Available online at www.sciencedirect.com

ScienceDirect

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

Computers & Security

An improved rule induction based denial of service attacks classification model

Rami Mustafa A. Mohammad[a,*], *Mutasem K. Alsmadi*[b], *Ibrahim Almarashdeh*[b], *Malek Alzaqebah*[c,d]

[a] *Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam, Saudi Arabia*
[b] *Department of MIS, College of Applied Studies and Community Service, Imam Abdulrahman Bin Faisal University, P.O.Box 1982, Dammam, Saudi Arabia*
[c] *Department of Mathematics, College of Science, Imam Abdulrahman Bin Faisal University, 31441, Dammam, Saudi Arabia*
[d] *Basic and Applied Scientific Research Center, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, 31441, Dammam, Saudi Arabia*

ARTICLE INFO

Article history:
Received 9 June 2020
Revised 1 August 2020
Accepted 20 August 2020
Available online 2 September 2020

Keywords:
Denial of service
Quality of service
Rule induction
Intrusion detection
Cloud computing
Classification

ABSTRACT

For assessing the quality of any internet and cloud computing services; accessibility is presumed a significant factor among other Quality of Service (QoS) factors. Distributed Denial of Service attack (DDoS) is considered a significant threat pertaining to all contemporary and emerging online-based services. Intelligent solutions centered on the utilization of data mining methods are looming on the horizon as possible solutions to counter this kind of attacks. Rule Induction (RI), which is a well-known data mining method is regarded as a possible approach for developing an intelligent DDoS detection system. The current article offers an "Improved RI algorithm" (*IRI*) which decreases the searching space for generating classification rules by removing all unimportant candidate rule-items along the way of creating the classification model. The main advantage of *IRI* is producing a group of rules that can be described as concise, easy to understand, and easy-to-implement. In addition, the classifiers generated by *IRI* are more compact in size which is heavily weighted when producing any classification system. The proposed algorithm is then applied for detecting DDoS attacks (IRIDOS). Empirical evaluations using the UNSW-NB15 dataset that has been obtained from the University of New South Wales confirmed the robustness of IRIDOS.

1. Introduction and background

The Denial of Service (DoS) attacks overwhelm networks infrastructure as well as online and cloud services by making use of a distributed set of malevolent and infected computers to carry out harmful actions. These harmful actions are mainly designed to damage the Information Technology (IT) infrastructure of a corporation. Even worse, it might also be designed to hurt the IT infrastructure of some public and governmental services which were essentially created to facilitate the people lifestyle. Normally, DDoS attacks may affect the accessibility and the availability of the targeted online service because of processing capacity overburden. The 22nd of July

* Corresponding author.
E-mail addresses: rmmohammad@iau.edu.sa (R.M.A. Mohammad), mkalsmadi@iau.edu.sa (M.K. Alsmadi), iaalmarashdeh@iau.edu.sa (I. Almarashdeh), maafehaid@iau.edu.sa (M. Alzaqebah).
https://doi.org/10.1016/j.cose.2020.102008
0167-4048/

1999 witnessed the first DoS attack at the University of Minnesota where a network of more than 100 computers infected with malware called "Trin00" were used for attacking one of the main computers at the university (Osterweil et al., 2019). One year later, several commercial online service providers including eBay, CNN and Yahoo experienced their first DoS attempts. In general, there are several motivations for this type of attack including: Cyber Warfare, Intellectual Challenge, Ideological Belief, Revenge, and Financial/Economic Gain.

DDoS attacks are to some extent easy to launch and at same time happens to be hard to track the real opponents. By running DDoS attacks, opponent utilizes a network of compromised machines for consuming the resources of the targeted system and thus the honest clients won't be able to access or use the services offered by the targeted system or organization. In general, DoS attacks aims to consume the network or the computing resources by launching a high-rate flooding (HRF) attacks towards the victim. There are several types of HRF arracks including Buffer Overflow (Pierson, 2020), ping of death (Feng et al., 2020), TCP SYN floods (Sahi et al., 2017), UDP flood (Sahi et al., 2017), Smurf (Visalatchi and Yazhini, 2020), Neptune (Gupta, 2020), and Teardrop (Feng et al., 2020). A DoS attack consists of 4 components namely: the real opponent who initiates the attack, the attacked servers, the packets produced by zombie bots (commonly known as botnet), and the targeted system. A botnet (*which is a set of malevolent machines taking part in the attack*) is started in order to overwhelm an online server by using asynchronous attacks which make the services offered by that server inaccessible to end users as a result of an exhaustion of its network or computing resources. Nowadays, probably the most effective botnets are usually make use of Internet of Things (IoT) equipment, considering that huge numbers of vulnerable IoT equipment happen to be connected and deployed and a lot of these equipment can be easily compromised and hacked (Khan and Salah, 2018). The systems targeted by DDoS can be for instance: business websites, web servers, firewalls, intrusion detection systems (IDS), Internet infrastructure such as Domain Name Server (DNS) protocol, and perhaps vitalized infrastructure including cloud services. Cloud computing is a collection of computing resources which are tailored to constantly offer upon-request services to end users and hence ensure the availability of services despite the huge workload flowing into the system. However, cloud computing systems are plagued by DDoS attacks. One method for conducting a DDoS attack in cloud systems is by intentionally sending large amount of authentication requests which will consume the computing resource of the targeted cloud system and consequently the response time for the genuine requests will increase causing honest clients unsatisfied by the QoS and may even switch to other service provider(s). Regarding the Internet of Things (IoT), DDoS have an exceptional opportunity to use IoT equipment to start DDoS attacks. Aside from using this equipment to start DDoS attacks, the accessibility and the availability of these equipment might also be affected. Hence, resulting in a devastating impact especially when it comes to equipment which playing major role on clients' day-to-day activities or being utilized in smart cities. DDoS attacks do not merely harm the clients but also have remarkable consequences on both private and public online service providers. For instance, in addition to harming its reputation; the targeted party might lose a substantial amount of money and time. Nevertheless, the DDoS attacks can result in further costs in the sense that the service providers should pay extra money because of the omission of their services along with some possible legal consequences according to the service level agreements (SLA). The scenario could be more disastrous when considering the services offered by the health-care providers due to the fact that failure in these systems cannot be forgiven for whatever reason. A new report published in 2019 (Pinson-Roxburgh, 2019) showed that DDoS attacks could cost more than $120000 in small companies or even over $2000000 for some enterprise companies. This large amount of money can be attributed due to apologies to clients and shareholders. Global estimations of the total number of DDoS attempts is projected more than 14 million attempts in 2022 based on 2017 statistics disseminated by the "Cisco Visual Networking Index" (VNI) (Crane, 2019). The second quarter of 2019 showed more DDoS attempts compared to the prior quarter of the same year. The U.S. and China ranked as being the main targets of DDoS in the 2nd quarter of 2019 with more than 60% and 17% of the attacks respectively (Kupreev et al., 2020). Imperva (*a major cybersecurity software and services provider*) revealed that the biggest application layer DDoS attack in history took place in spring of 2019 for more than 10 days (Simonovich, 2019). The attack has targeted streaming service clients; and reached up to more than 290000 requests for every second.

The nature of DoS is constantly changing as time passes because attackers adapt their attacking techniques to avoid current detection systems. Yet, DoS attacks are becoming more sophisticated than ever. Hence, developing intelligent DoS protection systems that are capable to recognize DoS attacks accurately with low false positive rates is an important and a timely issue. Therefore, different DDoS mitigation techniques were proposed in literature. Amongst others, Data Mining (DM) and Machine Learning (ML) methods are considered viable approaches for mitigating such an attack. DM can be viewed as a successful solution to simplify producing practically beneficial knowledge to make appropriate decision. DM or commonly referred to as "*Knowledge Discovery*" (Mohammad and Alqahtani, 2019, Mohammad, 2018) pertains to analyzing a training datasets from various perspectives, thereafter presenting them in sensible as well as practicable manner. Classification is a broadly researched technique in the DM. Classification is regarded as a frequently investigated strategy in DM community. Classification is generally thought as the task of building an intelligent model by making use of the historical datasets to finally forecast the label of a class variable linked to a hidden example (Mohammad, 2020). Classification can be described as a supervised learning method because each example in the training dataset employed for developing the classification model is associated with a value commonly called a class value. In addition to others, Rule Induction (RI) is amongst the approaches which usually come under the umbrella of the supervised classification DM strategy. Normally, RI techniques deliver a key aspect compared to other widely used classification techniques in the sense that RI methods have the capability to generate relatively simple rules which can be easily comprehended and also can be manually implemented by the classification model designer. Fur-

thermore, RI have the ability to learn extra beneficial hidden knowledge that occasionally missed by various classification strategies; therefore, the classification precision of the produced classifier might be improved.

PRISM is considered one of the most commonly used RI approaches that has been introduced in (Cendrowska, 1987) and later improved by other scholars such as the work done in (Almutairi et al., 2017). PRISM makes use of divide and conquer technique in knowledge discovery where it produces rule set depending on the labels shown in the class variable within training dataset(s). Typically, PRISM begins with a blank rule and continues attaching item(s) to the body of the rule till the rule achieves zero errors. As soon as this happens, a rule will be generated, and the dataset examples associated with the rule will be tossed out. PRISM keeps generating various other rules in a similar manner till no further data associated with the on-hand class value is available. In this case, the same strategy is carried out for the following class value till the training dataset ends up being empty.

Probably the most the obvious issue linked with PRISM is the large number of generated rules, that usually provides large-size classification model. This issue is caused by the manner PRISM generates the rules because it continues appending item(s) to the body of the generated rule until each rule gets to be a 100 % accurate regardless of the minimal data coverage. This basically means, PRISM doesn't mind generating lots of rules although some of them are covering only one dataset example, instead of generating a rule that might has 90% accuracy and covering more than one data examples. Such extreme learning strategy hinders the capability of PRISM to be an effective decision-making approach and probably overfits the training dataset(s).

In the present research, an innovative learning algorithm based on RI is created and is used for detecting DDoS attacks. The proposed algorithm finds out the rule gradually for each class value and mainly utilizes a frequency threshold for reducing the searching space for rules by removing several items having nonsufficient data representation. For every single rule, the suggested algorithm revises items frequency which emerged in the removed examples of the produced rule. That certainly generates a more practical classification model that has reduced number of rules resulting to a built-in rule pruning through the rule generation phase. The suggested algorithm restricts the utilization of the default class rules by producing rule(s) that have accuracy less than 100%. Typically, such rules are neglected by PRISM simply since they do not have zero errors. Such rules can be employed through class forecasting stage rather than the default class rules and if no 100% accuracy rules are capable to forecast the class of a testing instance.

