in [0, 1] : I(x, y) = \min(1, 1 - x + y) \quad (7)$$

Table 1
Primary attributes extracted for each session.

Id	Attribute name	Description
1	Trap file request	Shows that if a session includes requests for trap files which are the resources human users are unaware of their existence since there are no links to them in a website page
2	Session time	The approximated time difference between the first and the last requests in a session
3	% Night	The percentage of requests demanded between 12 a.m. and 7 a.m.
4	% NullReferrer	The percentage of requests with unassigned referrer
5	SD_RPD	Standard Deviation of Requested Page's Depth which states the standard deviation of page's depth across all requests sent in an individual session
6	% CSR	Percentage of Consecutive Sequential HTTP Requests of a session for all pages belonging to the similar directory
7	SCB	Number of bytes sent from the server to a client
8	Avg time	The average time between two requests
9	% Head	Percentage of HTTP requests of type HEAD sent in a single session
10	% 4xx	Percentage of erroneous HTTP requests sent in a single session
11	Penalty	A penalty value for every back-and-forward navigation or loop
12	Max barrage	Maximum rate of browser file requests sent in a single session
13	SF-FileType	Switching factor of file types for each individual session
14	SF-csbytes	Switching factor on number of bytes from clients to the server
15	SF-referrer	Switching factor on unassigned referrer field
16	Click number	The number of HTTP requests sent in a session
17	depth	The maximum depth of the tree within the graph showing all the HTTP requests sent in a session
18	totalPages	Total number of pages requested in a single session
19	PPI	Page popularity index showing the average value of page popularity index for all pages retrieved in a single session
20	HTML-to-Image ratio	The number of HTML page requests over the number of images requested in an individual session
21	% Zip	Percentage of zip/gz files requested in a session
22	% Binary Doc	Percentage of PDF/PS files requested in a session
23	% Binary Exec	Percentage of cgi/exe files requested in a session
24	MultiIP	Shows that if a session contains multiple IP addresses
25	MultiAgent	Shows that if a session contains multiple user agent strings
26	% 304	Percentage of HTTP requests sent in a single session with status code 304
27	% Multimedia	Percentage of multimedia files requested in a session
28	% Other	Percentage of other type of resources requested in a session
29	% POST	Percentage of HTTP requests of type POST sent in a single session
30	% GET	Percentage of HTTP requests of type GET sent in a single session

The infimum operator in Eq. (6) and returns the minimum of all implication values for all comparisons of s_i to other datums. Note that these values are thus very sensitive to datums exhibiting extremely small or large attribute values. To reduce this sensitivity, we apply the Ordered Weighted Average (OWA) function to squish $\underline{\mu_{DB}(s_i)}$ for each sample (Sadeghi & Hamidzadeh, 2016; Verbiest, Cornelis, & Herrera, 2013). OWA takes the result of every implicator and places them in descending order. Following this, OWA assigns ever larger weight coefficients to smaller implicator values. OWA then sums the product of the weights and implicator values as its result. Eq. (8) describes OWA on the generic input vector, $[I_1,..., I_{L-1}]^T$with values ordered in descending order:

$$OWA(I_1, ..., I_{L-1}) = 2((L-1)L)^{-1} \sum_{i=1}^{L-1} i.I_i \tag{8}$$

With the use of OWA Eq. (6) can be rewritten as:

$$\underline{\mu_{DB}(s_i)} = \underset{s_j \in DataSet, s_i \neq s_j}{OWA} \mathrm{I}(IR_{Attributes}(s_i, s_j), IR_{Label}(s_i, s_j)) \tag{9}$$

The value of $\underline{\mu_{DB}(s_i)}$ for sessions is considered in feature selection for our robot detection algorithm, which is described next.

3.3. SMART detection algorithm specification

SMART takes as input a web server access log file that records all requests to a server over a specific time, preprocessing it into visitor sessions by a thirty-minute timeout (Stevanovic et al., 2013). The literature on Web robot traffic analysis (Bomhardt et al., 2005; Kwon et al., 2012; Lee et al., 2009; Stevanovic et al., 2013; Tan & Kumar, 2002; Zabihi et al., 2014b) suggests that the user and session attributes listed in Table 1 are useful in segregating Web visitors by their visiting patterns.

SMART considers these attributes curated from past scientific studies. However, despite the ability of these attributes verified for different data sets, the varying behaviors of humans and web robots (Doran & Gokhale, 2009) across different websites (Dikaiakos, Stassopoulou, & Papageorgiou, 2005) suggest that the most relevant features for a given web server may not be adequate for another. SMART thus performs dynamic feature selection. For this purpose, it considers the *decision boundary* to be the behavior of an algorithm that labels robot and human sessions and the *universe of discourse* as the set of sessions.

Training data is necessary for FRS to learn its decision boundaries for dynamic feature selection. To create such training data, we consider a naïve approach to assign sessions to be "robot" or "human" induced. The approach, summarized in Fig. 2, assigns all sessions that contain the user-agent of a known web browser (used by humans) and do not access to the file *robots.txt* as a human. Otherwise, if a user-agent string matches that of a known web robot according to the resource *user-agents.org* and *botsvsbrowsers.com*, it is labeled as such. The labeled robot sessions are further divided into "benign" and "malicious" classes according to its classification on *user-agents.org* or *botsvsbrowsers.com*. The process is reasonable because, following past work and the expectations of human session behavior, we assume that sessions with a request to *robots.txt* is not a typical human session. Moreover, for robots not accessing *robots.txt*, we look up the user-agent string on one of two crowdsourced databases, thus requiring some expert to have identified and reported the agent as a web robot in the past.

