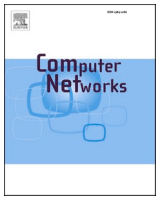
Computer Networks 201 (2021) 108591

Contents lists available at ScienceDirect

Computer Networks

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



Sustaining accurate detection of phishing URLs using SDN and feature selection approaches

Raniyah Wazirali [a], Rami Ahmad [b,*], Ashraf Abdel-Karim Abu-Ein [c]

[a] Department of Computer Science, College of Computing and Informatics, Saudi Electronic University, Saudi Arabia
[b] The school of Information Technology, Sebha University, Sebha 71, Libya
[c] Department of Electrical and Electronics Engineering, Huson University College, Al Balqa Applied University, Jordan

ARTICLE INFO

Keywords:
Recursive feature elimination
Phishing URL
SDN
CNN
Deep learning
Phishing detection

ABSTRACT

Phishing is an online fraud that deceives visitors by impersonating a legitimate website to steal their confidential or personal information. This is a well-known form of cybercrime. With the aim of detecting phishing sites, several phishing site detection techniques have recently been created. However, it fails to achieve the desired goal and has a large number of drawbacks, including low accuracy, long learning curve, and low-power embedded hardware. For covering such drawbacks, this work proposes an efficient URLs Phishing detection technique. Our technique depends on Software Defined Network (SDN) technology, clustering and feature method, and Conventional Neural Network (CNN) algorithm. Feature selection technique is based on Recursive Feature Elimination (RFE) with Support Vector Machine (SVM) algorithm. The SDN is used to transfer the URLs phishing detection process out of the user's hardware to the controller layer, continuously train on new data, and then send its outcomes to the SDN-Switches. RFE-SVM and CNN are used to increase accuracy of phishing detection. Therefore, the proposal model does not require retrieving the content of the target website or using any third-party services. It captures the information and sequential patterns of URL strings without requiring a prior knowledge about phishing, and then uses the sequential pattern features to quickly classify the actual URL. The experimental results showed that our proposal highlighted the robustness and accuracy in distinguishing between phishing and legitimate sites. Our suggestion achieves 99.5% phishing detection accuracy.

1. Introduction

Phishing websites are fraudulent websites with identical websites and Uniform Resource Locator (URL) addresses to legitimate websites, which can be used to trick users into accessing and inadvertently expose their personal information, ultimately serving the interests of criminal offenders [1]. Therefore, representatives of malicious threats can use several methods in creating a phishing campaign, with advertising and email campaigns being the most common. In addition, as the number of internet users' increases, the risks of a successful phishing attack also increase, and the successful phishing can take a huge toll on the victims [2]. The report published by Anti-Phishing Working Group (APWG) in 2019 (Quarter 2) reported that the number of phishing attacks increased by 30% from the previous quarter [3]. Moreover, Kaspersky statistics showed that it added 695,167 new phishing masks to its anti-phishing databases in the first quarter of 2021 [4]. Therefore, many administrations adopt security education training awareness programs to generate end-user awareness. However, users' skills in using these precautions vary, and the problem remains [5]. Other administrations use traditional methods such as blacklist and whitelist to block URLs that hackers had previously attempted to log [6]. However, blacklist and whitelist methods need time to update their latest lists, while most phishing URLs last for less than 2 h [7].

In addition to blacklisting and whitelisting, using Machine Learning (ML) technology to detect phishing URLs is a good solution. Many studies have used this method [8,9], since malicious URLs or deceptive web pages share some characteristics with each other. However, the ML training time will be high due to the use of all of the features of URLs. Thus, the process of identifying these features for phishing URLs in order to speed up the training process will be done manually. Therefore, other techniques have emerged that help identify the phishing features of popular URLs but not others, and then get more accurate and less expensive results. The deep learning algorithms such as Conventional Neural Network (CNN) and Long Short Term Memory (LSTM) could

* Corresponding author.
E-mail addresses: r.wazirali@seu.edu.sa (R. Wazirali), r_a_sh2001@yahoo.com (R. Ahmad), ashraf.abuain@bau.edu.jo (A.A.-K. Abu-Ein).

https://doi.org/10.1016/j.comnet.2021.108591
Received 8 July 2021; Received in revised form 15 October 2021; Accepted 30 October 2021
Available online 6 November 2021

automatically find patterns (features) from dataset that fulfill the classifier's criteria [1,10]. Beside, additional feature selection algorithms such as Recursive Feature Elimination (RFE) [11], Particle Swarm Optimization (PSO) [12], SelectKBest [13] and Mine Blast (MB) [14] are used to improve feature reduction and then phishing optimization as well, this increases the accuracy of phishing URL detection. In addition, the RFE is combined with Support Vector Machine (SVM) to support clustering between data. whereas, VSM takes into account the theory of statistical learning and applies the principle of minimizing structure risk in order to reduce experimental error and reduce complexity. The main goal of SVM is to build the optimal super-level with the largest margin [15]. However, these functions improve the accuracy of URLs phishing detection but raise the cost of analysis and hardware power consumption. Therefore, these technologies are not actually suitable for low-power devices such as wireless sensor networks [16,17].

The Software Defined Network (SDN) is a good solution for future network management, which separates the control from forwarding devices [18]. Thus, the control unit becomes centralized and responsible for managing all network parts and host devices. Therefore, this environment becomes good for dealing with phishing attacks on all types of host devices, regardless of their power. Despite the usefulness of this technique in the field of phishing deduction, we find that there is a lack of studies that have used it. Authors in [5,19] used SDN technology along with ML algorithms to improve detection of phishing sites, but there was use of whitelisting and blacklisting technology combined with it. Therefore, in this work we will improve the URLs phishing detection through merging between SDN, feature selection and clustering (RFE-SVM), and CNN algorithms to improve detection accuracy and reduce cost of time. The SD-Controller in the SDN will be responsible for managing the Feature Selection CNN (FS-CNN) function, then the output of this function is transmitted to the SD-Switches through updating its flow table rules. Each new packet that is transferred from the host to SD-Switch, the switch will pre-process it and subject it to the matching with the rules in the flow-table, if they match, the packet will pass or drop. The FS-CNN consists of various sequential features which are preprocessing, extraction of the feature matrix, feature learning and the classification. Therefore, one of the most important contributions of this work is that it does not require retrieving the content of the target website or using any of the third-party services. It captures the information and sequential patterns of URL strings without requiring prior knowledge about phishing, and then uses the sequential pattern features to quickly classify the actual URL.

The major contributions of this work are summarized as follows:

1- We proposed a new URL phishing detection environment through embedded SDN technology and deep learning network (CNN) algorithm called FS-SDN to transport the cost of detection training to SD-Controller.
2- We create a new deep learning technique called FS-CNN which combines feature extraction (one hot encoding), clustering (RFE-SVM), and CNN algorithm to improve detection accuracy.
3- Evaluate FS-CNN using real dataset and compare it with existing works as well as also examine the FS-SDN for real-world efficiency.

The rest of the paper is organized as follows. Section 2 provides an overview of related work in SDN, feature selection, and machine learning techniques. Section 3 explains the FS-SDN methodology. Section 4 discusses the experimental environment, performance metrics, and results. Finally, the conclusion and future work of the paper are presented in Section 5.