The next section of this article explores several intelligent techniques used for detecting DDoS attacks. The proposed algorithm is thoroughly explained in Section 3. Section 4 presents a practical example of the proposed algorithm. Two sets of experiments will be discussed in Section 5. The first set intends to assess the generalization capability of the proposed algorithm using several benchmark datasets. However, in the second set of experiments, the algorithm is used for detecting DoS attacks. Finally, the conclusion and future work are discussed in Section 6.

2. DDoS mitigation techniques

Nowadays, several trending technologies were used for mitigating DoS attacks. Among others, Cloud-based, DM and ML approaches are commonly used methods for mitigating DoS attacks. This section starts by reviewing several earlier studies that addressed DoS attacks using cloud-based techniques. Further, intelligent methods based on DM and ML algorithms for detecting DoS attacks are discussed thoroughly.

2.1. *Cloud-based approach*

Cloud computing facilitates delivering various resources and services over the Internet including hardware and software such as networking, storage, virtualization, servers, databases, and data centers. Cloud computing is a preferred choice for individuals and companies for several reasons such as reducing spending, increasing productivity, efficiency, and security. A lot of security services including anti-virus, anti-spam, DoS detection systems, and intrusion detection systems, might be provided as cloud services (Salah et al., 2013). Such services (that are usually structured as transparent solutions using an overlay network or endpoint services) can offer protection to both physical and virtual infrastructure. Cloud-based security systems have benefited from the exceptional storage and computation capacities offered by cloud providers to further update the security system as needed. This in fact was very useful for systems dealing with the dynamic nature of some cyberattacks. Typically, in cloud-based security systems the network's inbound traffic is redirected to cloud where a filtering sub-system examines the traffic before forwarding it back to the destination network. The authors in (Salah et al., 2013) designed an elastic cloud-based overly network architecture which grows on demand and uses cloud computing resources offered by the cloud provider. Such an architecture offers a collection of security solutions including anti-virus, anti-spam, and DDoS detection systems. This model assumes the availability of a secure cloud computing environment. A front end management center offers the applications to control those solutions, including Security Event Manager (SEM), Security Policies, and an Single-Sign-on-Proxy (SSO). Other security methods might simply be introduced as a result of the native cloud scalability. The model suggests placing a load balancer at the network input point. Hence, the incoming traffic is restricted merely to traffic received from the security overlay network. Later in 2015, a cloud-based firewalling system was suggested (Guenane et al., 2015). The system was an outsourcing of firewalling services that making use of the advanced accessibility, wide and powerful resource offered in the cloud to handle DoS attacks. The system consisted of three key components these are: Back-Gateway, Virtual firewall instances, and Front-Gateway. In (Du and Nakao, 2010), a cloud-based DoS detection system is proposed for blocking attacks targeting web servers. It is essentially a distributed system which works on cloud infrastructure as a protection overlay network to secure web servers by employing a collection of smart transparent and collaborative web proxy servers. In (Yassin et al., 2012), an interesting system centered on cloud technology referred to as Cloud-based Intru-

sion Detection Service (CBIDS) was introduced. The system is based on the principles of Software as a Service (SaaS) model. Such a system facilitates the detection of malevolent actions from various areas in a network and address the deficiencies of traditional IDS. The proposed system might be applied to recognize several kinds of attacks in public and private clouds.

Nevertheless, the cloud-based DoS detection systems might not be appropriate for IoT and Industrial IoT environments (Further information about the IIoT can be found in (Khan et al., 2020)) in case the attack traffic is originating from inside the local network . In addition, the outsourcing strategy might not instantly detect and prevent the attacks close to the attacking sources (Khan and Salah, 2018). Most importantly, cloud scalability should be taken into account when building any cloud-based security system in general and DDoS attack in particular due to the fact huge amount of traffic might be generated by the attackers (Khan and Salah, 2018, Al-Haidari et al., 2013). The research study offered in (Al-Haidari et al., 2013) is one of the interesting researches that investigated the effect of adjusting the CPU's upper threshold as well as the scaling size variables on the ability of the cloud services. Such a study concluded that input loads is an important factor for tuning such parameters. Therefore, if cloud scalability is not considered there will be no guarantee that the cloud-based DoS mitigation system will be able to offer appropriate load balancing of inbound service requests. The authors in (Yu et al., 2014), suggested a system which was designed primarily to fight against DOS attacks. The system is considering the dynamic resources allocation process which allocates additional resources provided by cloud resources and more virtual machines (VM) are going to be cloned depending on image files of the master intrusion system. As soon as the amount of DOS attack packets goes down, the system is going to minimize the number of cloned VMs and releases any additional resource. An innovative analytical model proposed in (Salah, 2013), has also aimed at investigating how to achieve appropriate elasticity for cloud-based services. The model is essentially using queueing theory to specify at any specific period of time and workload circumstances the minimal set of computing resources required for carrying out a huge number of parallelized tasks within a cloud cluster. The queuing principle has also applied for analyzing the performance of fog computing (Said and Salah, 2017). Fog computing were also used for building DoS detection systems. For instance, the model proposed in (Bhardwaj et al., 2018), is an example of such systems. Such a system consists of three elements: edge applications which are designed to extract the useful information from traffic; the locally generated traffic where data is delivered to specific web service using a quick path; and in the web service the traffic information is collected and used to detect DoS attacks. In order to minimize the response time, a new system proposed in (Choi et al., 2018) which is also based on fog computing technology. The suggested system is using the oneM2M standard and it includes three subsystems these are: data manager, monitoring manager, and device manager.

There are several commercial cloud-based DoS detection systems such as the system offered by Imperva Imperva (2002), McAfee (1987), and Netscout (1984).

2.2. DM and ML approach

DDoS is deemed a typical binary classification problem in the sense that the network traffic can be classified as either an attack or a normal one. The most commonly used intelligent DM and ML approaches that might be used for detecting DDoS attacks are Support Vector Machine (SVM), K-Nearest Neighborhood (KNN), Random Forest (RF), Naive Bayes (NB), Neural Network (NN), and Decision Trees (DT).

Throughout their research study, the authors in (Siaterlis and Maglaris, 2005) explored the applicability of Multi-Layer Perceptron (MLP) in detecting DDoS attacks. For collecting the training dataset, a Gigabit Ethernet connection between an academic institute and a service provider is established for collecting packets from web traffic, peer to peer applications, as well as video and audio traffic. In the feature selection phase, the researchers used two attribute sets as input features for building the classification model. The first attribute set is together from capturing packets instantly, and the second attribute set is collected from analyzing the net flow data. The later feature set can be described as a higher-level profile of a stream traffic and it includes several features such as the destination and the source ports, the IP address of the destination and the source, and the used protocol. After the two set of input features are gathered, a MLP model was created for classifying inbound packets as normal or a DDoS attack attempt. The overall performance of the suggested model was evaluated depending on two evaluation metrics those are, False Positive Rate (FPR), and True Positive Rate (TPR) and the achieved results were very promising.

One interesting study proposed a DDoS attacks recognition model to locate the attackers' locations using DT and traffic-flow pattern-matching (Wu et al., 2009). In (Braga et al., 2010), the (Self-organizing map) SOM algorithm is employed for detecting DoS attack by inspecting the flow statistics pertained to DoS attempts. This technique showed high detection rate. Yet, one drawback of this technique is the fact that the detection has some delay and the attack is not recognized accurately and timely.

The research in (Subbulakshmi et al., 2011) aimed at detecting DDoS attacks with the help of an improved SVM algorithm. A 2-level hybrid strategy which includes 2 anomaly detection systems as well as 1 misuse recognition system had been considered. In (Gupta et al., 2013) the researchers offered a rule-based DoS attack detection model. Such a model used unsupervised learning approach and BN for detecting suspicious traffic with the aim of safeguarding cloud network from several kinds of including DDoS flooding attacks.

Backscatter traffic can be described as a complication of spoofed DDoS attack. The victims reply to the spoofed packets like it typically would, and therefore traffic that resulted from such responses is called backscatter traffic. Backscatter analysis is the process of inspecting backscatter packets that arrive at a statistically significant portion of the internet protocol address space to figure out attributes of victims and DDoS attacks. One interesting study that uses backscatter traffic is conducted in (Balkanli et al., 2014). Classification and Regression Tree (CART) algorithm as well as NB were used to create intelligent models using a dataset extracted from backscat-

ter traffic and it included several features such as delta time, packet length, time to live, SYN flag, protocol, and the IP address for both the destination and the source. A considerable portion of the dataset consisted of TCP packets whereas the remaining portion of the dataset was consisted of ICMP requests from several ports. Classification accuracy had been used for evaluating the generated classification models and the results were very promising and stimulated doing further studies to confirm the applicability of DM and ML techniques in detecting DDoS attacks.

An intelligent intrusion detection model for detecting DDoS attacks by analyzing inbound network traffic using SVM had been introduced in (Kato and Klyuev, 2014). Radial basis kernel function was used for building the SVM model. The training dataset was obtained from CAIDA (Andersen et al., 2019) and it included around one hour of anonymized bi-directional traffic records from and to victim machines. Several features were used in this study including Total number of bytes for every single IP address, Time interval, Total number of packets, Number of bytes per second, Number of packets per second, Mean packet size, and Total number of bytes. Three datasets had been produced by changing the used attributes in every dataset. Features 1, 2, and 3 were used for creating the first dataset. In the second dataset, features 2, 4, and 5 were used. Yet, for the last dataset, features 2, 3, 5, 6, and 7 were used. The generated datasets have been divided into testing and training datasets and different splitting ratios were considered. Subsequently, several experiments were conducted, and the overall performance was evaluated using the common evaluation metrics obtained from the confusion matrix.

In (Aljumah and Ahamad, 2016), a model for identifying DDoS attempts was created and it was essentially used NN and Chaos theory. Lyapunov coefficient was used to differentiate normal from a DDoS attack attempt and the obtained detection accuracy reached up to 99%.

A NN model was also proposed in (Saied et al., 2016). The researchers employed 3 particular topologies of the MLP with the aim of predicting 3 categories of DDoS attempts depending on background protocol utilized to carry out each attack including ICMP, UDP and TCP. The proposed model was able to accurately detects zero-day attacks. However, one main drawback of the proposed model is its high resource requirement.