This approach strives to minimize uncertainty by only labeling sessions as robot induced if they request *robots.txt*. Moreover, robot sessions are only labeled as malicious if their user-agent field is already blacklisted. While the process may cause some robot sessions to go unlabeled, it ensures that SMART is trained against gen-

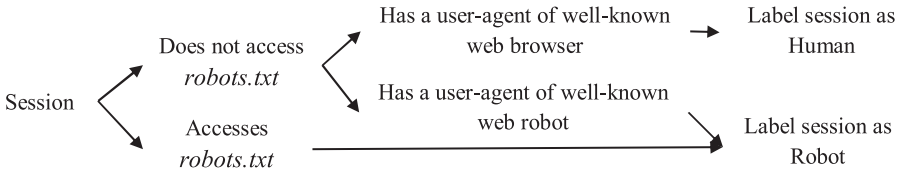

Fig. 2. Process to manually label sessions as "robot" or "human".

uine (malicious) robot and human sessions. Note that this same naïve procedure was also followed in another state-of-the-art web robot detection approach (Stevanovic et al., 2013).

We apply the Forward Approximation Algorithm (FAA) (Qian, Wang, Cheng, Liang, & Dang, 2015) to perform feature selection based on FRS because of its ability to expedite the selection by merging sampling and dimensionality reduction into a single step. FAA endeavors to discover a minimal group of attributes ϕ that maximizes a *describability measure* γ_ϕ, which is the degree to which the attributes define a boundary that correctly classifies sessions into human and robot groups. γ_ϕ is a real number in the range [0,1] and is given by:

$$\gamma_{\phi \subseteq Attributes} = \frac{\left|\sum_{s_i \in DataSet} \underline{\mu_{DB}(s_i(\phi))}\right|}{|Data\ Set|} \tag{10}$$

where $s_i(\phi)$ is a datum representing the ith session in the data set using only the set of attributes ϕ. This above equation essentially computes the average value of $\underline{\mu_{DB}(s_i(\phi))}$ across the data set. FAA is motivated by the fact that higher values of $\underline{\mu_{DB}(s_i(\phi))}$ suggest that $s_i(\phi)$ is better described by a decision boundary, thus the ϕ maximizing γ_ϕ is a set of features that best describes the features of web robot and human traffic. To find this best ϕ, FAA is a greedy iterative process that starts with a value of γ for each singular attribute. The attribute with the maximum value for γ is selected as the first member of the set ϕ. In the next iteration, the change in γ_ϕ if each attribute outside of ϕ were added is calculated, and the attribute yielding the largest increase is added. This process continues until the addition of any further attributes does not increase the value of γ_ϕ.

It is worth mentioning that although FAA significantly reduces the computational time for high dimensional data through its greedy approach (Qian et al., 2015), it still suffers from computational complexity because computing γ_ϕ means computing $O(n^2)$ Implicator operations (where n is the number of sessions in a log), and that is just for one possible setting of ϕ. To address this issue, we draw inspiration from validation methods and construct ten *folds,* which are randomly drawn, disjoint subsets of the set of sessions in a Web server log. To develop the feature set ϕ, we run FAA separately for each package and examine the frequency with which a feature appears in the final set ϕ of each package. We then define the *frequency index* of an attribute a as the percentage of packages where a appears in ϕ. The final set of features used is of those whose frequency index exceeds a threshold value *FRS_thr*.

After feature selection, SMART uses the same ten folds of labeled sessions in an MCL routine to cluster them using the features listed in Table 1. First, sessions from nine of the ten folds are translated into a complete graph where nodes correspond to sessions and an edge from session i to j is annotated according to the similarity of the session attributes included in ϕ. To compute this similarity, we first use min–max normalization (Jain & Bhandare, 2011) to normalize the attribute values to fall in the range [0, 1] to avoid bias towards attributes located in wider ranges:

$$\forall a \in Attributes : a' = \frac{a - \min_a}{\max_a - \min_a} \tag{11}$$

where $\min_a$ is the lowest observed value of attribute a while $\max_a$ is the highest observed value. For every session i, the normalized values of the attributes in ϕ are compiled into a vector σ_i. Recognizing that attribute values may be small or sparsely defined because of the diversity of behaviors sessions present, a useful measure of similarity for MCL is cosine similarity (Liao & Xu, 2015):

$\forall A, B \in Web\ sessions, n = Number\ of\ Attribute\ s\ in\ \phi$:

$$Sim(A,B) = \frac{\sigma_A . \sigma_B}{\|\sigma_A\| . \|\sigma_B\|} = \frac{\sum_{i=1}^{n} \sigma_{A_i} . \sigma_{B_i}}{\sqrt{\sum_{i=1}^{n} {\sigma_{A_i}}^2} . \sqrt{\sum_{i=1}^{n} {\sigma_{B_i}}^2}} \tag{12}$$

We compose the cosine similarity of all pairs of sessions into a similarity matrix M_S. These similarity scores are transformed into an adjacency matrix, M_{adj} by setting the $k/2$ indices in each row with highest similarity scores in M_S to 1 in M_{adj}. This heuristic, while simple, is essential to help Algorithm 1 converge in reasonable time and still yields a clustering result useful to find similar human, benign robot, and malicious robot sessions (Van Dongen, 2001). We further apply singleton filtering (Szilágyi, Medvés, & Szilágyi, 2010) after each iteration of MCL to eliminate any session that seems to be isolate. Such sessions will have zero elements along their corresponding row and column except for the element on the diagonal of the adjacency matrix, and hence, appear as an isolate that no random walk may ever reach. Neither the *expand* nor *inflate* operations can impute any non-zero value into this row and column, so we remove them prior to the next execution of the MCL main loop to reduce the dimensionality of the matrix and improve the performance of the approach.

We assign the inflation parameter $r = 2$ according to the recommendation of the authors of the MCL algorithm. The numerous resulting clusters, formed from nine of the ten folds, represent Web sessions driven by agents that behave under a similar behavioral profile. Each cluster is assigned the label "benign robot", "malicious robot", or "human" according to the majority type of session seen in the resulting clusters. SMART is subsequently tested by labeling each session in the one fold not used for clustering by the label of the cluster it's feature vector is closest to (as defined by cosine similarity to the cluster centroid). Finally, the flowchart of the proposed algorithm is shown in Fig. 3.