2. Background and related works

In this section we will present a technical background and literature reviews for detecting phishing websites through the use of SDN technology, machine learning algorithms, and feature selection mechanisms in order to give readers a glimpse into the basics of these techniques. Therefore, we will provide a brief background on SDN technology, intrusion detection, and feature selection technique. Then, we will discuss the recent works of URL phishing detection mechanisms.

2.1. Background

2.1.1. SDN technology

SDN architecture is a promising one that overcomes the limitations of traditional network architecture in terms of control, scalability, and management [1]. The basic idea of SDN is to separate the control plane (Core) from the data plane (Access), where the control logic is extracted from network hardware and is centralized into an independent control plane which is called "Controller". The controller also acts as a proxy between the infrastructure plane and application plane. It translates high-level instructions at the application layer into low-level rules and then forwards them to the switches at the infrastructure plane as illustrated in Fig. 1. Therefore, the controller software such as Open Day light [20] or Open Network Operating System (ONOS) [21] will be loaded onto devices with huge capabilities in order to withstand.

Fig. 1 shows the SDN architecture consisting of three main planes: application, control, and infrastructure. The SDN infrastructure plane includes the forwarding devices that forward packets based on a set of streaming rules configured by the controller [23]. In the application plane, the network applications used by users such as adaptive routing, network management, traffic monitoring, intrusion detection system and intrusion prevention system [24] appear in this layer. Moreover, the same Figure shows that these SDN layers communicate among themselves using two interfaces; Southbound Interface (SI) and Northbound Interface (NI). Moreover, the controller uses the OpenFlow protocol to update the SDN switch tables, in order to change behavior and rules for each flow [22]. In SDN, switches are simple devices that are just involved for packet forwarding. They keep one or more flow tables with flow entries in them. The OpenFLow protocol uses p*acket_In* message to forward packets from switch to controller, and uses *packet_Out* message to forward action from controller to switch. Moreover, the controller may also decide to install flow rules into the switches along the way by sending *flow_Mod* messages.

Therefore, the SDN technology will be a good choice for the building of our proposal, especially since the detection of URL phishing will take place in the controller. This will give all embedded devices connected to the infrastructure plane the ability to detect phishing websites without any computational cost or power.

2.1.2. Features selection mechanism

Feature sub-group selection is a major difficulty in intrusion detection [25] due to the large dimensionality of the data features. As a result, the development of innovative ways to deal with feature selection remains an active area of research, mainly for feature detection. The goal of feature selection is to increase performance in areas including accuracy, data visualization and simplification for model selection, as well as reduce dimensionality to remove noise and irrelevant features [14]. The features used and data distribution have a significant impact on the performance of classifier systems. The latter tends to obtain local minima rather than the global minimum. The acquired results are often excellent, especially when the initial features are fairly far apart. This is due to the fact that algorithms can usually identify the primary category or class in a set of data. Common examples of feature selection technology such are Recursive Feature Elimination (RFE) [26], Particle Swarm Optimization (PSO) [12], Simulated Annealing (SA) [8], Genetic Algorithm (GA) [12], and Water Cycle (WC) [27]. In addition, another type of feature selection is called SelectKBest which is a method that selects features according to the highest scores for the chosen scoring function [28]. However, since the dataset we will be dealing with is of a linear type, RFE works very well. RFE is simply a reverse selection of predictors, it starts by creating a model based on the entire list of

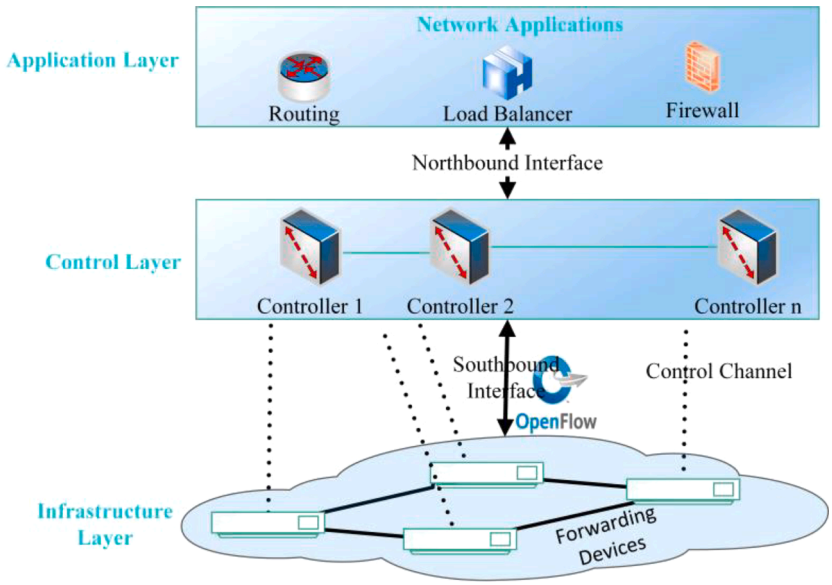


Fig. 1. The standard SDN architecture [22].

predictors and determining the relevance of each one. The model is then recreated, and the significance scores are calculated again after the least significant predictor(s) are eliminated. In fact, the analyst decides how many forecast sub-groups to examine and how big each sub-group should be. As a result, the sub-group size is a tuning parameter for RFE. To pick predictors based on relevance ratings, the sub-group size that enhances performance criteria is employed. The final model is then trained using the best subgroup [26]. Therefore, the strength of this algorithm is that it does not affect the correlation methods as the ranking criterion is calculated with information about a single feature. In addition, a criterion for rating a good feature is not necessarily a good criterion for rating a sub-group of features.

2.1.3. Machine learning techniques

The primary goal of this learning methodology is to develop a model that defines the relationships and dependencies between input features and predicted objective outcomes. Therefore, to improve the accuracy of the outcomes, feature selection methods are integrated with machine learning algorithms. Machine learning is a technique for improving or learning from an interpretation or experience without the need for manual setup [29]. It is divided into two sections: supervised and unsupervised learning. Two types of supervised learning are classification and regression. The major forms of classification are statistical learning (SVM and Bayesian), logic-based (Decision Tree), instance-based (KNN), and deep learning (Recurrent Neural Networks (RNN), Long Short Term Memory (LSTM), Convolutional Neural Networks (CNN)) [30]. In deep learning techniques, CNN is widely used because it has the advantage of automatically detecting of important features [31].

CNN is based on computational models and belongs to the family of Artificial Neural Networks. Artificial neural networks are highly inspired by the characteristics of biological neural networks. The simple patterns in data are identified by CNN deep learning technique that will be used to develop complex patterns in the subsequent layer [32]. For building CNN, there are two types of layers used. One is the pooling layers and the other is the convolution layer. The purpose of the convolution layer is to identify local combinations of features and the pooling layer merge similar features into a single feature. The first layer is the convolution layer that defines the best features and then the pooling layer defines the dimensions of the feature. Classification is controlled by the fully connected layer. The important factors affecting the CNN performance include the number of layers and filters. The number of layers must deal with the non-linearity and complexity of data that need to be analyzed. In deeper layers, more abstract features are extracted into the CNN that are defined by the number of layers. However, the number of features is determined by the number of filters applied at each stage. As the higher number of filters and layers increases, the computational complexity also increases. It is observed that more complex architectures result leads to poor prediction accuracy and an overtraining potential of a model. To overcome the problems, techniques named “batch regularization and “Dropout” leakage are implemented [33].