A dataset that contain advanced DDoS attacks including SQL injection DDoS attacks (SIDDoS) as well as HTTP Flood, had been obtained in several network layers. After that, NB, RF, and MLP were used for building intelligent DDoS attacks detection system (Alkasassbeh et al., 2016). A research study that compared the performance of several DM and ML algorithms including SVM, NB, DT, and MLP in detecting DDoS is conducted in (Sofi et al., 2017). Due to the fact that there is no reliable dataset that covers contemporary DDoS attack such as SIDDoS, a new dataset had been collected. Among others, MLP achieved the highest detection accuracy.

The utilization of Software Defined Network (SDN) is attracting security researchers for exploring its capabilities of providing advanced security schemes particularly for IoT systems. In (Bawany et al., 2017), a model is suggested for large-scale network systems like smart cities for DoS attacks mitigation and detection using the SDN architecture. In (Lim et al., 2015), the researchers suggested a scheduling-based framework for the SDN controller which results in a practical DoS attacks detection strategy. In a research study conducted in (Vizváry and Vykopal, 2014), described the implementation, recognition and mitigation of DoS attacks within the SDN environment. The research study in (Dayal et al., 2017), described current approaches for detecting DoS attacks using SDN and grouped them into 3 categories: statistical-based techniques, rule-based techniques, and ML approaches. An interesting method proposed in (Mousavi and St-Hilaire, 2015) suggested an entropy-based for mitigating DoS attacks. Rule-based methods collect the attributes of various DoS attacks and after that exchange such attributes at the flow table to prevent DoS attacks (Wei and Fung, 2015). The main advantage of rule-based techniques is the fact that they have reasonably excellent accuracies, however, the main issue with this approach is that attributes should be re-identified and extracted whenever emerging attack strategies arise. In (Ye et al., 2018), a dataset collected from SDN controller was used for building a SVM model for detecting a DDoS attacks. SDN can be defined as a network architecture technique that facilitates employing software applications for intelligently and centrally controlling or programming the network. Several features were used in this study including source port speed, source IP speed, flow bytes deviation, flow entry speed, pair-flow ratio, and standard deviation for flow packets.

In (Cheng et al., 2018) an adaptive DDoS attack recognition system using multiple kernel learning technique is suggested. The ensemble learning strategy is employed for building such a system. The empirical evaluation revealed that such a system was able to accurately detect zero-day DDoS attacks. A study that compared the applicability of several DM and ML approaches in detecting DDoS attacks was carried out in (Doshi et al., 2018). Five algorithms were considered in this research these are KNN, DT, RF, SVM, and NN. The obtained results showed that SVM obtained the worst prediction accuracy. The researchers justified such a result due to the fact the dataset was not linearly separable. On the other hand, all other classification algorithms achieved high detection accuracy that reached up to 99%.

Various Deep Learning (DL) algorithms, including Gated Recurrent Units Neural Network (GRUNN), Long-Short Term Memory (LSTM), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), and the Principal Component Analysis (PCA) and RNN framework were used for detecting DDoS attacks (Meng et al., 2019). A protection system for mitigating DDoS attacks suitable to the cloud and the fog environments was created in (Priyadarshini and Barik, 2019).

In addition to these traditional DM and ML approaches, several advanced methods were also used. For instance, the work carried out in (Xu et al., 2007) had used a combination of reinforcement learning and hidden Markov that proved its ability to separate normal packets from DDoS attempts. Such a model was depending essentially on calculating the likelihood of the inbound IP address chronological sequence. Nevertheless, the intermediate data traffic had also been employed to create a NB model for recognizing the DDoS packets using the IP address of source and destination as well as several other shared data within the network nodes (Berral et al., 2008).

Several feature selection approaches were also used for enhancing the overall performance of the intelligent DDoS attack models. One of the first examples of such studies is the work which had been suggested in (Shon et al., 2005). In this work, Genetic Algorithm (GA) was used for selecting the features from data traffic, whereas SVM was used for detecting DDoS attacks. In (He et al., 2017) an intelligent DDoS detection model had been suggested. The model is based on SVM and NB by making use of the statistical features provided by cloud servers and virtual machines to prevent the data packets to transfer outside the network.

Overall, intelligent techniques proved their superiority in detecting DDoS attacks. This study aims to explore the applicability of rule induction methods in detecting DDoS attacks. Hence, an improved rule induction algorithm is proposed and is used for this purpose. Such an algorithm is called IRIDOS. The algorithm minimizes the searching space to obtain classification rules by eliminating all insignificant candidate rule-items in the process of developing the classification system.

3. Improved rule induction algorithm (IRI)

3.1. *General definitions related to classification data mining approach*

Considering a training dataset D that contains *x* different features, *"$Feature_1$, $Feature_2$, …, $Feature_x$"*, and one class variable *"Class"*. The cardinality of D is |D|. A feature could be nominal or continuous values. Nominal features are normally mapped into a collection of positive integer values. On the other hand, continuous features are pre-processed by discretizing the values with the help of some discretization techniques. The main goal is to build a classification model "M" from *"D"* such that *"M: Feature → Class"*, that predicts the class value of some hidden instances.

The *IRI* algorithm suggested in this study utilizes a predefined threshold value called *"Frq"*. Such a threshold is considered the fine line that differentiates between useful rule-items and un-useful ones according to the computed occurrence of each item in *"D"*. A rule-item that has a frequency greater than the predefined *"Frq"* is assumed a useful rule-item, otherwise, it is considered an un-useful one. Hereunder the most important terms related to the RI in general and the *IRI* algorithm, in particular, are defined.

Term 1: An item (F_i, v_i) is a feature and the value associated to it.

Term 2: An instance or an example is a set of features and their values (F_{j1}, v_{j1}), (F_{j2}, v_{j2}),..., (F_{jn}, v_{jn}) and a class value C_j.

Term 3: A rule-item involves 2 parts those are the rule antecedent and the rule consequent <*rule antecedent, rule consequent*>. The rule antecedent includes a set of items, whereas the rule consequent includes the class value.

Term 4: Frequency parameter *Frq*, is a value defined by users.

Term 5: Body frequency for rule-item (r_m) denotes how many dataset instances in *D* that exactly matches the body of r_m.

Term 6: The frequency of a rule-item denotes how many instances in *D* which exactly matches the *rule-item*.

Term 7: A useful rule-item is the one that passes *Frq*.

Term 8: Rule-Power is the minimum threshold accuracy a rule should has in order to be generated. This term is related to the *IRI* only and not to the general *RI* algorithms. For instance, In PRISM *(a well-known RI algorithm)* the rule is only generated if its error rate is 0%. However, in this research this term is introduced with the aim of minimizing the chances of producing an overfitted model, knowing that overfitting is a common issue in PRISM (Almutairi et al., 2017).

3.2. *IRI in detail*

The suggested *IRI* algorithm depicted in Algorithm 1 includes two main phases: Rule Creation phase and Class Assigning phase.

Algorithm-1: The Proposed IRI Algorithm

Input: Training Dataset, Rule-Power Threshold, Frq Threshold.
Output: A Classification Model includes a set of "If … Then" statements
Begin
 Foreach feature in D do
  Repeat /*for each class in D*/
   Repeat /*generating Rule R_j*/
    Compute Feature in C_i p(C_i=i|Feature); where C_i is class i in the training dataset D.
    Add the feature that has the highest accuracy (C_i=i|Feature) to the body of rule R_j.
   Until (accuracy of R_j=100%) or (R_j no longer can be improved and its Power>=Rule-Power Threshold)
  Generate R_j.
  Remove all training examples that are matched by body of R_j.
  Update the frequencies for all impacted items.
 Until all remaining unclassified items have Frequency<Frq; or no more classes left in D.
 Return Classification Model.
End

In the first phase, the algorithm uses the training dataset for generating a set of rules which all have accuracy > *Rule-Power*. *Rule-Power* is like the confidence parameter that is utilized in association classification approach (Antonelli et al., 2015) whose goal is to come up with best possible rule set in addition to rules that have 100% accuracy rate as in PRISM. The suggested training strategy assures the creation of rule set which have 0 error-rate in addition to rules which pass the *Rule-Power* threshold. The algorithm stops developing the classification model whenever no further rules obtain a sufficient accuracy or perhaps if the training dataset gets to be empty. Whenever this happens, all rules will be combined together in order to create the final classification model. Another threshold utilized in the first phase that will minimize the searching space of items is the *Frq* threshold. *Frq* is like the support parameter in the association classification approach. The *Frq* threshold is mainly used to distinguish between frequent items *(that appeared frequently in the dataset)* and the ones that aren't frequent. This definitely removes items that have low frequencies *(i.e. frequencies<Frq)* early and therefore investing computing resources in addition to making sure that just important items will partici-

pate in creating rule's body. Yet, infrequent item(s) are maintained in PRISM hoping to produce rules that have a 100% accuracy, that certainly is one of the main deficiencies in PRISM.

As soon as a rule is created, all dataset example that are associated with it will be removed, and the frequencies of the waiting possible item(s) that could be appended that shown within the removed data, will be automatically updated. Such an update requires decreasing their frequencies. This might be viewed as being a quality insurance strategy by not depending on the initial items frequencies that have been calculated using the initial training dataset. However, *IRI* uses a dynamic frequency for every item which constantly adjusted every time a new rule is created. With that said, the proposed algorithm is a RI strategy which doesn't permit items within rules to exchange training dataset instances, thus ensuring creating a model that doesn't depend on a static dataset, rather a dynamic one which decreases every time a new rule is created. In the second phase, the produced rules will be employed to forecast the class value of some testing examples. The proposed technique presumes that the features within the dataset hold nominal values or otherwise they should be discretized prior to the rule creation phase. The rule creation phase and class assigning phase are discussed in detail hereunder.

3.2.1. *Rule creation phase*

The suggested algorithm goes through the dataset in order to calculate the frequency for every single item within it. After that the first rule will be generated by adding the item that if included within the rule will obtain a better accuracy. On the other hand, the item(s) that has/have a frequency less than *Frq* is neglected. The algorithm keeps adding item(s) to the body of the rule so that its accuracy reaches a 100%. However, for any rule that could not achieve a 100% accuracy, the algorithm compares the obtained accuracy against the *Rule-Power* threshold value. If rule-accuracy is greater than *Rule-Power,* then the rule is added to the rules within the classification model. Otherwise, the rule will be removed. Two main differences can be concluded between *IRI* and PRISM, these are:

1- In *IRI*, an item is appended to the body of the rule only if its frequency is bigger than the predefined *Frq* threshold. On the other hand, if an item has a frequency less than *Frq* then it will be ignored, and it will not be appended to the body of the rule.
2- In PRISM, all the generated rules should have a 100% accuracy. However, in *IRI*, some rules will be added to the final classification model although their accuracy didn't reach a 100%.