4. Experimental results

This section presents the results of the experiments to study the performance of SMART. In practice, a web robot detection system must be able to determine if sessions as-yet-unseen should be labeled as robots or humans given a history of previous sessions. Thus, our experimental approach does not use synthetic or simulated data, but rather, performs a *playback* of the sessions seen in the real web server logs, thus testing SMART in scenarios where it observes real collections of sessions before making a classification decision on an unknown session. A 10-fold cross-validation strategy provides an average performance of our system when trained with, and tested against different partitions of the data over time. For this evaluation, we evenly partitioned the web logs into 10 folds ordered by time, considering the collection of sessions that started their visits within the bounds of a fold together during

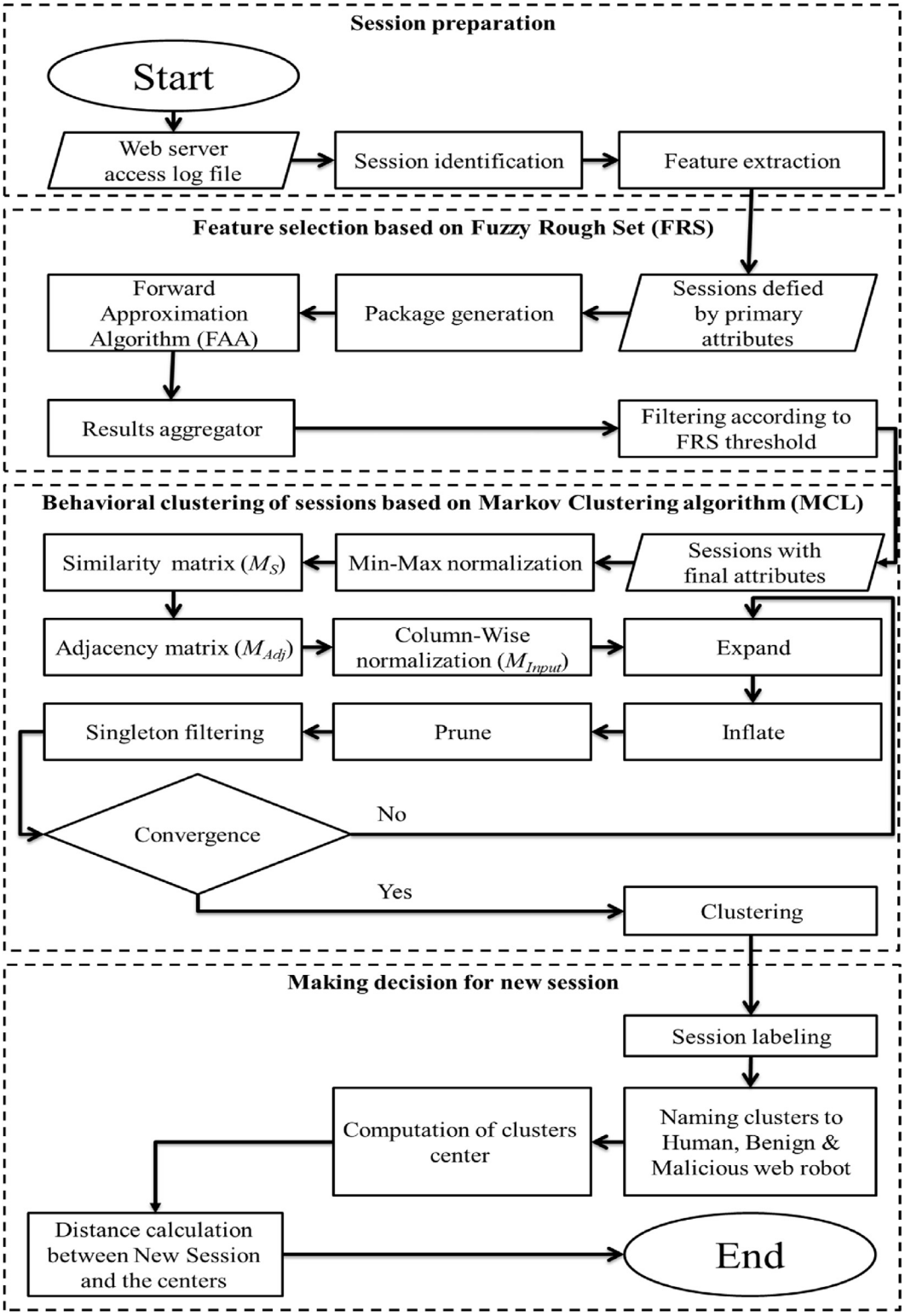

Fig. 3. The flowchart of SMART.

training or testing. For each fold, we evaluate SMART and its competing algorithms (described below) by considering it as a "test" fold and the trained algorithms over the 9 unselected folds. The algorithms are thus trained over collections of sessions on a web server over contiguous periods. We first describe the datasets used for evaluation, then study the sensitivity of SMART's performance against the value of FRS_thr parameter, and finally compare SMART against stat-of-the-art algorithms.

4.1. Data set description

Several experiments were conducted over two real web server access logs to evaluate SMART. The first log file is from the Data

Table 2
Description of data sets utilized in this paper.

Data set name	# of requests	# of sessions	# of benign robots	# of malicious robots	# of humans	Session length average (min)
DS-IR (Imam Reza International University, 2015)	102,796	2453	139	241	2073	14.6
DS-AB (Article Baz, 2013)	435,420	2627	916	680	1031	21

Set of Imam Reza International University (DS-IR) (Imam Reza International University, 2015) during a 2-week interval. The second data set comes from a 6-week long access log from ArticleBaz (DS-AB) (ArticleBaz, 2013), which is a Persian website providing free-to-access scientific articles in different fields for Persian students to download. As DS-IR contains information of a university with huge number of students, it has a larger number of visitors compared to DS-AB. We thus randomly sampled a subset of sessions from the DS-AB Web servers to ensure an evaluation of similar numbers of web sessions from each domain.