2.2. Websites phishing detection techniques

The issue of capturing URL Phishing websites is not new. There are traditional methods based on blacklist and whitelist of pre-defined URLs that have been added to the databases. However, what is new in this topic is the use of machine learning or deep learning algorithms, which provide a good and advanced opportunity to try to understand the phishing detection behavior and predict any new URL that could be a threat. Therefore, reliability in these techniques depend on the detection accuracy provided by these algorithms. On the other hand, another problem appears with the detection accuracy such as the cost and power that the devices need to be able to detect each URL phishing process. In this sub-section, we will review recent studies of these proposals to improve accuracy and cost of URL detection.

The authors in [19] used SDN technology to improve the detection of phishing websites. The optimization was based on the combination between traditional methods (blacklist and whitelist) with features extraction process, which relied on the URL and content of websites. The blacklist and whitelist are updated based on the output of features extraction of packets that come from users. The naïve Bayes classifier was used for the feature extraction process. Next, the controller updates the flow rule table and then sends it to switches to perform actions for each packet that matches those rules. If the packet does not match any value in the rule action table, the previous process will be repeated. Despite the improvements shown by their results, the proposed solution is large and complex. Similar to the same approach, the authors in [34] used machine learning based on stacking methods to detect the URL packets that are not blacklisted or whitelisted. Moreover, the authors in [5] used CNN along with SDN to improve the URL detection accuracy. The CNN is used in the controller to classify the URL in a signature-based database to different types of phishing attacks. Based on this classification, the coming packet inspection will either be forward directly to

destination or go into slow mode. In slow mode, it will perform more inspection to update the signature-based database. However, all three did not consider feature selection in their proposals despite its economic feasibility in reducing the training and improving performance.

Another different technique based solely on deep learning for URL phishing detection was proposed in [1], where the authors combined the CNN with a multi-head self-attention technique. A multi-head self-attention technique used to detect the internal dependency relationships between different characters of a single sentence. Each URL string will be used as input to the CNN to extract features. Then, the output of feature extraction process will be as input to a multi-head self-attention algorithm to find the relationships between URL characters though evaluating each feature weight [32], also used URL characters to embed into CNN algorithm to obtain high accuracy. Moreover, authors in [35] combined text features and semantic features through integrating independent recurrent neural network and a parallel joint neural network to analyze and detect malicious URLs. In [10,36], the authors used CNN and the Long Short-Term Memory (LSTM) algorithm to build a URL classification model. In addition, the authors in [37] proposed a heuristic classification framework based on term frequency-inverse document frequency, URL, and phish-hinted words for the URL feature selection process. This model was evaluated on the basis of 8 different ML algorithms, through which the Random Forest (RF) approach showed the highest accuracy. Its main drawback was the high training time. Furthermore, in the same context of using heuristic classifications, the authors proposed in [38] a new feature selection technique for ML algorithms. The new accumulative distribution function gradient algorithm is engineered as an automatic identifier for features selection to produce a combined set of core features, and then data perturbation and function perturbation techniques are implemented to these core features to generate the mixed set features. The output of the previous processes will be used as input to various classifier algorithms such as SVM, Naive Bayes, and RF for performance analysis. In [39], the authors used a neural network with feature validity value to detect phishing URLs. Furthermore, authors in [40] used two classifiers to detect URL attacks from page layout features, and in [9] authors used (rule-based) heuristic method along with RF for the same detection.

However, as it is noted, there is a shortage of studies dealing with the use of SDN along with machine learning algorithms in the process of detecting URLs phishing, although these technologies are the most promising field in the future for network management and ensuring its security.

3. Proposed FS-SDN scheme

In this section, a proposed website phishing detection architecture named FS-SDN is explained. The FS-SDN architecture utilizes SDN, deep learning, and feature selection techniques for improving URL phishing detection. The goal of using SDN, deep learning and feature selection technologies is to maintain continuous training process for modern phishing attacks, improve URL accuracy, and move training and implementation costs from users' devices to the network (controller and switches). Therefore, in this proposal, some modifications will be made to the OpenFlow protocol [23] in the SDN. Propose a technique called FS-CNN with the aim of increasing the efficiency and accuracy of attack detection. FS-CNN works inside the SD-Controller in order to provide the necessary training process, and its output is sent via the OpenFlow protocol to SD-Switches to be applied to the packages coming from the user devices, as will be explained as follows. Fig. 2 runs a detailed logical diagram for our proposal model. Moreover, we will assume that the proposed model will work in an Internet Service Provider (ISP) that uses SDN in a network environment. Therefore, all user devices are controlled by SD-Controller and the SD-Controller is responsible for URL packet filtering.

From Fig. 2, we can see that the process of FS-CNN technique will be maintained in the SD-Controller, and each URL request by users will go through the SD-Switches to the SD-controller, and then to the internet, and vice versa. The controller will perform the URL request for each user to detect whether this website is phishing or not. The FS-CNN will be used for this purpose through utilizing the power of the SD-Controller. That means we will move the cost of URL anomaly detection from all user devices to SD-Controller and that would be a great idea for devices with limited power and memory. The FS-CNN consists of various sequential features which are preprocessing, extraction of the feature

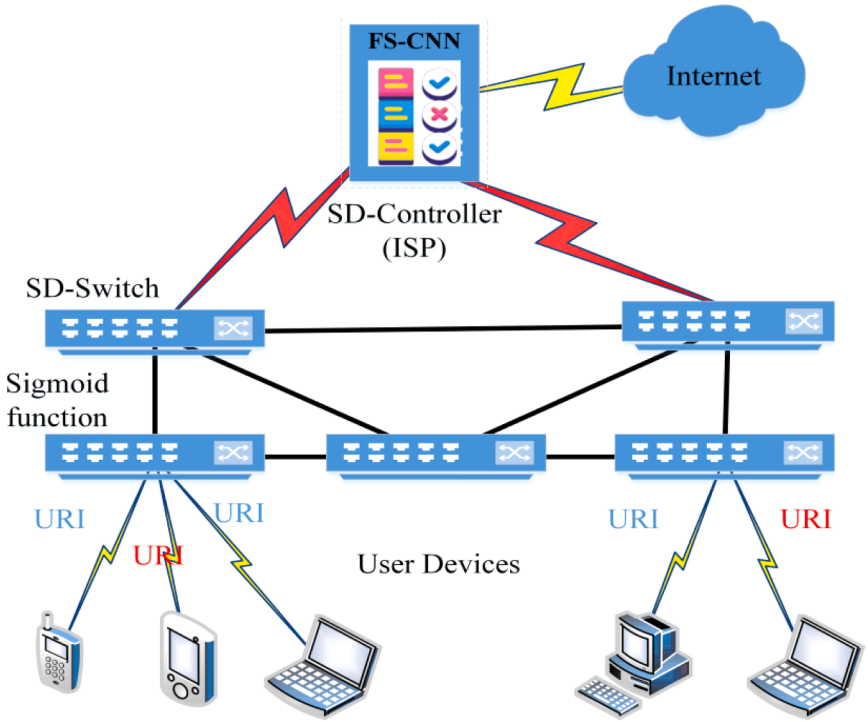


Fig. 2. FS-SDN proposal model.

matrix, feature learning and the classification. After completing these steps, each URL will show whether it is legitimate or phishing, and the controller will update the flow entries in switches with the new update of the sigmoid function parameters for the next matching flow, and the switches perform actions for each new URL packet that matches this function.