Once the rule is created, the training dataset examples that match the body of the generated rule are tossed out. Then the algorithm re-calculates the items frequencies in order to start creating the next rule. The *IRI* keeps generating rules relating to the current class until no further items have a frequency greater than *Frq*. In this case, the algorithm goes to the next class value. This process is carried out over and over again until no more examples left in the dataset or the remaining examples have frequencies less than *Frq*. If there are some examples still un-covered because their frequencies are less than *Frq,* then a default class rule is generated, and it will be connected to the most frequent class in the remaining instances. The suggested rules discovery process ensures dynamic items ranks calculation especially because an item frequency associated with a class will constantly adjusted whenever a new rule is generated. This in fact offers a unique advantage for the algorithm in identifying items that turned out to be useless throughout the classifier construction process. This will minimize the searching space for candidate items and will create more compact classification models. In addition, the suggested algorithm has a built-in rule pruning process which decreases overfitting and provides rules that have much more data coverage compared to PRISM. To elaborate further, PRISM continues appending items to the rule's body no matter how many dataset examples are covered. Essentially, the main aim of PRISM is maximizing the accuracy of the generated rule no matter if it covers a single dataset instance. This might result in producing an overfitted model and leads to creating large number of *"low coverage rules"*. An example that could be cited here is when applying PRISM on the "WEATHER" dataset (Dua and Graff, 2017) because it will create a classification model that includes 6 rules where 2 out of them cover only 2 dataset examples *(more detail are discussed in Section 4)*. In contrast, the proposed algorithm ignores these rules because they don't have enough data representation. It is worth pointing out that there isn't any rule preferencing method in PRISM due to the fact that the generated rules have a 100% accuracy. Nevertheless, the classification model constructed with the proposed algorithm may have some rules with accuracy rates less than a 100% and therefore the *IRI* utilizes a mechanism for sorting such rules to distinguish between them. The *Rule-Power* and *Frq* are the primary measures used for sorting the rules. If several rules have a similar *Rule-Power,* then the algorithm makes use of the *Frq* for breaking such a tie. Yet, if several rules have the same *Rule-Power* and *Frq* then the rules that have fewer items in the rule body are preferred over others.

3.2.2. *Class assigning phase*

The *IRI* creates a classification model that includes 2 kinds of rules. The first kind of rules is the set of rules that have a 100% accuracy. However, any rule has an accuracy less than 100% belongs the second kind of rules. The first kind of rules will be referred to as "Primary-Rules", whereas "Minor-Rules" denotes the second kind of rules. The proposed algorithm sorts the rules in a top-down manner where the "Primary-Rules" come at the beginning, and the "Minor-Rules" come after. Once a new unseen instance arrives, the proposed algorithm starts with the "Primary-Rules" for anticipating the class value. Yet, if no "Primary-Rule" was able to classify the unseen example, the algorithm moves to the "Minor-Rules". The first rule that has a body consisting of a set of items that exactly match the items extracted from the unseen example is used to decide on the final class value. Such a class assigning technique decreases the utilization of default class rule which is normally fired if none rules in the classifier is able classify the unseen instance.

Table 1 – Considered training dataset.

Outlook	Temp	Humid	Wind	Play	Rule#
Sunny	Hot	High	False	No	2
Sunny	Hot	High	True	No	2
Overcast	Hot	High	False	Yes	1
Rainy	Mild	High	False	Yes	3
Rainy	Cool	Normal	False	Yes	3
Rainy	Cool	Normal	True	No	Default
Overcast	Cool	Normal	True	Yes	1
Sunny	Mild	High	False	No	2
Sunny	Cool	Normal	False	Yes	3
Rainy	Mild	Normal	False	Yes	3
Sunny	Mild	Normal	True	Yes	Default
Overcast	Mild	High	True	Yes	1
Overcast	Hot	Normal	False	Yes	1
Rainy	Mild	High	True	No	Default

4. A practical example

In the current section, a practical example will be discussed thoroughly to understand how the proposed algorithm learns a new classification model and how such learning process differs from the method applied by one of the well-known RI algorithms i.e. PRISM. The dataset depicted in Table 1 is considered as the training dataset.

IRI requires two more input parameters these are *RulePower* threshold and *Frq* threshold. Such parameters are assumed to be 80% and 3 respectively. As soon as the frequency of each item is calculated, the proposed algorithm selects the item with the highest accuracy when associated with a class. Table 2 shows that the highest accuracy was 100% (i.e. 4/4) and it was achieved when the item "Outlook=overcast" is associated with class "Yes". Therefore, the algorithm produces the following "Primary-Rule":

"$If Outlook = Overcast then class = Yes$" . . . . . $Rule\#1$

Thereafter, the algorithm removes the dataset instances *(Blue highlighted instances in Table 1)* that match the rule body and updates the frequencies of every single item shown in the eliminated instances as illustrated in Table 2. The Yellow highlighted items in Table 2 didn't pass the *Frq* threshold value, hence, they will not be considered for building a rule. Obviously, the proposed algorithm has considerably minimized the searching space by maintaining powerful (useful) items only *(un highlighted items in Table 2)*. On the other hand, PRISM maintains such items hoping to generate some extra rules.

As soon as Rule #1 is generated, 3 items associated with class "Yes" are removed because they become un useful those are "Temp=Cool", "Humid=High" and "Windy=True". Their frequencies are marked in Yellow in Table 2 particularly in the 3rd column. The proposed algorithm will start generating the next rule using the rest of the data instances without considering the eliminated data of RULE #1. As shown in column 4 and column 5 of Table 2, and using the updated frequencies, the highest accuracy was 80%, and it has been achieved when "Humid=High" which linked with class "No". Hence the proposed algorithm generates the next rule:

"$If Humid = High then class = No$"

Because Rule #2 doesn't archive a 100% accuracy, all examples linked to it were moved to Table 3 with the aim of finding additional items that may improve the rule's accuracy.

Using the data shown in Table 3, the algorithm will calculate the items frequencies. Table 4 shows that the "Outlook=Sunny" achieved the highest accuracy at 100% (i.e. 3/3).

Hence, this item will be added to Rule #2. After adding the item to Rule #2, the accuracy become 100%. Hence, Rule #2 will be generated, and all instances linked with it *(Green highlighted instances in Table 1)* will be removed.

"$If Humid = High and Outlook = Sunny then class = NO$" . . . $Rule\#2$

The algorithm will then update the frequencies and accuracies of the remaining affected items.

As a result of producing Rule #2, 4 items were neglected because they turned out to be useless *(Yellow highlighted frequencies in sub-column titled "F" under main column named "After creating Rule #2" in Table 2)*. In fact, there isn't any more items associated with class *"NO"* because no item among the remaining ones has a frequency greater than *Frq*. Rule #1 and Rule #2 have successfully covered 7 instances of the original dataset and 7 more instances were left. The algorithm again computes the accuracies of the leftover items. With an accuracy of 100% (i.e. 4/4), item "Windy=False" has the highest accuracy if associated with class "Yes". Therefore Rule #3 will be produced.

"$If Windy = False Then class = Yes$" . . . . . $Rule\#3$

Rule #3 covers 4 dataset instances *(Blue highlighted instances in Table 1)*. However, 3 more examples left in the dataset that were not covered by any rule. Hence, the algorithm proceeds in creating Rule #4. However, after calculating the frequencies of the remaining items, it has been shown that none of them passes the *Frq* threshold. Therefore, the algorithm will produce a default rule. Such a rule is linked with the class that has appeared more than others *(frequent class)* which is the "No" class value. Hence, Rule #4 is produced. After creating the fourth rule, the proposed algorithm has successfully covered all examples in the dataset.

"$Else class = No$"

The proposed algorithm was able to create a classification model that includes 3 rules and 1 default rule. On the other hand, when using WEKA (Hall et al., 2011), the classification model created by PRISM includes 6 rules as shown in Fig. 1.

Digging deep into the rules generated by the proposed algorithm and those generated by PRISM, it has been shown that some of the rules created by PRISM cover few number of dataset instances. For instance, Rule# 3 and Rule# 4 generated by PRISM cover only 1 instance. Also, Rule# 6 covers only 2 instances. Yet, each rule generated by the proposed algorithm cover at least 3 instances. This implies that the proposed algorithm derived almost 33% less rules than PRISM from a

Table 2 – Frequencies[1] and Accuracies[2] of items during rule generation process.

Items & Classes	While creating Rule #1		After creating Rule#1		After creating Rule#2		After creating Rule#3	
	F[3]	A[4]	F[1]	A[2]	F[1]	A[2]	F[1]	A[2]
"Outlook=Sunny" → "No"	3	60%	3	60%	0			
"Outlook=Rainy" → "No"	2							
"Temp=Hot" → "No"	2							
"Temp=Mild" → "No"	2							
"Temp=Cool" → "No"	1							
"Humid=High" → "No"	4	57%	4	80%	1			
"Humid=Normal" → "No"	1							
"Windy=True" → "No"	3	50%	3	75%	2			
"Windy=False" → "No"	2							
"Outlook=Overcast"→ "Yes"	4	100%	0					
"Outlook=Sunny" → "Yes"	2	40%						
"Outlook=Rainy" → "Yes"	3	60%	3	60%	3	60%	0	
"Temp=Hot" → "Yes"	2		0					
"Temp=Mild" → "Yes"	4	67%	4	67%	4	80%	2	
"Temp=Cool" → "Yes"	3	75%	2					
"Humid=High" → "Yes"	3	43%	1					
"Humid=Normal" → "Yes"	6	86%	4	80%	4	80%	1	
"Windy=True" → "Yes"	3	50%	1		2			
"Windy=False" → "Yes"	6	75%	4	67%	4	100%	0	

[1] "F" Denotes frequency
[2] "A" Denotes accuracy
[3] "F" Denotes frequency
[4] "A" Denotes accuracy

Table 3 – Dataset instances linked with item "Humid=High".