Table 2 offers a summary of the number of the robots and humans found across the two web servers by the simple heuristic approach from Fig. 2. It finds that both web servers serve a substantial number of requests (102,796 and 435,420 per month), but are composed of different proportions of robot sessions and session lengths. Moreover, the use of ArticleBaz as a repository for free download of scientific articles for Persian students attracts many web robot agents to select the site for investigation. Although an approximately similar number of sessions were extracted from both logs, because of security policies enforced by Imam Reza International University against some known robot visitors, only 1/4 of the number of robots on the DS-AB server (1586) were identified over DS-IR (380). The university filters resulted in the heuristic approach to find just 15.4% of all requests to be from robots on DS-IR, but 60% of all requests to be from robots on DS-AB. We also note that, while the session length average of DS-IR is about 14.6 min, this value is approximately equal to 21 min for DS-AB. In contrast to the university website, where humans may only be interested in finding specific information before leaving, visitors of ArticleBaz may spend more time searching and browsing articles. These descriptive statistics suggest that the DS-IR and DS-AB are web logs exhibit different characteristics, with one greatly preferred by web robots compared to the other. The two datasets thus allow us to exercise SMART against playbacks of real data where a web server is faced with varying levels of web robot traffic. Using the sessions identified from the heuristic approach as ground truth labels for human, benign, and malicious robot sessions, we next evaluate the performance of SMART over the logs.

4.2. FRS threshold evaluation

Recall that the hyper-parameter *FRS_thr* influences the features SMART uses to form the robot and human traffic clusters. We thus explore how different settings of *FRS_thr* affect the accuracy of the algorithm via the Rand Index (Amigó, Gonzalo, & Verdejo, 2013) and the total number of attributes considered in building the robot and human clusters. The Rand Index (*RI*) is a popular measure used to evaluate the clustering performance by reporting the proportion of correct detections to all identified results as shown in Eq. (13):

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

For this evaluation, we say that *any* a robot session labeled as *any* type of robot by SMART as a *true positive (TP)* session, and a human labeled session in the human behavior cluster as a *true negative (TN)* session. Human sessions labeled as a robot are called *false positive (FP)* sessions while robot labeled sessions in the human behavior cluster are called *false negative (FN)* sessions.

Table 3
Attributes chosen by SMART for the DS-AB and DS-IR web servers.

Data set	Final attributes	Frequency index
DS-AB	Night	100
	%NullReferrer	100
	TrapFileRequest	90
	MultiAgent	87
	MultiIP	70
	PPI	65
	%4xx	59
	%CSR	35
	HTML/Image	18
	SD-RPD	16
DS-IR	PPI	100
	%NullReferrer	100
	Night	87
	MultiIP	79
	%Other	74
	%CSR	56
	Depth	48
	SD-RPD	41
	%304	37

Fig. 4 plots the Rand Index and number of attributes of the SMART algorithm over the DS-AB dataset for various settings of *FRS_thr*. It shows that SMART attains the highest Rand Index when *FRS_thr* is set to ten, with ten features satisfying this threshold. The Rand Index decreases for larger threshold values as the number of attributes used for clustering reduces when the threshold is greater than or equal to twenty. Fig. 5 shows the results of the same analysis over the DS-IR dataset. We find that a threshold of thirty yields the best Rand Index. It is interesting to note that smaller thresholds on the DS-IR dataset would consider a larger number of session features for MCL, but would also yield a smaller Rand Index. This supports the idea that careful feature selection, encoded into SMART, is essential for the Web robot detection problem: simply using as many features as possible does not necessarily yield the strongest detection accuracy.

With $FRS\_thr = 10$ for DS-AB and $FRS\_thr = 30$ for DS-IR, Table 3 lists the ultimate set of attributes chosen and their corresponding frequency indices[1] chosen by SMART. The table identifies sets of common and unique features that distinguish robot and human visitors on the two servers. For example, as humans and web robots may utilize multiple IPs and user-agent strings in their requests, *MultiIP* and *MultiAgent* are relevant attributes for both web servers. Moreover, the percentage of requests sent in night hours may be an appropriate indicator for sessions of web robots, and as the navigational patterns of humans and web robots are noticeably different, attributes related to page depth (*%CSR, SD-RRPD*) can be useful. On the other hand, since DS-AB is an educational website which provides mostly HTML web pages with few images, it may only have a small number of visitors interested in viewing image

[1] A description of these attributes is listed in Table 1.

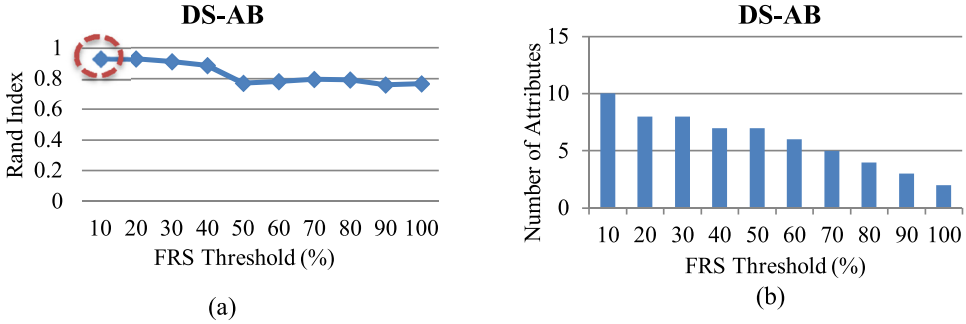

Fig. 4. Feature selection evaluation over DS-AB dataset (a) Rand Index; (b) number of attributes.