On the user side, the process begins when the user's data packet reaches at the switch. The switch first verifies if the URL packet flow is entered. Then, the sigmoid function is applied, if the resultant sign is positive, the data packet is forwarded directly. If its output is negative, the switch sends a URL packet to the controller to be processed using FS-CNN.

In the details of the FS-CNN scheme, it will be discussed as follows.

3.1. FS-CNN model

In this model, the URLs are described as legitimate or phishing. Thus, the gist of the solution is that the phishing site detection rate has to be very accurate. In this proposal, if the cost of phishing detection is high in terms of cost, we will give priority to accuracy since we use SDN technology. The following flow chart will explain the FS-CNN model.

Based on Fig. 3, the FS-CNN framework includes the sequential process which are the preprocessing URLs, extraction of the feature matrix, normalization, feature selection, feature learner and the classification. Each of these processes will be discussed in detail.

3.1.1. URL

URL consists of 6 parts that are interconnected to indicate the location of the resources as illustrated in Fig. 4. These parts are protocol, subdomain, second level of domain, top level of domain, path information and filename.

Phishers usually try to generate phishing URLs equivalent to the original site to deceive users online. However, the URL of legitimate websites is logged and the attackers cannot reuse it, but they use a URL similar to the original one. Therefore, the two types of difference (legitimate and phishing) are recognized based on a variety of URL features. In the dataset that we will be using, it contains 30 different features to differentiate them.

3.1.2. Preprocessing

The dataset that we will use collected from the Internet contains a large number of URLs that split between legitimate and phishing. Therefore, we applied this process (Preprocessing) to the dataset to remove unnecessary and ambiguous characters that are not useful for building efficient feature matrix. Ambiguous characters such as protocol (http or https) and subdomain (www) are removed from the URL characters before the feature matrix is extracted. Another operation that will be performed on the dataset is to convert all uppercase characters in the URL to lowercase URL, hence, our method ensures that uppercase alphabetic characters are the same as lowercase alphabetic characters.

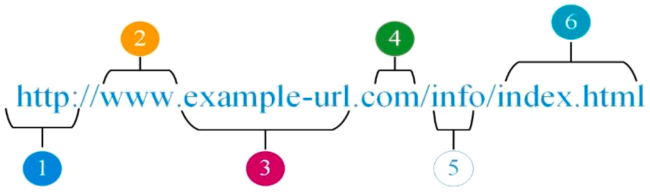


Fig. 4. The URL structure.

3.1.3. Feature extraction (one hot encoding)

The feature extraction techniques applied in this study include calculating the characters in the URLs [41]. As URLs are made up of alphabets (AZ), numbers (0–9) and alphanumeric ('~x223C ', '@', '#', '$', '%', '^', '&', '.", '<', '>', '.', '_', '+', '*', '!', '=', ':', ';', '-', ') ',' (', '|', '[', ']'). The feature matrix is extracted by counting the number of each character in the processed URLs. For example, the feature extraction from the "https://www.google.com" is given by the feature matrix in Table 1, ignoring ambiguous characters. Furthermore, any character out of what was previously mentioned is also ignored.

3.1.4. Normalization

This process is applied to enhance the quality of the feature matrix by converting it into the scale of 0–1.The z-score normalization method [42] is applied for this transformation which is given by the equation described below:

$$x_i = \frac{x_i - mean\ (x)}{std(x)} \tag{1}$$

where x_i is the feature x of observation i, $mean(x)$ is the average of feature x, and $std(x)$ is the standard deviation of feature x.

3.1.5. Feature selection and clustering

In this process, the relevant features set from the feature matrix is selected to reduce the dimension of the feature matrix. Since the dataset that we will use deals with linear relationship and statistical learning, the RFE with support machine vector (SVM) [15] is used for this proposed [43]. RFE is a popular feature selection method that involves the elimination of the feature based ranking. RFE attempts to find the best subset feature for a given machine learning algorithm. This is achieved by training the machine learning algorithm with features with the aim of measuring the features importance (highest weight). The feature subset with higher weight squares is fed to the SVM using the linear kernel to measure feature classification. The RFE-SVM is shown in Algorithm 1

Table 1
One hot encoding feature matrix.

C	e	g	L	m	o	.
1	1	2	1	1	3	1

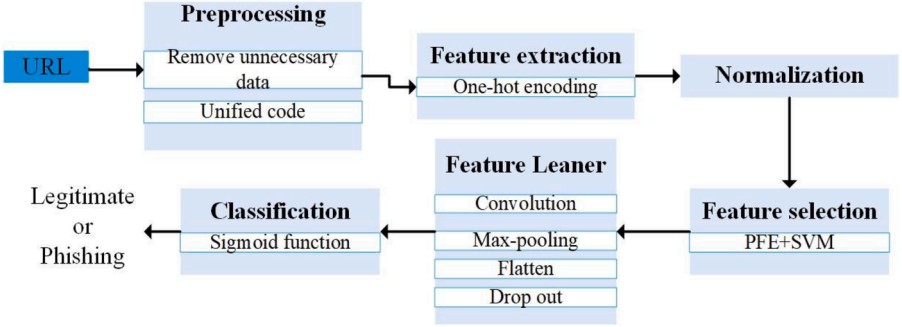


Fig. 3. The framework of FS-CNN.

Algorithm 1
RFE-SVM.

1. Input data $\{x_i, y_i\}$, $i = 1, \ldots, N$
2. Output **R** (ranked feature list)
3. S= {1,2,……, D}
4. ***R***=Empty
5. **While** $S \neq$ Empty **do**
6. X_j= Remaining of S
7. Train SVM on X_j
8. Calculate the ranking score $(C_k = w_k^2,\ k = 1, \ldots, |S|)$
9. p= *lowest* C_k
10. ***R(R**={p} U **R)** // add p into **R***
11. *S(S = x003D S\p) // remove feature p*
12. **end while**

[11]. where w represents the weight vector of the SVM.

3.1.6. Feature learner

This part of FS-CNN is responsible for extracting features from the feature matrix. The CNN is used with some additional features for this proposal [32]. The feature learner consists of a convolutional layer, a max pooling layer, a flatten layer and drop out. The structure of the feature learner is described in the following Figure.

Based on Fig. 5 diagram, the output of the feature matrix will be an input to the convolutional operator.