Outlook	Temp	Humid	Wind	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Rainy	Mild	High	False	Yes
Sunny	Mild	High	False	No
Rainy	Mild	High	True	No

Table 4 – Updated frequencies and accuracies for complete generating rule #2.

Items linked with class "No"	Frequencies	Accuracies
"Outlook=Sunny"	3	100%
"Outlook=Rainy"	2	50%
"Temp=Hot"	2	100%
"Temp=Mild"	3	67%
"Windy=True"	2	100%
"Windy=False"	3	67%

Prism rules

If Outlook = Overcast then Yes ..... Rule# 1
If Humid = Normal and Windy = FALSE then Yes ..... Rule# 2
If Temp = Mild and Humid = Normal then Yes ..... Rule# 3
If Outlook = Rainy and Windy = FALSE then Yes ..... Rule# 4
If Outlook = Sunny and Humid = High then No ..... Rule# 5
If Outlook = Rainy and Windy = TRUE then No ..... Rule# 6

Fig. 1 – Rules generated by PRISM.

dataset includes only 14 instances. This implicitly means that the proposed algorithm is able to produce more compact models. As a matter of fact, in addition to the other contributions the proposed algorithm introduces, this particular improvement is considered a major contribution to the RI approach specifically with high dimensionality datasets and with large datasets.

5. Emperical evaluation

In this section two sets of experiments will be completed. The first set of experiments aims to evaluate the generalization capacity of the IRI algorithm using several benchmark datasets. Whereas, in the second set of experiments, the IRI algorithm is used for detecting DoS attacks. The experimental settings, the dataset used in the experiments and the obtained results are explained in detail.

5.1. *Experimental settings*

The "10-fold cross validation" technique is used for validating the models created using the IRIDOS and other comparable algorithms. This technique is normally used to calculate the *"error-rate"* because it minimizes overfitting (Mohammad, 2016, Mohammad, 2019). In addition, it is usually employed in building classification models in different fields (Gonsalves et al., 2019, Mohammad, 2020). Most of the classification techniques when applied on datasets assess and validate the performance of the created classifiers using one-error rate validation metric. By employing this measure, the produced classification model predicts the class value of every test case withing the testing dataset and compares it with the real value. If the predicted value matches real value that will be considered an accurate classification. If not it is considered a misclassification. The "one-error rate" is calculated by simply dividing the number of misclassified testing instances by the total number of instances withing the testing dataset, and it assesses the predictive abilities of the created classification models.The most popular validation technique in DM is tenfold cross validation. This technique firstly splits the input dataset randomly into 10 parts where 9 are utilized to train the classification model while the remaining part is utilized for testing the created model. The process is repetitively invoked 10 times on the training dataset and the created outcomes i.e. one-error rates of all runs will be then averaged. Random sampling is done basically to make sure class representation in every part of the subsets as well as the testing subset. This process is known as stratification that ensures that all classes are existing after the split is performed.

All experimental tests were carried out using a Microsoft Windows 10 Enterprise with 16.0 GB Memory and Intel(R) Core (TM) i7-8650U CPU @ 1.90GHz, 2112 Mhz, 4 Core(s). The IRIDOS has been implemented in WEKA *("an open source Java platform that was developed at the University of Waikato, New Zealand")* to ensure using a similar testing platform for the IRIDOS and other comparable algorithms. WEKA includes several implementations of DM algorithms that can be applied for various tasks such as Feature Selection, Association Rule Mining, Regression, Clustering, and Classification. In order to evaluate the applicability of *IRI*, 15 publicly published datasets were used before applying it on the DoS dataset. Three well-known RI-based algorithms were used in the experiments namely OneRule (OR), Conjunctive Rule (CR), and PRISM. PART algorithm has also been considered for the comparison purposes. So, a total of 4 algorithms will be used. The bases behind selecting such algorithms is due to the fact that these algorithms use different learning techniques for crating the classification models that are normally included a set of *"If…Then"* rules. In addition to the training dataset, the IRIDOS requires 2 more input parameters namely *Frq* and *Rule-Power*. These threshold values were set to 2 and 60% respectively. Such values were selected after conducting several warming up tests in which the outcomes showed a balance between the classification accuracy and the size of the created models. The assessment metrics that are considered for comparing the performance of the proposed algorithm and other comparable algorithms include: the error rates, the classification model size (how many rules does the model includes), the number of dataset instances scanned when generating the classification model *(An importance of this metric is to assess the decrease and increase in the searching space)*, the time required for building the model, the number of covered dataset examples, and the rule length (on average, how many features used in the rule). These metrics in addition to their mathematical equations will be discussed further in this section.

Table 5 – Considered datasets and the error rates (%) obtained from IRI and all other considered algorithms.

Dataset	Number of Instances	Number of Features	Number of Classes	IRI	OR	CR	PRISM	PART
Contact Lenses	24	5	3	33.3	29.2	37.5	45.8	16.7
Labor	57	17	2	22.8	31.6	25.6	35.4	17.5
Zoo	101	11	7	6.9	93.1	40.6	57.4	7.9
Iris	150	5	3	10	4	39.3	12	4.7
Glass	214	10	7	47.7	48.6	53.7	51.4	39.3
Breast Cancer	286	10	2	32.5	34.3	33.6	41.5	30.4
Vote	435	17	2	6.4	4.4	4.9	6.7	4.9
Soyabean	683	36	19	10.1	66.5	73.8	14.6	8.5
Autos	690	15	2	17.9	14.5	14.5	21.3	14.2
Diabetes	768	9	2	27.1	26.4	34.4	39	26.6
Tic-Tac	958	10	2	8.5	30.1	31	4.3	5.8
German-Credit	1000	16	2	28.8	28.9	3.6	36.2	30.7
Segment Challenge	1500	20	7	13.1	47.6	70.9	14.7	11.8

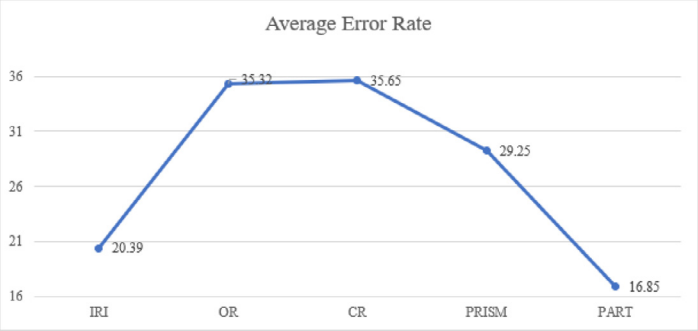

Fig. 2 – Average error rates for *IRI* and all considered algorithms.

5.2. *Results on benchmar datasets*

5.2.1. *UCI datasets preparation*

Several datasets obtained from a well-known training dataset repository called the *"University of California Irvine (UCI) Repository"* (Dua and Graff, 2017) will be used to generalize the capability of *IRI*.

Such datasets were chosen subject to several factors such as the type of features within the dataset, number of examples, number of features, and number of classes as depicted in Table 5. In addition, these datasets are commonly used as benchmark for evaluating different DM techniques.

5.2.2. *Results discussion*

Table 5 showed the error rates obtained from each considered DM algorithm, i.e. CR, PRISM, OR, PART, and IRI. Fig. 2 reveals the average error rates achieved from *IRI* and all other comparable DM techniques. Such a figure revealed that the *IRI* outperformed all other RI algorithms. For instance, the *IRI* has obtained an average error rate at 20.39 which is less than CR, PRISM, and OR by 15.26%, 8.86%, and 14.93% respectively. These gains have been acquired as a direct result from the innovative dynamic rule creation procedure employed by the *IRI* algorithm which produced accurate rules that lead to a noticeable enhancement of the overall predictive ability of the generated classification models.

Yet, PART surpasses the *IRI* and all other algorithms and that can be justified due to the fact that PART employs entropy *"information theory"* based pruning technique called *"Reduced Error Pruning"* which in turn has increased the overall performance of the produced classification model. More specifically, PART has an average error rate slightly better than *IRI* at 16.85% which is a 3.54% better than the *IRI* algorithm. As a matter of fact, one of the future improvements on *IRI* is to consider using a more advanced rule pruning methods such as entropy for improving the overall performance of the classifiers created by *IRI*. In general, considering that the *IRI* has produced an average error rate that is competitive to PART and created classification models that surpass other RI based algorithms is an achievement.

Digging deep into the obtained results, and in an effort to assess the significance in overall performance, a paired t-test has been carried out amongst the outcomes of the *IRI* and other algorithms. Paired t-test is a statistical technique for contrasting the results of evaluating a set of data two times, i.e. using 2 different algorithms. The comparison depends on the error rate across the 13 chosen datasets. Table 6 demonstrates the comparison findings were the confidence level is set to 95% where *"Won"* indicates how many times the *IRI* achieved an error rate less than the compared algorithm, and *"Los"* indicates how many times the *IRI* achieved an error rate higher than the compared algorithm.

The results depicted in Table 6 clearly show that the *IRI* obtained a significant improvement when compared against all RI algorithms, i.e. CR, PRISM, and OR. However, PART still doing better than the *IRI* although with marginal fractions in most

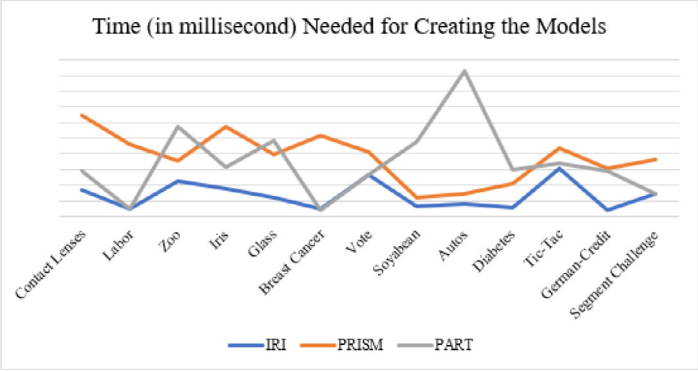

Fig. 3 – Time (in millisecond) needed for creating the models.

Table 6 – Results obtained from Paired t-test.

Dataset	Won	los
CR	10	3
PRISM	12	1
OR	8	5
PART	2	11

cases. As a matter of fact, in the worst case, PART achieved an error rate better than *IRI* with a margin of 16.6% when comparing the results obtained from "Contact Lenses" dataset, and in the best case it reached just 0.5% when looking into the results achieved from the "Diabetes" dataset.