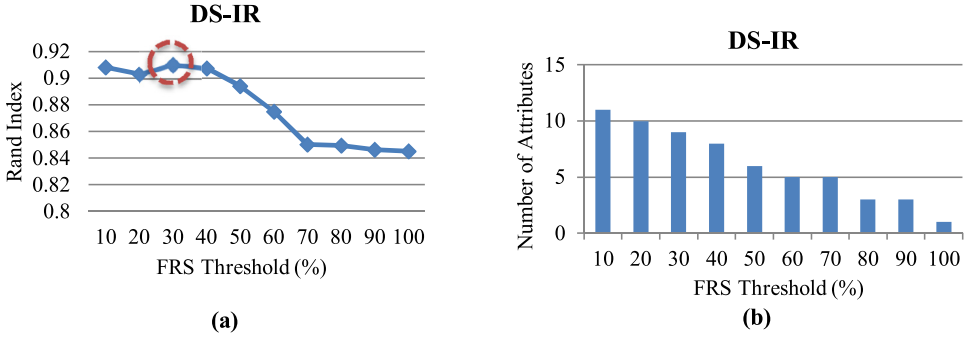

Fig. 5. Feature selection evaluation over DS-IR dataset (a) Rand Index; (b) number of attributes.

resources. This may explain why the feature *HTML/Image*, which measures HTML-to-Image ratio in a request session, is a significant attribute to separate robots from humans. Furthermore, while the percentage of 4xx response codes is an important feature on DS-AB, SMART finds the percentage of 304 responses is more relevant for DS-IR.

4.3. Clustering evaluation

We now examine the sets of sessions labeled as human, benign robot, and malicious robot by SMART and compare them against the clusters found by other leading soft detection methods. Specifically, we compare SMART against Stevanovic et al.'s method (Stevanovic et al., 2013), which we call NNRD (Unsupervised neural network for Web Robot Detection), and Zabihi et al.'s method (Zabihi et al., 2014a, 2014b), named DBC-WRD. NNRD combines the SOM and ART2 clustering algorithms to categorize visitors characterized by ten static attributes into four groups of humans, well-behaved crawlers, malicious crawlers, and unknown visitors. DBC-WRD cluster web sessions into humans and robots' classes by the DBSCAN clustering algorithm without considering the malicious robots. It then applies a *t*-test to select relevant attributes among those specified for each session.

We implement these two methods, as well as SMART, on the same machine with an Intel processor at 3.70 GHz with 4 GB of RAM. All three algorithms were implemented in MATLAB 8.3.0.532 (R2014a) where the *epsilon* and *minPoints* parameters of DBSCAN were set to the default values 0.9 and 6, respectively. The SOM was constructed with 100 neurons over a 10-by-10 hexagonal arrangement with 200 epochs. The supporting ART2 clustering algorithm had its inverse vigilance parameter, dynamic parameter, and requested number of output node parameters which were set to 1.2, 0.4, and 3 according to guidance by the method's authors. In the implementation of SMART, the expand and inflate parameters have been set to 2 based on the suggestions in Van Dongen (2001) and the pruning parameters (a and b) have been set to 1. In this evaluation, we first examine the TN, TP, FN, and FP rates of each method as explained in Section 4.1.

Table 4 lists the number of sessions classified as TP, FP, TN, and FN by the three approaches over each dataset. We see that, for both servers, SMART correctly classifies a session as a robot or human more often compared to DBC-WRD and NNRD. We also compare the Rand Index and, to compensate for biasness toward classifying sessions into the majority class (Kanji, 2006), also consider the Jaccard measure (Amigó et al., 2013) given by:

$$Jaccard = \frac{TP}{TP + FP + FN} \tag{14}$$

Note therefore that larger positive classifications into the robot's behavior cluster, and not positive classifications into the human's behavior class, influence the Jaccard measure.

According to Fig. 6(a), SMART achieves a superior Rand Index across both datasets. Moreover, although DBC-WRD algorithm utilizes *t*-test to choose final attributes, NNRD achieves a better Rand Index. As mentioned previously, the main reason of this difference is the poor performance of a *t*-test when data attributes are not normally distributed, like those used in this research (Kanji, 2006). Also, regardless of the data distribution a *t*-test is just based on comparing a summary statistic over the attribute of robot and human sessions, while SMART incorporates the similarity of all attributes simultaneously when making a feature selection decision. Therefore, the key reason that SMART improves on the performance of DBC-WRD may be through our feature selection strategy, which employs the more expressive theory of FRS. On the other hand, Fig. 6(b) finds that SMART achieves a larger Jaccard measure compared to the other approaches, and hence, is able to better segregate robot's behaviors from the human's ones compared to NNRD and DBC-WRD. Because SMART performs dynamic feature selection, the superior Jaccard measure may offer further support that FRS is a more promising approach to perform feature selection compared to *t*-testing.

Table 4
Confusion matrices of humans versus web robots gained by each compared algorithm on each data set.