However, since the convolutional layer is responsible for the convolutional process that involves the combination of a feature matrix with a predefined initial filter [44], we have defined a one-dimensional input set of **64** elements. Therefore, this layer is responsible for integrating the input set with the feature matrix through the Rectifier Linear Unit (ReLU) activation function [45]. The activation function shows the feature matrix in hierarchical NNs, and the ReLU is characterized by its support for the sparsity of that matrix. The convolution function and ReLU activation function are shown in Eqs. (2) and (3).

$$s[t] = (x * w)[t] \tag{2}$$

$$s[t] = \max(x, s[t]) \tag{3}$$

where $s[t]$ is feature map, x is attribute data, and w is kernels.

After applying a convolutional layer, it will be the max-pooling turn to select the best output. The max-pooling establishes in-position invariance over larger local areas and down-samples the input objects [46]. Therefore, this process helps the convolutional procedure performance through selecting the most important features, and in the next conventional layer, the complexity of calculation feature maps will be reduced. Thus, we have applied a max pooling layer with a pool size of 2 in order to reduce the dimensions of the feature map. However, the pool size=2 was selected 2 through performing experimental results on a range of different values for pool size, and the best results were when the pool size is 2. In the next layer, flatten is used for vectorising of feature maps. In the classification stage, the flat input vector will be redirected to the network to produce a number in each output neuron [47]. This creates a single long layer from the convolutional layer. In the drop out layer, the over-fitting is prevented from fully connected layers.

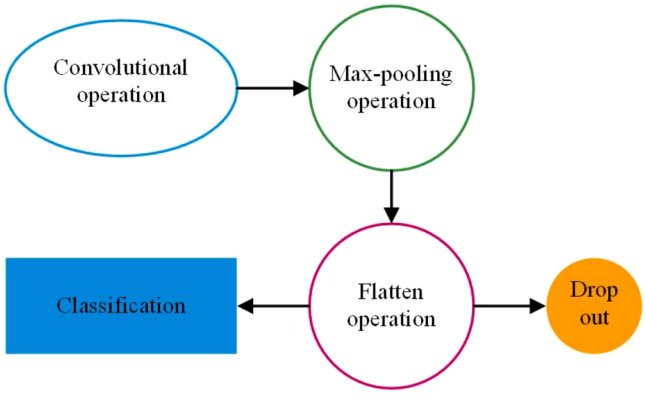


Fig. 5. The Feature Leaner procedure.

3.1.7. Classification

Finally, in the classification layer, the feature map is classified as legitimate or phishing using the sigmoid activation function.

$$s[t] = \frac{1}{(1 + e^{-(w^t\ x+b)})} \tag{4}$$

This classification is based on the value of *s[t]* which lies between 0 and 1. If $s[t] \leq 0.5$, the URL will be classified as phishing, otherwise it will be classified as legitimate.

However, the parameters of the sigmoid function of the switch will be redirected from the controller using *Packet_In* message, and the switch will update its flow table. Therefore, when new URL packets arrive at the switch, the switch will preprocess it for it and find its label. The URL inspection process is illustrated in Algorithm 2.

4. Experimental results

In this section, we will test the FS-SDN model using simulation software and compare it to the recent studies. The datasets were collected from different places, the legitimate URLs were from https://5000best.com/websites collected in May-20121 [48], while the phishing URLs were also collected from (https://www.phishtank.com) [3] in same time. The dataset contains 51,200 URL samples, of which about 40,000 are legitimate and the rest are phishing. Fig. 6 shows a sample of this raw dataset and its classification.

Moreover, our proposal compared to [1,9,10,36,40] based on the FS-CNN technique.

4.1. Simulation settings

We have simulated our proposal using Mininet tool [49] and Open Network Operating System (ONOS) controller [21]. Fig. 7 shows the FS-SDN simulation scenario.

Based on Fig. 7, the FS-SDN proposal scenario consists of one controller (ONOS), one SDN-switch called Open vSwitch (OVS), and one host. FS-CNN will run on ONOS and the data flow rules at that time will be updated in OVS based on the FS-CNN outputs. Next the user will check the URLs randomly to see how accurate the FS-CNN is. The work is run on a machine equipped with iMac21.1 model identifier, Chip Apple M1, Cores 8, and 8 GB LPDDR4-RAM. Furthermore, in the simulation scenario, we set the parameters of the FS-CNN as shown in Table 2.

4.2. Performance metrics

In this work, five performance measures are used to analyze FS-SDN technology. The targets are the highest number of True Positive (TP) and

Algorithm 2
URL inspection in SD-Switch.

1. Input the *URL packet*
2. do a pre-processing on the URL packet
3. do one hot encoding on the URL packet
4. do the normalization process for the URL packet
5. Find attribute data (x) of the URL packet
6. **Output**= Appliyg sigmoid activation function
7. **If** Output $\leq$ 0.5 **then**
8. drop the packet
9. **else**
10. Forward the packet to the Internet
11. **endif**

URLs	Category
http://bcpzornaseguras.com/bbvacontinentalpe-enlinea-20938209d23kjdh23d90238d23jwxj23/	Phishing
https://www.google.com/	Legitimate
https://www.microsoft.com/en-gb/	Legitimate
http://kelberdesigner.com/adesao/eng/2.html	Phishing
http://www.stopagingnews.com/wp-admin/js/wells/ibrowellsup/identity.php	Phishing
http://orientality.ro/RENNE/ourtime.com/ourtime.com/ourtime.html	Suspicious
http://bcpzornaseguras.com/	Suspicious
http://www.vinaros.org/locale/es/fb/	Phishing
http://www.vinaros.es/locale/es/fb/	Phishing
http://bcpzonasegura.viai1bcp.com/bcp/0peracionesnlinea/	Phishing
http://riquichichichi.tk/ptm/web/	Phishing
http://unitedstatesreferral.com/santos/gucci2014/gdocs/gucci.php?Acirc=A?A?=A?Auffc0=	Phishing
http://www.Legitimategovbr.com/SIIBC/	Phishing
http://201.73.146.167/teste/	Phishing
https://my.anglia.ac.uk/CookieAuth.dll?GetLogon?curl=Z2F&reason=0&formdir=3	Legitimate
https://uk.yahoo.com/?p=us	Legitimate

Fig. 6. Raw URLs and their category labels.

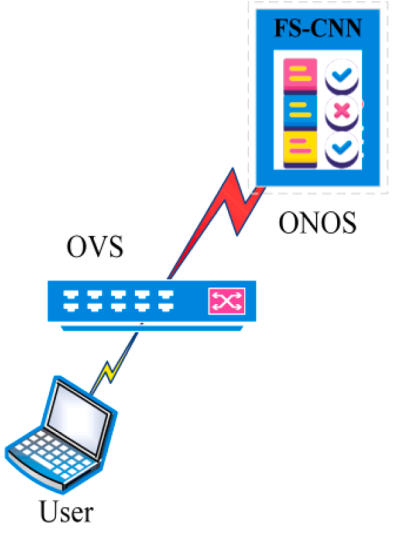


Fig. 7. The FS-SDN simulation scenario.

Table 2
FS-CNN parameters settings.