The obtained results also showed that the size of the classification models created by *IRI* are considerably smaller than that created by PRISM. Interestingly, the *IRI* pruning strategy has considerably minimized the number of rules without negatively affecting the performance of the generated classification models. To elaborate further, the average number of rules created from *IRI* and PRISM were 33,3 and 117,2 respectively. This significant classifier size reduction of *IRI* is caused by firstly the removing of inadequate items throughout the rule creation process. Secondly, the creation of none 100% accurate rules has a good impact on the final classification models. Both of these innovative contributions have led to more compact models of *IRI* in contrast of PRISM. The dynamic update of the items' frequencies decreased the searching space and thus fewer number of items are considered. Therefore, eliminating the overlapping of the dataset examples amongst rules offers an exceptional impact on the size of the classification models. Particularly, *IRI* makes sure that the frequency of each item is revised immediately each time a new rule is created, this minimizes possible number of items for the following rule(s), and so, rules produced more quickly and most likely classifies more instances.

The time needed to build the classification models have also been recorded to assess the efficiency of the algorithms. Fig. 3 illustrates the times in millisecond[1] (ms) needed by *IRI*, PRISM and PART for building the classification models. Such a figure clearly shows that *IRI* consumes a shorter time to construct the models compared to PART and PRISM owing to the fact that no need to continue dividing dataset examples so as to maximize the accuracy of each rule. This has distinct benefit at least when talking about PRISM in the sense that it continues adding items to the rules in an attempt to reach an accuracy of 100%. Yet, this might result in some rules that are covering few dataset examples and hence creating a lot of redundant rules. Not only this, but also more time is needed for scanning the dataset examples to find the most frequent item that could be appended to the rule. In addition, such a technique might result in producing an overfitted model. PART, on the other hand, is slower than *IRI* because of the multiple pruning methodology employed by the algorithm.

Table 7 – Number of scanned instances while building the classification models.

Dataset	IRI	PRISM
Contact Lenses	171	1503
Labor	17905	13796
Zoo	20111	72852
Iris	16562	20613
Glass	56813	325951
Breast Cancer	62521	355215
Vote	227339	212805
Soyabean	90362	185563
Autos	78233	192532
Diabetes	418680	1979939
Tic-Tac	117235	509652
German-Credit	213556	6235442
Segment Challenge	938263	7147542

The searching space utilized by PRISM and IRI have also been investigated in an effort to recognize the main variations between both algorithms when building the classifiers. Therefore, how many times the dataset instances were scanned by both methods during the rules creation process were recorded. Table 7 demonstrates the main findings. In general, PRISM needed to scan more instances to produce the final models with the exception of the "Labor" and "Vote" datasets. However, PRISM needed more than 7 Million scans when it comes to the "Segment Challenge" dataset that in-

[1] 1 Second = 1000 Millisecond

cludes 20 features and 1500 examples. However, for the same dataset, the *IRI* needed less than 1 Million scans.

In fact, the *IRI* algorithm stops producing rules early as soon as a rule comes with a satisfactory error rate. This may be an obvious indication that the *IRI* algorithm had considerably decreased the searching space which could also be considered as an extra contribution of the *IRI*. To conclude, the *IRI* surpasses PRISM and all RI methods. The *IRI* algorithm clearly revealed high predictive results on large, medium and small datasets due to the effective learning approach used by *IRI* which make it a favorable method for making accurate decisions. In addition, capability to decreases the utilization of the default rule offers an advantageous impact on the accuracy of the *IRI* algorithm. Furthermore, pruning useless and redundant rules improved the abilities of the *IRI* by making sure that just reliable and useful rules will be utilized in the class assigning phase.

5.3. *IRI for detecting DoS attacks (IRIDOS)*

As discussed in Section 3.2, the proposed IRI algorithm consists of two main phases namely rule creation phase and class assigning phase. In order to apply the IRI for detecting DoS attacks, firstly the training dataset should be preprocessed and then passed to the rule creation phase for producing several rules that have an accuracy greater than the predefined rule-power. Unlike the traditional PRISM rule induction algorithm, a set of rules that have an accuracy rate of 100% as well as a set of rules that have an accuracy rate $>$ rule-power will be produced. If no further rules can be generated or the training dataset becomes empty, that means the rule creation phase is fully completed. Thus, the set of the extracted rules are combined together to create the DoS classification model. It should be mentioned that one of the threshold values that should be defined in the rule creation phase is the Frq threshold value. With the aim of minimizing the rule-items' searching space throughout the rule creation phase, the Frq threshold value is used. Both Frq and Rule-Power were set to 2 and 60% respectively and the rational behind using these values were discussed in Section 5.1. Once the rule creation phase is completed, the created model is evaluated and validated as discussed in Section 5.1. The training dataset preparation and the obtained results are discussed in detail in the following subsections.

5.3.1. *DoS dataset preparation*

In this set of experiments, a publicly available dataset called UNSW-NB15 (Moustafa, 2018) will be used. Three key motives stimulated adopting such a dataset. Firstly, it covers latest legitimate and attack traffics. Secondly, it is well organized and easy to understand. Lastly, it is more advanced when compared to other datasets allowing it to be an excellent benchmark to evaluate the performance of the *IRI* algorithm. Such dataset comprises 9 kinds of contemporary cyber-attacks and latest patterns of legitimate traffic. In addition to the class attribute, it consists of 48 input attributes divided into 5 categories namely *"Basic attributes, Content attributes, Flow attributes, Time attributes and Additional generated attributes"*. The dataset includes a sum of 257,705 instances labeled as being an attack or a legitimate one. Intended for reliable assessment of the suggested algorithm, legitimate and DoS instances were extracted from the dataset. The updated dataset includes 109,353 instances where 16353 belong to DoS and the remaining 93000 belong to normal (legitimate) traffic. Obviously, the resulted dataset criticized of being imbalanced dataset because the instances that are associated with "Normal" class constitutes 5 times the "DoS" class. In fact, class imbalance is a common problem when building any classification model. Two solutions can be applied to solve this issue namely oversampling and undersampling. In the oversampling technique the instances in the minority class will be randomly duplicated. However, in the undersampling approach some instances of the majority class are dropped so that both classes have almost the same number of instances. In this study, the undersampling without replacement technique is used. The updated dataset includes 32755 instances divided into 16402 as normal traffic and 16353 as DoS attacks. Several further preprocessing tasks on the updated dataset were implemented. For instance, the first 14 extra features within the dataset were dropped. Further, it is well-known that the DoS attacks typically using reflectors which most likely are set of legitimate computers controlled by the attackers who make use of the IP address of these computers to perform DDoS attacks. Therefore, in the DoS attacks the IP addresses do not provide enough useful clues for classifying the traffic as either normal or an attack. Consequently, the features that contain information about source and destination IP addresses were removed from the dataset. Actually, there are several studies that confirm our assumption about the non-importance of the IP addresses in revealing DoS attacks, among others we mention (Moustafa and Slay, 2015, Idhammad et al., 2017, Janarthanan and Zargari, 2017, Kumar et al., 2019). This enables generating a reduced dataset of 33 attributes. The Correlation based Feature Selection (CFS) technique is applied to decide on the best attributes for detecting DoS attacks. CFS is an effective filtering approach which ranks a set of attributes based on a correlation based heuristic assessment algorithm. CFS measures subsets of attributes based on a simple hypothesis, that is: *"beneficial attribute subsets consist of attributes highly correlated with the class, yet uncorrelated to each other"*. The final dataset is decreased from 33 attributes to 8 attributes. Table 8 describes theses attributes.

5.3.2. *Results discussion*

In this section, three more evaluation measures will be assessed in addition to these described in Section 5.1. The first metric is the True Positive Rate (TPR) which also known as *"Recall"* and *"Sensitivity"* and it measures the percentage of testing instances that are classified correctly. The second metric is Precision (PS) which is the percentage of relevant examples amongst the complete retrieved examples. And the last metric is the F1-measuer which is the weighted average of PS and TPR.

Table 9 reveals the results obtained from the *IRI* and all other comparable DM algorithms when apply them on the preprocessed dataset that has been described in part A of the current section.

Such a table showed competent and encouraging results from the proposed algorithms. To elaborate further, the *IRI* attained major classification improvements when compared to

Table 8 – Attributes selected from CFS.

Attribute	Description
proto	Transaction protocol
dur	Record total duration
sttl	Source to destination time to live value
dttl	Destination to source time to live value
sloss	Source packets retransmitted or dropped
Dpkts	Destination to source packet count
Dintpkt	Destination interpacket arrival time
ct_dst_sport_ltm	No of connections of the same destination address and the source port in 100 connections according to the last time

Table 9 – Experimental results on DoS dataset.

	IRI	OR	CR	PRISM	PART
Error Rate	6.10%	11.80%	17.80%	8.40%	5.20%
TPR	93.30%	88.20%	82.20%	91.60%	94.80%
PS	94.50%	88.20%	86.70%	92.00%	94.80%
F1-Measure	93.90%	88.00%	81.60%	91.50%	94.80%

OR and CR. For instance, the *IRI* obtained an error rate less than OR and CR with margins of 5.7% and 11.7% respectively. In addition, the *IRI* achieved better TPR when compared to the OR and CR in the sense that it surpassed them with margins of 5.1% and 11.1%. In terms of PS, the suggested algorithm beaten both OR and CR with margins of 6.3% and 7.8%. However, the highest improvement in evaluation metrics is shown when considering the F1-measure because the *IRI* surpassed OR and CR with margins of 5.9% and 12.3% respectively. However, the IRI has also shown improvement when compared to PRISM although modest improvement in some cases. Particularly, the *IRI* surpassed PRISM in terms of error rate, TPR, PS, and F1-mesaure with margins of 2.3%, 1.7%, 2.5%, and 2.4% respectively. The innovative learning strategy employed by the *IRI* is believed to be the main reason behind achieving such exceptional remarkable results. Another reason of attaining these results is the *IRI* ability to reduce using the default class rule. Pruning inadequate and unnecessary rules has also improved the capability of *IRI* in producing reliable rules for predicting the class value. Interestingly, the PART algorithm achieved marginally the lowest error rate and the highest TPR, PS, and F1-measure with margins of 0.9%, 1.5%, 0.3%, and 0.9% respectively when compared to *IRI*. These results can be seen as a motivation for enhancing the suggested algorithm in the near future by incorporating an effective post rule pruning procedure in the sense that the advanced post rule pruning method employed by PART is believed the main reason for achieving such results.