Data sets	Methods	Recognized as Humans		Recognized as web robots	
		Correctly (*TN*)	Incorrectly (*FN*)	Correctly (*TP*)	Incorrectly (*FP*)
DS-AB	SMART	969	117	1479	62
	DBC-WRD	58	152	1444	973
	NNRD	937	118	1478	94
DS-IR	SMART	2055	202	178	18
	DBC-WRD	2032	284	96	41
	NNRD	2055	254	126	18

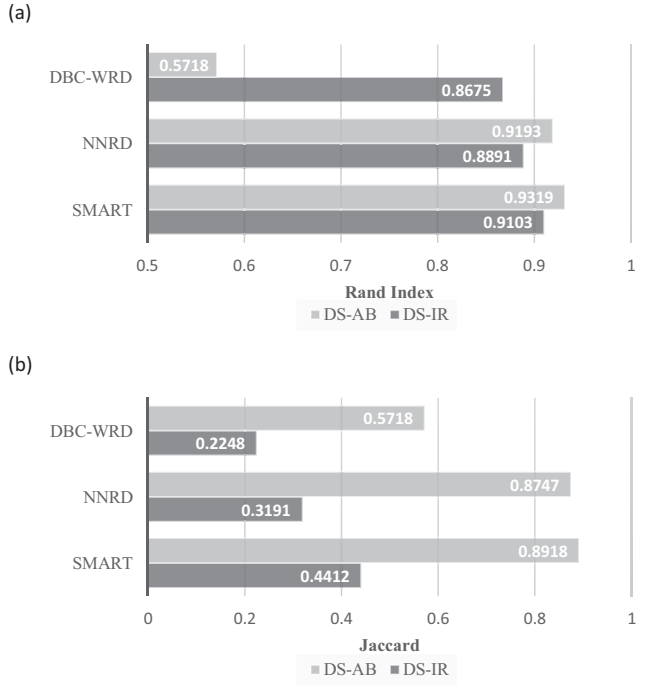

Fig. 6. Comparison of evaluation metrics of SMART and state-of-the-art algorithms (a) Rand Index (b) Jaccard.

4.4. Scrutinizing malicious and benign robot clusters

Next, we evaluate the benign and malicious agents unearthed by SMART. Since DBC-WRD does not try to capture malicious robots, we only compare the types of sessions discovered in clusters of malicious robots derived by SMART and NNRD. In Figs. 7 and 8, the numbers and percentages of humans could indicate false positives, or may be malicious robots who change their user-agent string into human ones to mask their identity. However, the numbers and percentages of benign robots definitively summarize the miscategorization rates of the detection algorithm in separating benign and malicious robots. The figures demonstrate not only that SMART better distinguishes between benign and malicious robots, but also that its potential false positive rates are lower compared to NNRD. For example, SMART over DS-AB only finds 10 human sessions compared to 25 found by NNRD, and for both web servers, correctly labels a larger number of malicious web robot sessions.

Fig. 8 repeats the above analysis, but this time for sessions falling into the benign robot cluster. It shows how SMART labels a greater proportion of sessions in this cluster as benign, even if it places fewer sessions into the benign robot cluster compared to NNRD on the DS-IR web server. However, a greater number, and proportion, of benign robots are correctly identified on DS-AB. Finally, irrespective of the number of sessions falling into the benign robot cluster, SMART misclassifies a smaller number and lower proportion of sessions.

In summary, our comparison against state-of-the-art methods demonstrate that SMART offers better performance in distinguishing between humans and web robots. In particular, measurements with the Jaccard measure demonstrate that SMART is able to more accurately identify true web robot sessions. Further scrutiny of whether the detected robots are malicious or benign, as demonstrated in Figs. 7 and 8, shows further improvements by SMART. There are thus multiple implications for the use of SMART in practical systems. For example, the ability of SMART to find and label web robots as “malicious” allows web server administrators to block access by robots who are known to be visiting purely to cause harm, while still allowing benign robots who are visiting for indexing resources, and perhaps, to make the web site more visible to search engines and web site aggregation tools. Filtering out malicious web robots thus allows robots to fulfill their important role in the web ecosystem while still protecting a web server and its content from information retrieval campaigns launched by harmful actors.

5. Conclusion and future work

This article introduced a novel approach, called SMART, to identify human, benign robot, and malicious web robot traffic on a web server. Through a pipeline of soft computing approaches for feature selection and session labeling, SMART achieves accurate detection of benign and malicious web robot traffic with features tailored to any specific web server. An experimental evaluation compared SMART against other leading soft computing algorithms for this task, using data from some live web servers for different domains of the internet, and demonstrated the variety of features considered. SMART gives state-of-the-art performance in identifying benign and malicious web robot sessions on a web server.

There are several directions for future work. One plan is to investigate alternative methods and heuristics for labeling training data to ensure that an even larger proportion of malicious robot sessions are considered. For example, there may be malicious robots that do not access *robots.txt* to discover directories an administrator does not want crawled. Therefore, running the trained detectors on sessions that never requested *robots.txt*, and examining those sessions it determines are malicious robots may be insightful to develop new heuristics for labeling robots as malicious. Another avenue for future work is to begin implementation of SMART as a plug-in for Apache or the nginx load balancing systems so as to begin actively protecting Web systems from malicious Web robot traffic. This undertaking, which will be a multifaceted software engineering task, will be thoroughly tested and released as an open source plug-in for Apache (and possibly IIS) in the future. Another future plan is to utilize evolutionary algorithms to tune the hyper-parameters of SMART that govern the FRS feature selection and MCL clustering. Some kinds of evolutionary algorithms already show promise, such as the Bat Algo-