Parameter	value
Minibatch	45
Momentum	0.5
Learning coefficient	From 0.1 to 0.01
Kernel Filter	3 × 3
Conv1D	(None, 52,64)
Max pooling	(None, 26, 64)
Dropout	(None, 32)
Batch size	32
Flatten	(None, 1664)
Epochs	300
Dense	(None, 32)
Dense 1	(None, 16)
Dense 2	(None, 2)

True Negative (TN) and low number of False Negative (FN) and False Positive (FP). The TN number indicates that the URLs are valid, while the TP is the probability of the URLs being classified as phishing. Furthermore, FN shows the probability that the URLs are regular URLs, while FP represents the probability that the following regular URLs are phishing URLs. Recall indicates related items that have been successfully retrieved. It consists of a fraction dividing (TP) by the sum of (TP) and (TN). On the other hand, precision is used to measure how accurate phishing URLs are detected. Additionally, accuracy is the percentage of accurate predictions the model produces for all types of resulting predictions. Furthermore, the F1 score is used to comprehensively measure the accuracy of the model. The following equations represent all performance measures used in this analysis:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$F1 - score = \frac{2 * TP}{2 * TP + FP + FN} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$Precision = \frac{FP}{TP + FP} \tag{6}$$

Moreover, packet processing time is also taken into account when evaluating FS-SDN. The Percentage Value (PV) will measure accuracy and the F1 score against the optimal value of (1), and the FPR and FNR value against the optimal value of (0). Furthermore, memory used and number of URLs processed per second were also taken into account in the performance metrics.

4.3. Parameter settings for the FS-CNN

The dataset we used contains about 51,000 URLs, ranging in length from 15 to over 200 characters. Thus, in order to view the effect of URL length on FS-CNN performance metrics, we analysed it on the longest from 15 to 200 characters. As most of these URLs are 10–60 characters long, then average between 61 and 100 and then fall back 101–200 characters. Fig. 8 shows these analyses.

Based on Fig. 8, FS-CNN's performance metrics (accuracy, precision, recall, and F1-score) are distributed using different URL lengths. Each URL length (L) distributes its impact on these four metrics. In addition, the same Figure shows that the performance of the URL length when it is close to 100 and when it is 50 gives the best results, as their output values are in the range of 0.99–0.993 out of the optimal value (1). The rest of the URL lengths show good results, except for 15,20, 50 and greater than 150 characters, which show worse results, and the variance between the output of the performance metrics is growing. Likewise, Table 3 presents the results of these variations in performance metrics at different URL lengths in more detail.

However, from this explanation we can conclude, L can't be too short or too lengthy, according to the results. It demonstrates that short URLs

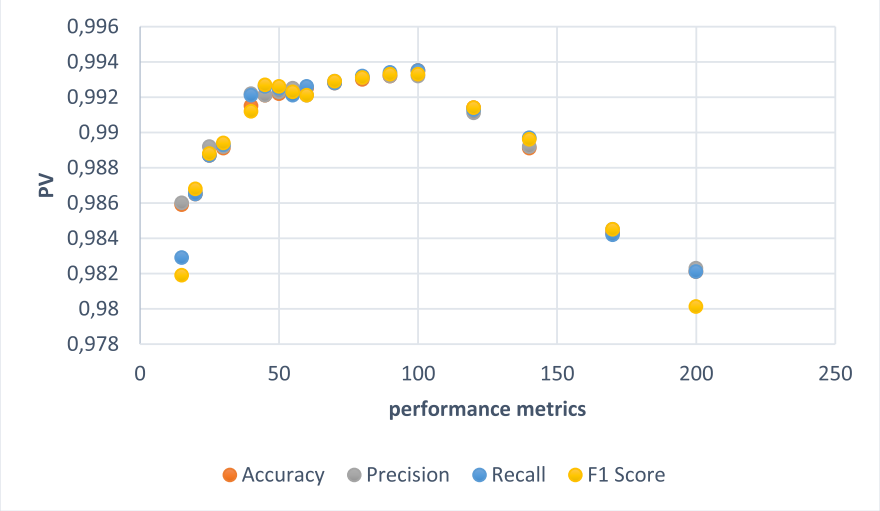


Fig. 8. Distribution of FS-CNN performance metrics with Different URL Lengths.

Table 3
FS-CNN Performance Analysis with Different URLs Length.

L	Accuracy	Precision	Recall	F1-Score
15	0.9859	0.986	0.9829	0.9819
20	0.9865	0.9866	0.9865	0.9868
25	0.9887	0.9892	0.9887	0.9888
30	0.9891	0.9892	0.9893	0.9894
35	0.9912	0.9914	0.9913	0.9916
40	0.9915	0.9922	0.9921	0.9912
45	0.9921	0.9921	0.9926	0.9927
50	0.9922	0.9923	0.9925	0.9926
55	0.9924	0.9925	0.9921	0.9923
60	0.9925	0.9921	0.9926	0.9921
70	0.9928	0.9929	0.9928	0.9929
80	0.993	0.9931	0.9932	0.9931
90	0.9932	0.9932	0.9934	0.9933
100	0.9935	0.9932	0.9935	0.9933
120	0.9914	0.9911	0.9913	0.9914
140	0.9891	0.9892	0.9897	0.9896
170	0.9843	0.9845	0.9842	0.9845
200	0.9821	0.9823	0.9821	0.980122

may be lacking in functionalities. For example, URLs that only contain an IP address are short, but they contain insufficient information for our form to learn characteristics from them. Long URLs also contain a lot of useless information that is intended to confuse consumers. The model would not be capable of detecting phishing URLs if it acquires a lot of worthless features.

Furthermore, feature selection is another important issue regarding the performance of our FS-CNN. Therefore, in order to provide a proper explanation for why RFE-SVM was chosen, we examined a group of feature selection methods on FS-CNN performance metrics. These feature selection methods include the SelectKBest-Chisquare [13], SelectKBest-Anova, SelectFromModel-L1 [50], and RFE-SVM. Fig. 9 shows these performance metrics for these algorithms based on the same dataset and the same FS-CNN parameter settings.

Based on the Figure, the RFE-SVM technology provides the highest results in three performance metrics with a slight difference from the K-Anova method. RFE-SVM shows the highest performance metrics due to its great ability to ignore weak features in small sampling problems. On the other hand, the same Figure shows the performance metrics of the FS-CNN method without using feature selection technology, which shows the lowest values. Table 4 also reviews the same performance

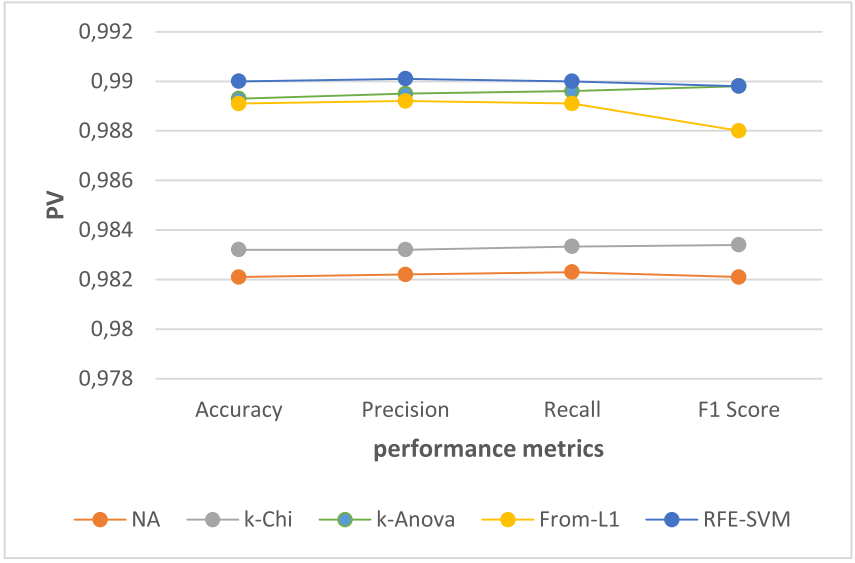


Fig. 9. FS-CNN performance metrics with different feature selection methods.