When studying the number of generated rules, it has shown that with 449 rules, PRISM has produced 93 extra rules when compared to *IRI* which constitute a margin of 1.26% as shown in Fig. 4.

Such a figure stresses that the *IRI* algorithm created more reduced classification models when compared to PRISM. Digging deeply into the rules generated by *IRI* it has shown that almost 25% of the rules generated by *IRI* didn't achieve a 100% accuracy. This in fact the main reason why *IRI* has produced

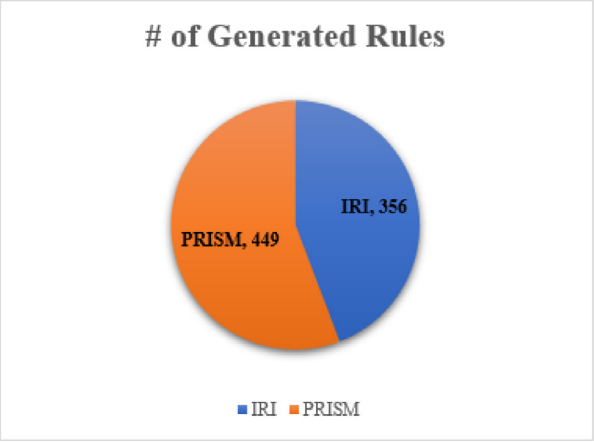

Fig. 4 – Number of rules generated by IRI and PRISM.

smaller number of rules than PRISM. On the other hand, 63 rules generated by PRISM cover only less than 2 instances which could be a reason of producing an overfitted model. This definitely is an indicator that creating rules not always having 100% accuracy doesn't only minimizes the size of the classification model but additionally cover higher number of training dataset instances for each rule.

For the purposes of this study, an extra measure is proposed and named Average Number of Features (ANF) and it is computed as per Eq. (1). ANF assesses the average number of features used for building the rules within the final classification model.

$$ANF = \frac{\sum_{i=1}^{T} Number\ of\ Features\ in\ Rule\ (i)}{T} \quad (1)$$

The results depicted in Fig. 5 clearly showed that the ANF needed by PRISM for building its model is higher than that needed by *IRI*. In other words, PRISM spent extra efforts for selecting features that could be added to each rule with the aim of achieving rules with 100% accuracy.

The number of times the dataset instances were scanned for building the classification models were also studied. It has shown that PRISM scanned the training dataset instances 41,524,891 times while IRI scanned the dataset instances 11,417,362 times. Such results obviously revealed that the IRI algorithm has significantly decreased the searching space when building the classification model. Considering that PRISM needs to pass over several extra millions of dataset

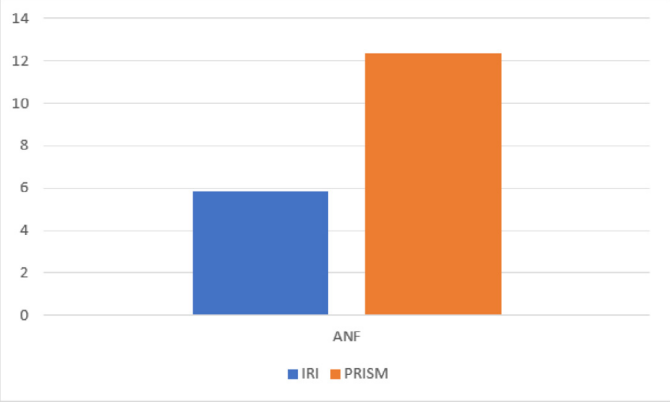

Fig. 5 – The ANF for IRI and PRISM.

instances sheds light on the poor rules finding approach utilized by PRISM. The *Frq* threshold utilized in IRI managed to effectively eliminate several items which have inadequate data representation.

6. Limitation of IRI and possible enhancements

One of the limitations that the IRI algorithm might face is when the training datasets are distributed across various separate data centers (as in cloud computing) due to the fact the IRI is designed to work in a local mode. However, although cloud computing poses a challenge for the IRI it also offers a set of affordable computing resources that could help in improving the IRI algorithm to be able to cope with not only scattered datasets but also with big datasets. For instance, Hadoop and Spark (Jorge et al., 2015) are probably the most noticeable cloud-based distributed systems meant for processing distributed and big datasets. Both of them are Apache top-level solutions. Nevertheless, its necessary to be aware of the aspects associated with each one before deciding which one to use. Another limitation that the IRI might face is how to deal with unstructured datasets such as webpages, presentations, audio files, photos, videos, word processing documents, and e-mail messages. Typically, unstructured datasets are not saved in a traditional column-row form. In fact, there are two major challenges associated with unstructured dataset:

1- Converting it into a form that can be easily understood by DM algorithms in general and IRI in particular.
2- Exploring insightful features within the dataset so that the IRI algorithm can be able to find useful patterns.

Therefore, such datasets demand substantial amount of time and efforts for converting them into a form that can be easily processed. Deep Artificial Neural Networks (DANN) happen to be very effective for dealing with these two challenges. Therefore, it would be interesting to find an algorithm that leverage strengths of IRI and DANN. Technically speaking, and with the aim of producing more robust classification models, it is recommended to employ post-pruning methods with the IRI algorithm. Post-pruning methods normally used after producing the final model with the aim of removing the redundant and unusual rules that are created during the learning phase.

7. Conclusion and future work

DoS is regarded as a serious cybersecurity threat that affecting most of emergent online-based services. DoS attacks overwhelm networking infrastructure by using a distributed set of malicious and controlled computer systems for starting the DoS attack. Rule Induction is seen as an appealing classification strategy in DM which enticed scholars owing to its simplicity, high classification capability, and robustness. In this article, an Improved Rule Induction algorithm is proposed and applied for detecting DoS attacks. Such an algorithm is called IRIDOS. The suggested algorithm reduces the searching space for creating classification rules by eliminating all insignificant items during the process of creating the classification models. In fact, the algorithm stops learning a rule as soon as the rule meets a threshold value called “rule-power”. A major benefit of the suggested algorithm is generating classification models that are more compact in size which is heavily weighted when producing any classification system. In addition, the IRI algorithm solves the problem of producing rules that cover limited number of training dataset instances. The WEKA platform was benefited for implementing the proposed algorithm. To evaluate the generalization ability of the proposed technique, 13 different datasets obtained from UCI repository were used for assessing the overall performance of the algorithm. Various evaluation metrics were used in the experiments. The obtained results were very promising especially when compared against several other DM algorithms. Interestingly, the proposed algorithm was able to produce more compact classification models and that can be as direct result of the effective learning strategy that the algorithm uses. The algorithm is then used for detecting DoS attacks and again the algorithm showed comparative results. However, the two sets of experiments confirmed that the algorithm can further be incorporating an advanced post pruning method. This in fact left as a future work which we intend to implement soon.

Declaration of Competing Interest

None.

CRediT authorship contribution statement

Rami Mustafa A. Mohammad: Conceptualization, Data curation, Formal analysis, Investigation, Supervision, Writing - original draft. **Mutasem K. Alsmadi:** Conceptualization, Methodology. **Ibrahim Almarashdeh:** Data curation, Writing - original draft. **Malek Alzaqebah:** Data curation, Writing - review & editing, Methodology.

REFERENCES

Al-Haidari F, Sqalli MH, Salah K. In: 2013 IEEE 5th International Conference on Cloud Computing Technology and Science. Impact of CPU utilization thresholds and scaling size on autoscaling cloud resources Bristol, UK; 2013.

Aljumah A, Ahamad T. A novel approach for detecting DDoS using artificial neural. Int. J. Comput. Sci. Netw. Secur. 2016;16(12):132–8.

Alkasassbeh M, Hassanat A, Al-Naymat G, Almseidin M. Detecting distributed denial of service attacks using data mining techniques. Int. J. Adv. Comput. Sci. Appl. 2016:436–45.

Almutairi M, Stahl F, Bramer M. In: International Conference on Innovative Techniques and Applications of Artificial Intelligence. Improving modular classification rule induction with G-prism using dynamic rule term boundaries; 2017.

D. Andersen, M. Baitaliuc, P. Biglete, K. Claffy, A. Dainotti, J. Eshabarr, P. Hick, B. Huffaker, Y. Hyun, K. Keys, A. King, R. Koga, A. Ma, R. Mok, J. Polterock, J. Weber, E. Yulaeva and M. Zhang, "The CAIDA "DDoS attack 2007" Dataset"Jan 2019. [Online]. Available: https://www.caida.org/data/passive/ddos-20070804_dataset.xml. [Accessed 25 Oct 2019].

Antonelli M, Ducange P, Marcelloni F, Segatori A. A novel associative classification model based on a fuzzy frequent pattern mining algorithm. Expert Syst. Appl. 2015;42(1):2086–97.

Balkanli E, Alves J, Zincir-Heywood NA. In: 2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS). Supervised learning to detect DDoS attacks Orlando; 2014.

Bawany NZ, Shamsi JA, Salah K. DDoS attack detection and mitigation using SDN: methods, practices, and solutions. Arab. J. Sci. Eng. 2017;42(1):425–41.

Berral JL, Poggi N, Alonso J, Gavaldà R, Torres J. Adaptive distributed mechanism against flooding network attacks based on machine learning. AISec '08: Proceedings of the 1st ACM workshop on Workshop on AISec, 2008.

Bhardwaj K, Miranda JC, Gavrilovska A. In: USENIX, Workshop on Hot Topics in Edge Computing. Towards IoT-DDoS prevention using edge computing Boston , MA, USA; 2018.

Braga R, Mota E, Passito A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. Proceedings of the 35th Annual IEEE Conference on Local Computer Networks (LCN '10), 2010.

Cendrowska J. PRISM: an algorithm for inducing modular rules. Int. J. Man Mach. Stud. 1987:349–70.

Cheng J, Zhang C, Tang X, Sheng VS, Dong Z, Li J. Adaptive DDoS attack detection method based on multiple-kernel learning. Secur. Commun. Netw. 2018;2018(1):1–19.

Choi J, Kim Y, Shin S-Y, Hong J. In: 33rd Annual ACM Symposium on Applied Computing. Smart IoT monitoring framework based on oneM2M for fog computing Pau, France; 2018.

C. Crane, "The 15 top DDoS statistics you should know in 2020," Nov 2019. [Online]. Available: https://cybersecurityventures.com/the-15-top-ddos-statistics-you-should-know-in-2020/.