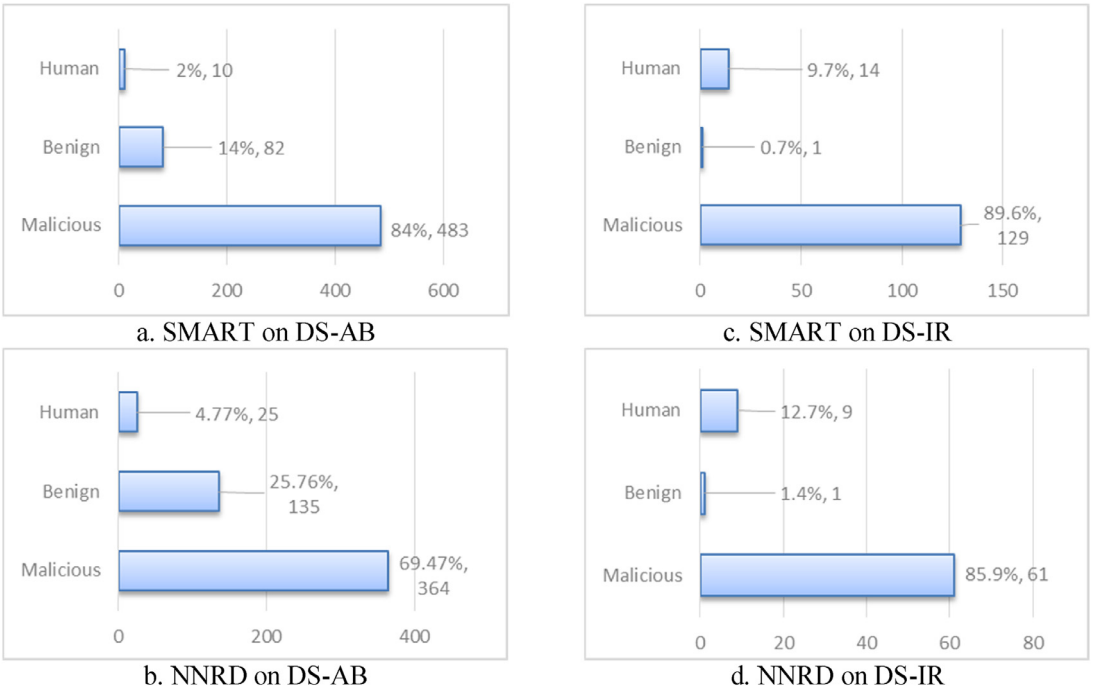

Fig. 7. The number and percentage of true session types found in robot clusters labeled "malicious" for both algorithms and servers.

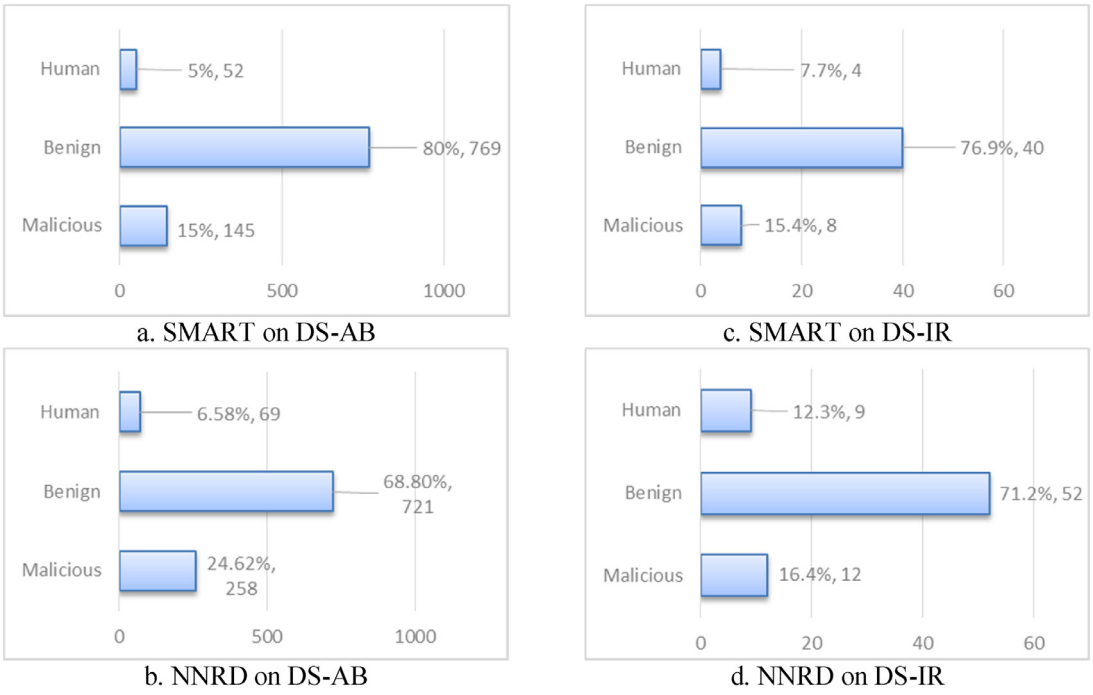

Fig. 8. The numbers and percentages of each session type found in the benign robots' clusters for both algorithms.

rithm that can successfully choose hyper-parameters in other expert systems (Sadeghi & Hamidzadeh, 2016). Another direction is to explore the use of *online enhancing* of the background knowledge used by SMART in its FRS and MCL steps. Online updating of new information, as additional sessions are made in real-time, may lead to the replacement of some features or the use of new MCL parameter settings. In fact, this online component may be considered as a hybrid system in practical settings that cannot only cluster sessions offline to collect visitor statistics and present them to website administrators, but also allow SMART to adapt to changes in the characteristics of sessions.

Finally, we may explore the noting of incorporating *semantic* features of the resources requested by web robots and humans as features considered by FRS. Semantic features are interesting in that they may allow us to differentiate web robots and humans per the type of information that is encoded in the resources. Some predictive power may exist in such features since humans may target a website to find resources that contain some specific kind of in-

formation they are after, while web robots may crawl without care for the information the resources they scrape encode.

Acknowledgment

This paper is based on work supported by the National Science Foundation (NSF) under Grant no. 1464104. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

Incapsula, 2015 bot traffic report: Humans take back the web, bad bots not giving any ground. (Dec. 2015). https://www.incapsula.com/blog/bot-traffic-report-2015.html.

Amigó, E., Gonzalo, J., & Verdejo, F. (2013). A general evaluation measure for document organization tasks. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 643–652). ACM.

Article Baz. (Nov. 2013). http://www.articlebaz.com.

Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer Networks, 54*(15), 2787–2805.