Table 4
FS-CNN performance analysis with different feature selection methods.

Feature Selection Method	Classifier	Accuracy	Precision	Recall	F1 Score
None	None	0.9821	0.9822	0.9823	0.9821
SelectKBest	Chi square	0.9832	0.9832	0.98333	0.9834
SelectKBest	Anova	0.9893	0.9895	0.9896	0.9898
SelectFromModel	L1	0.9891	0.9892	0.9891	0.988
Recursive elimination	SVM	0.9900	0.9901	0.9900	0.9898

measures for numerically different feature selection methods.

In Table 4, the first row describes the performance metrics without adding any feature selection method, while the rest of the rows describe the performance metrics for both feature selection methods and their respective algorithm. Furthermore, in order to obtain the best performance output, the architecture of the convolutional neural network is manipulated using different multipliers. The objective is to obtain a different size of the feature map relating to the current feature map (64, 32, 16). A multiplier of 0.5 will yield (32, 16, and 4), the Table below shows the performance result for different multipliers or feature maps.

Based on Table 5 outputs, the selected feature map size is (128,64,32,1) because it produced the highest performance metrics for the FS-CNN algorithm.

4.4. Results

In this section, a competency comparison between FS-CNN and the existing works will be discussed in relation to performance metrics. FS-CNN is measured by four of the existing works ([1,32,37,51,[52]]) and the outputs of our algorithm are the highest. Briefly, The authors in [1] used multi-headed self-attention feature selection with a CNN classifier and the authors in [32] used CNN. Moreover, the authors in [37] used RF and the work in [51] used Decision Tree (DT). Finally, the work in [52] used LSTM. In FS-CNN, FET-SVM is used as a preemptive step to improve feature extraction before it enters the classification process.

Fig. 10 shows the outputs of performance metrics for the six methods, and FS-CNN achieved the highest accuracy, Recall, and F1-Score. The reason for this is due to the set of filtering and regular steps of that algorithm, which includes preprocessing, extraction of the feature matrix, feature learning and the classification. In addition to the use of RFE-SVM, which improved the output by defining the important dataset features. On the other hand, these steps are known to require hardware and processor power, so this problem was solved by using the SDN Controller. While the outputs of [32] achieved results close to that of FS-CNN, also because similar procedures were adopted using CNN without using the feature selection method. Bahnsen et al. [52] also gave a good results compared to the rest of the techniques used due to the use of method LSTM.

Furthermore, the same Figure shows the output of the performance measures for [37] which appears to be the worst because no performance improvements or input processing assistance is used for those algorithms. Besides, RF classifier was used with the feature selection method, as the RF algorithm is fast and good in the classification process, but it cannot be compared to the CNN algorithm in terms of efficiency in a big dataset. To further clarify and present the output ratios of the methods used, Table 6 reviews the comparisons of these methods.

Table 5
Test different feature map sizes with performance metrics.

Feature Map Size	Accuracy	Precision	Recall	F1-Score
(16, 8, 4,1)	0.9869	0.9876	0.9869	0.9870
(32, 16, 8,1)	0.9882	0.9888	0.9882	0.9884
(64, 32, 16,1)	0.9908	0.9911	0.9908	0.9909
(128, 64, 32,1)	0.9912	0.9915	0.9912	0.9913
(256, 128, 64,1)	0.9910	0.9913	0.9910	0.9910

With regard to the memory size that were used in the techniques, it has been shown in Fig. 11. The Figure showed that the Sahingoz et al. [51] and Bahnsen et al. [52] are the least consuming methods for memory space. As the Sahingoz et al. [51] used the DT classifier, and the Sahingoz et al. [51] used the LSTM, these techniques are known for their simplicity and speed. FS-CNN had an average consumption size of 570 MB, which is a reasonable due to the use of complex technologies in the training process. The FS-CNN algorithm does not attention about memory space, because it uses SD-Controller for training process, and the training process is independent of the users' devices connected to the SD-Switches. Therefore, the FS-CNN is to raise the accuracy state in the first place and then sustain a regular rate in the training processes to make the phishing follow-up process up-to-date without any training burden on the users' devices. We also find Xiao et al. [1] is the most memory-intensive method due to the complex structures used in this technique and the complexity of storing the model parameters.

In addition to the performance of the comparison methods in the number of URLs processed per second, Fig. 12 shows this performance.

The Sahingoz et al. [51] showed the best performance in terms of processing 580 URLs per second, due to the fact that DT is one of the fastest classifiers in processing. Furthermore. we find that the FS-CNN is able to process 460 URLs/s, and Xiao et al. [1] achieved the lowest number by 260 URLs/s. This reflects the method's speed on processing, not its accuracy. As the increase in model parameters and the complexity of training processes leads to an increase in the training time. We should also note that not necessarily increasing the training time will lead to an increase in accuracy but it is assumed that increased validation and organization of the training process are among the things that increase technique accuracy.

On the other hand, the FS-CNN implementation process takes place collaboratively between the controller and the switch. Therefore, one of our goals is to transfer the URL verification to the switch. The controller implements the FS-CNN process and then forwards its output to the switch, and the switch performs the process of obtaining every URL coming from the user. Thus, it is important to know how long it takes to inspect the URL packets coming from the user. Fig. 13 shows the effect of adding a packet inspection to the switch's working time.

Fig. 13 shows the times it takes for the switch to inspect incoming URL packets from the user. The time increases as the number of packages increases regularly. The processing time increased from 77 ms when the number of packets was 100 to 1129 ms when the number of packets reached 5000. It should be noted that the sizes of the URL were taken at random when collecting the results from the switch.

Furthermore, to illustrate the relationship between URL size and inspection time, Fig. 14 reviews this relationship.

Fig. 14 also shows that the amount of increase in the size of the URL leads to an increase in the switch processing time on a regular basis, because URL size also increases regularly. The switch inspection time increased from 53 ms when the URL was 15 characters long to 134 ms when the URL length reached 100.

Traditional phishing detection solutions use inline inspection procedures such as the Intrusion Prevention System (IPS) or the proxy service based on static string matching in traditional Intrusion Deduction (IDS) as in SNORT [53]. However, in previous studies on the use of different techniques to improve the performance of URL phishing detection as in [32,36,37,51]. Applying these techniques in real applications is difficult due to the fact that it takes a long time to detect phishing. Furthermore, the special rules for detecting these URLs are static and cannot be changed dynamically to deal with real-time phishing. Therefore, in this work, we have attempted to transfer these costs to the SDN to be shared between the SD-Controller and the SD-Switches, in the hope of performing and optimizing these tasks.