Dayal N, Maity P, Srivastava S, Khondoker R. Research trends in security and DDoS in SDN. Secur. Commun. Netw. 2017;9(1):6386–411.

Doshi R, Apthorpe N, Feamster N. In: 2018 IEEE Security and Privacy Workshops (SPW). Machine learning DDoS detection for consumer internet of things devices San Francisco, CA; 2018.

Du P, Nakao A. In: 2010 IEEE Network Operations and Management Symposium - NOMS 2010. DDoS defense as a network service Osaka, Japan; 2010.

Dua D, Graff C. UCI Machine Learning Repository. Irvine: University of California, School of Information and Computer Sciences; Jan 2017. [Online]. Available: http://archive.ics.uci.edu/m [Accessed 26 July 2019].

Feng J, Hong B-K, Cheng S-M. DDoS attacks in experimental LTE networks. Artif. Intell. Netw. Appl. 2020;1150(1):545–53.

Gonsalves AH, Thabtah F, Mohammad RM, Singh G. Prediction of coronary heart disease using machine learning: an experimental analysis. ICDLT 2019: Proceedings of the 2019 3rd International Conference on Deep Learning Technologies, 2019.

Guenane F, Nogueira M, Serhrouchni A. In: 2015 IEEE Trustcom/BigDataSE/ISPA. DDOS mitigation cloud-based service Helsinki, Finland; 2015.

Gupta S, Kumar P, Abraham A. A profile based network intrusion detection and prevention system for securing cloud environment. Int. J. Distrib. Sens. Netw. 2013;9(3):1–12. doi:10.1155/2013/364575.

Gupta BD. A comprehensive survey on DDoS attacks and recent defense mechanisms. In: Handbook of Research on Intrusion Detection Systems. IGI Global; 2020. p. 33.

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. Waikato Environment for Knowledge Analysis. University of Waikato; 2011. [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/ [Accessed 20 December 2011].

He Z, Zhang T, Lee RB. In: 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud). Machine learning based DDoS attack detection from source side in cloud New York, NY; 2017.

Idhammad M, Afdel K, Belouch M. DoS detection method based on artificial neural. Int. J. Adv. Comput. Sci. Appl. (IJACSA) 2017;8(4):465–71.

Imperva. DDoS Protection. Imperva; 2002. [Online]. Available: https://www.imperva.com/products/ddos-protection-services/ [Accessed 21 July 2020].

Janarthanan T, Zargari S. In: 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE). Feature selection in UNSW-NB15 and KDDCUP'99 datasets Edinburgh, UK; 2017.

Jorge LR-O, Luca O, Anguita D. Big data analytics in the cloud: spark on hadoop vs MPI/OpenMP on beowulf. Procedia Comput. Sci. 2015;53(1):121–30.

Kato K, Klyuev V. An intelligent DDoS attack detection system using packet analysis and support vector machine. Int. J. Intell. Comput. Res. 2014;5(3):464–71. doi:10.20533/ijicr.2042.4655.2014.0060.

Khan MA, Salah K. IoT security: review, blockchain solutions, and open challenges. IoT Sec. 2018;82(1):395–411.

Khan WZ, Rehman MH, Zangoti HM, Afzal MK, Armi N, Khaled S. Industrial internet of things: recent advances, enabling technologies and open challenges. Comput. Electr. Eng. 2020;81(1):1–13.

Kumar V, Das AK, Sinha D. Statistical analysis of the UNSW-NB15 dataset for intrusion detection. Comput. Intell. Pattern Recogn. 2019;999(1).

O. Kupreev, E. Badovskaya and A. Gutnikov, "DDoS attacks in Q2 2019," 18 3 2020. [Online]. Available: https://securelist.com/ddos-report-q2-2019/91934/.
Lim S, Yang S, Kim Y, Yang S, Kim H. Controller scheduling for continued SDN operation under DDoS attacks. Electron. Lett. 2015;51(16):1259–61.
McAfee. McAfee. McAfee; 1987. [Online]. Available: https://www.mcafee.com/enterprise/en-us/products/mcafee-connect/denial-of-service.html [Accessed 21 July 2020].
Meng L, Zhang Y, Yan J. DDoS attacks detection using machine learning algorithms. Digital TV Multimed. Commun. 2019;1009:205–16. doi:10.1007/978-981-13-8138-6_17.
Mohammad RM, Alqahtani M. A comparison of machine learning techniques for file system forensics analysis. J. Inf. Sec. Appl. 2019;46(1):53–61.
R. Mohammad, "An ensemble self-structuring neural network approach to solving classification problems with virtual concept drift and its application to phishing websites," University of Huddersfield, 2016-A.
Mohammad RM. In: 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA). A neural network based digital forensics classification Aqaba; 2018.
Mohammad RM. An enhanced multiclass support vector machine model and its application to classifying file systems affected by a digital crime. J. King Saud Univ. - Comput. Inf. Sci. 2019 In press. doi:10.1016/j.jksuci.2019.10.010.
Mohammad RM. An improved multi-class classification algorithm based on association classification approach and its application to spam emails. IAENG Int. J. Comput. Sci. 2020;47(2):187–98.
Mohammad RM. A lifelong spam emails classification model. Appl. Comput. Inf. 2020 In press. doi:10.1016/j.aci.2020.01.002.
Mousavi SM, St-Hilaire M. In: 2015 International Conference on Computing, Networking and Communications (ICNC). Early detection of DDoS attacks against SDN controllers Garden Grove, CA, USA; 2015.
Moustafa N, Slay J. In: 2015 Military Communications and Information Systems Conference (MilCIS). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set) Canberra, ACT; 2015.
N. Moustafa, "The UNSW-NB15 dataset description," UNSW-NB15, Nov 2018. [Online]. Available:https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. [Accessed 20 April 2019].
Netscout. Arbor Threat Mitigation System. Netscout; 1984. [Online]. Available: https://www.netscout.com/product/arbor-threat-mitigation-system [Accessed 21 July 2020].
Osterweil E, Stavrou A, Zhang L. 20 years of DDoS: a call to action. arxiv Netw. Int. Arch. 2019;1(1):1–11.
F. L. Pierson, "Enhanced protection of processors from a buffer overflow attack". US Patent US10564969B2, 13 2 2020.
Pinson-Roxburgh O. Bulletproof Annual Cyber Security Report 2019. Bulletproof; 2019.
Priyadarshini R, Barik RK. A deep learning based intelligent framework to mitigate DDoS attack in fog environment. J. King Saud Univ. - Comput. Inf. Sci. 2019:1–7.
Sahi A, Lai D, Li Y, Diykh M. An efficient DDoS TCP flood attack detection and prevention system in a cloud environment. IEEE Access 2017;5(1):99–112.
Said EK, Salah K. Efficient and dynamic scaling of fog nodes for IoT devices. J. Supercomput. 2017;73(1):5261–84.
Saied A, Overill RE, Radzik T. Detection of known and unknown DDoS attacks using Artificial Neural Networks. Neurocomputing 2016;172(1):385–93.
Salah K, Calero JA, Zeadally S, Al-Mulla S, Alzaabi M. Using cloud computing to implement a security overlay network. IEEE Sec. Priv. 2013;11(1):44–53.
Salah K. In: 2013 IEEE Sixth International Conference on Cloud Computing. A queueing model to achieve proper elasticity for cloud cluster jobs Santa Clara, CA, USA; 2013.
Shon T, Kim Y, Lee C, Moon J. A machine learning framework for network anomaly detection using SVM and GA. Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop, 2005.
V. Simonovich, "Imperva blocks our largest DDoS L7/Brute force attack ever (Peaking at 292,000 RPS)," 2019 Jul 2019. [Online]. Available: https://www.imperva.com/blog/imperva-blocks-our-largest-ddos-l7-brute-force-attack-ever-peaking-at-292000-rps/.
Siaterlis C, Maglaris B. Detecting DDoS attacks using a multilayer Perceptron classifier. In: 9th IFIP/IEEE International Symposium on Integrated Network Management. IEEE; 2005. p. 1–14.
Sofi I, Mahajan A, Mansotra V. Machine learning techniques used for the detection and analysis of modern types of DDoS attacks. Int. Res. J. Eng. Technol. (IRJET) 2017;4(6):1085–92.
Subbulakshmi T, BalaKrishnan K, Shalinie SM, AnandKumar D, GanapathiSubramanian V, Kannathal K. In: 2011 Third International Conference on Advanced Computing. Detection of DDoS attacks using enhanced support vector machines with real time generated dataset Chennai; 2011.
Visalatchi L, Yazhini P. The survey DDoS attack prevention and defense technique. Int. J. Innov. Sci. Res. Technol. 2020;5(2):65–8.
Vizváry M, Vykopal J. Future of DDoS attacks mitigation in software defined networks. Lect. Notes Comput. Sci. 2014;8508(1):123–7.
Wei L, Fung C. In: 2015 IEEE International Conference on Communications (ICC). FlowRanger: a request prioritizing algorithm for controller DoS attacks in software defined networks London, UK; 2015.
Wu Y-C, Tseng H-R, Yang W, Jan R-H. In: 2009 Third International Conference on Multimedia and Ubiquitous Engineering. DDoS detection and traceback with decision tree and grey relational analysis Qingdao; 2009.
Xu X, Sun Y, Huang Z. In: Pacific-Asia Workshop on Intelligence and Security Informatics, PAISI 2007. Learning, defending DDoS attacks using hidden Markov models and cooperative reinforcement Berlin, Heidelberg; 2007.
Yassin W, Udzir NI, Muda Z, Abdullah A, Abdullah MT. In: 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec). A Cloud-based intrusion detection service framework Kuala Lumpur, Malaysia; 2012.
Ye J, Cheng X, Zhu J, Feng L, Song L. A DDoS attack detection method based on SVM in software defined network. Hindawi Secur. Commun. Netw. 2018;2018:8–17. doi:10.1155/2018/9804061.
Yu S, Tian Y, Guo S, Wu DO. Can we beat DDoS attacks in clouds? IEEE Trans. Parallel Distrib. Syst. 2014;25(9):2245–54.

Dr Rami (the corresponding author) holds Ph.D in Computer Science and Informatics from the University of Huddersfield. Dr Rami has several publications that span the field of computer science and Information Security. Specifically, Web based security, Information Security, and Digital Forensics. Currently, Dr Rami works as an assistant professor in the departement of Computer Information Systems, College of Computer Science and Information Technology in the Imam Abdulrahman Bin Faisal University – Saudi Arabia.