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

Dikaiakos, M. D., Stassopoulou, A., & Papageorgiou, L. (2005). An investigation of web crawler behavior: Characterization and metrics. *Computer Communications, 28*(8), 880–897.

Doran, D., & Gokhale, S. S. (2009). Classifying web robots by K-means clustering. In *Proceedings of the international conference on software engineering and knowledge engineering* (pp. 97–102).

Doran, D., & Gokhale, S. S. (2011). Web robot detection techniques: Overview and limitations. *Data Mining and Knowledge Discovery, 22*(1–2), 183–210.

Doran, D., Morillo, K., & Gokhale, S. S. (2013). A comparison of web robot and human requests. In *Proceedings of the 2013 IEEE/ACM international conference on advances in social networks analysis and mining* (pp. 1374–1380). ACM.

Dubois, D., & Prade, H. (1990). Rough fuzzy sets and fuzzy rough sets. *International Journal of General System, 17*(2–3), 191–209.

Dutta, A., Meilicke, C., & Stuckenschmidt, H. (2015). Enriching structured knowledge with open information. In *Proceedings of the 24th international conference on world wide web* (pp. 267–277). ACM.

Gržinić, T., Mršić, L., & Šaban, J. (2015). Lino – An intelligent system for detecting malicious web-robots. In *Proceedings of Asian conference on intelligent information and database systems* (pp. 559–568). Springer International Publishing.

Heinz, E., Selkrig, J., Belousoff, M., & Lithgow, T. (2015). Evolution of the translocation and assembly module (TAM). *Genome Biology and Evolution, 7*(6), 1628–1643.

Huang, Z., Xu, Z., & Lu, Q. (2016). Some new inequalities for the Hadamard product of a nonsingular M-matrix and its inverse. *Linear and Multilinear Algebra, 64*(7), 1362–1378.

Imam Reza International University. (Aug. 2015). http://www.imamreza.ac.ir.

Jain, Y. K., & Bhandare, S. K. (2011). Min max normalization based data perturbation method for privacy protection. *International Journal of Computer & Communication Technology, 2*(8), 45–50.

Kanji, G. K. (2006). *100 statistical tests*. Sage.

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: Vol. 4* (pp. 43–48). Springer-Verlag.

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

Liao, H., & Xu, Z. (2015). Approaches to manage hesitant fuzzy linguistic information based on the cosine distance and similarity measures for HFLTSs and their application in qualitative decision making. *Expert Systems with Applications, 42*(12), 5328–5336.

Lourenço, A. G., & Belo, O. O. (2006). Catching web crawlers in the act. In *Proceedings of the 6th international conference on web engineering: Vol. 263* (pp. 265–272). ACM.

Parthasarathy, S., Ruan, Y., & Satuluri, V. (2011). Community discovery in social networks: Applications, methods and emerging trends. In *Social network data analytics* (pp. 79–113). Springer.

Pawlak, Z. (1982). Rough sets. *International Journal of Computer & Information Sciences, 11*(5), 341–356.

Qian, Y., Wang, Q., Cheng, H., Liang, J., & Dang, C. (2015). Fuzzy-rough feature selection accelerator. *Fuzzy Sets and Systems, 258*, 61–78.

Radzikowska, A. M., & Kerre, E. E. (2002). A comparative study of fuzzy rough sets. *Fuzzy Sets and Systems, 126*(2), 137–155.

Ruan, Y., Fuhry, D., & Parthasarathy, S. (2013). Efficient community detection in large networks using content and links. In *Proceedings of the 22nd international conference on world wide web* (pp. 1089–1098). ACM.

Rude, H. N., & Doran, D. (2015). Request type prediction for web robot and internet of things traffic. In *Proceedings of 2015 IEEE 14th international conference on machine learning and applications* (pp. 995–1000). IEEE.

Sadeghi, R., & Hamidzadeh, J. (2016). Automatic support vector data description. *Journal of Soft Computing*, 1–12. https://doi.org/10.1007/s00500-016-2317-5.

Satuluri, V., Parthasarathy, S., & Ucar, D. (2010). Markov clustering of protein interaction networks with improved balance and scalability. In *Proceedings of the first ACM international conference on bioinformatics and computational biology* (pp. 247–256). ACM.

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, 3*(2), 1–10.

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

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

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

Suchacka, G., & Sobkow, M. (2015). Detection of Internet robots using a Bayesian approach. In *Proceedings of 2015 IEEE 2nd international conference on cybernetics* (pp. 365–370). IEEE.

Sun, Y., Councill, I. G., & Giles, C. L. (2010). The ethicality of web crawlers. In *Proceedings of 2010 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology: Vol. 1* (pp. 668–675). IEEE Computer Society.

Szilágyi, L., Medvés, L., & Szilágyi, S. M. (2010). A modified Markov clustering approach to unsupervised classification of protein sequences. *Neurocomputing, 73*(13), 2332–2345.

Tan, P.-N., & Kumar, V. (2002). Discovery of web robot sessions based on their navigational patterns. *Data Mining and Knowledge Discovery, 6*(1), 9–35.

Van Dongen, S. M. (2001). *Graph clustering by flow simulation*. The Netherlands: University of Utrecht.

Verbiest, N., Cornelis, C., & Herrera, F. (2013). OWA-FRPS: A prototype selection method based on ordered weighted average fuzzy rough set theory. In *Proceedings of international workshop on rough sets, fuzzy sets, data mining, and granular computing: Vol. 8170* (pp. 180–190).

Zabihi, M., Jahan, M. V., & Hamidzadeh, J. (2014a). A density based clustering approach for web robot detection. In *Proceedings of 2014 4th international e-conference on computer and knowledge engineering* (pp. 23–28). IEEE.

Zabihi, M., Jahan, M. V., & Hamidzadeh, J. (2014b). A density based clustering approach to distinguish between web robot and human requests to a web server. *The ISC International Journal of Information Security, 6*(1), 77–89.