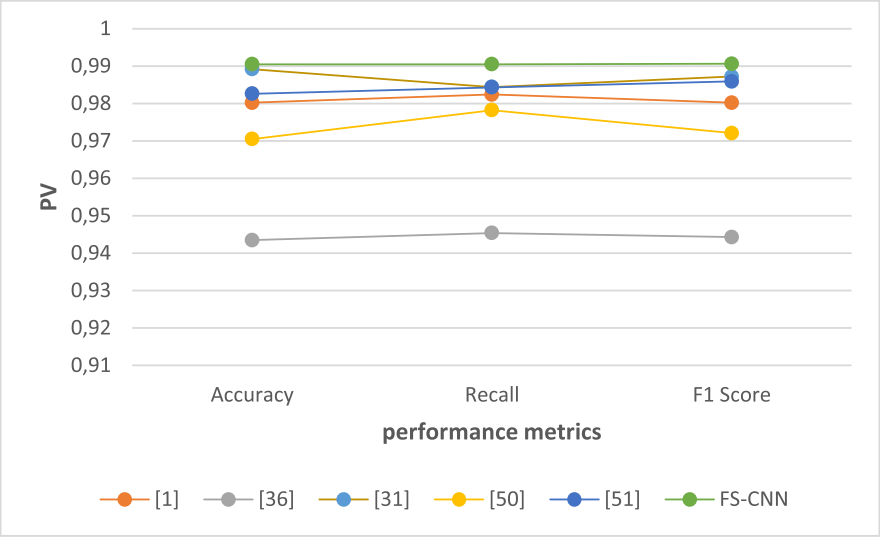


Fig. 10. Output of performance metrics in different methods.

Table 6
Comparison of the FS-CNN and Other Benchmarking Methods.

Method	Accuracy	Recall	F1-Score
Xiao et al. [1]	0.9802	0.9824	0.9802
Rao et al. [37]	0.9435	0.9454	0.9443
Wei et al. [32]	0.9892	0.9844	0.9872
Sahingoz et al. [51]	0.9705	0.9782	0.9721
Bahnsen et al. [52]	0.9826	0.9843	0.9859
FS-CNN	0.9905	0.9905	0.9906

5. Conclusion and future work

Phishing website attacks are a significant hazard to internet users, and they have been on the rise in recent years. Therefore, we need to improve anti-phishing software. Machine learning algorithms are one of the most common technologies used in this domain, but with the development of the Internet of Things, we need to develop these functions to keep pace with new technologies. Thus, in this study, we have improved the process of phishing detection by proposing a new approach that combines RFE-SVM and CNN method in sequential features including (sequential features which are preprocessing, extraction of the feature matrix, feature learning and the classification) along with SDN technology. The combination between RFE-SVM and CNN gave birth to a new approach called FS-CNN, and FS-CNN runs on the SDN controller. The output of FS-CNN will be forwarded to the SD-Switch to inspect every packet requested for a URL. The purpose of using the SDN is to sustain the training for new types of phishing attacks, update phishing operations in the controller, and completely separate them from users' devices. Furthermore, users' devices are updated with all kinds of phishing attacks at no additional cost. The performance metrics used to evaluate the functionality of the FS-CNN are Accuracy, Precision, Recall, and F1-Score. For SD-Switch we evaluate it through calculating the URL packets inspection time. The Mininet and ONOS controller were used to simulate our simulation scenario. FS-CNN achieved 99.5% in phishing detection accuracy and higher than the existing works that we compared.

In future work, we will analyze the efficiency of FS-SDN by adding wireless networks and various types of embedded devices to the work environment. Moreover, FS-SDN power consumption analysis on both controller and switches. Furthermore, the queued packets inspection process should be reconfigured in order to speed up the SD-Switch process by making it run in parallel.

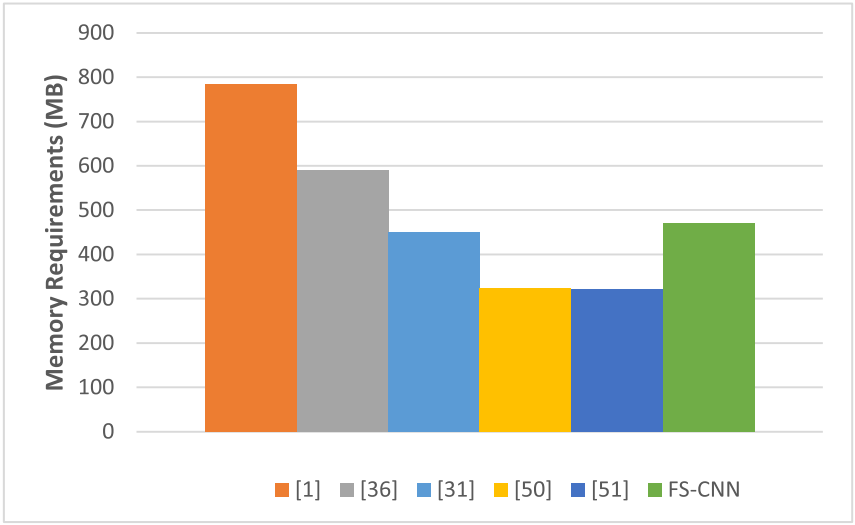


Fig. 11. Comparison of the memory used by each method.

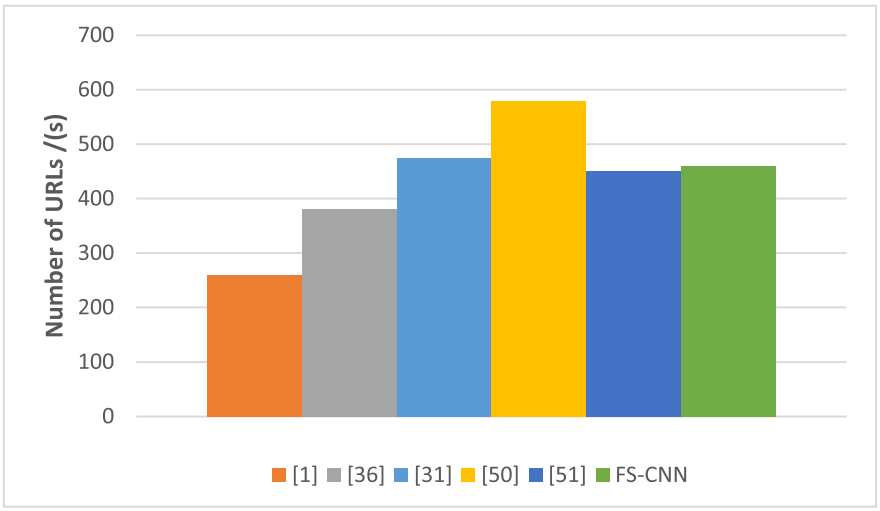


Fig. 12. Comparison of the number of URLs treated every second by each method.

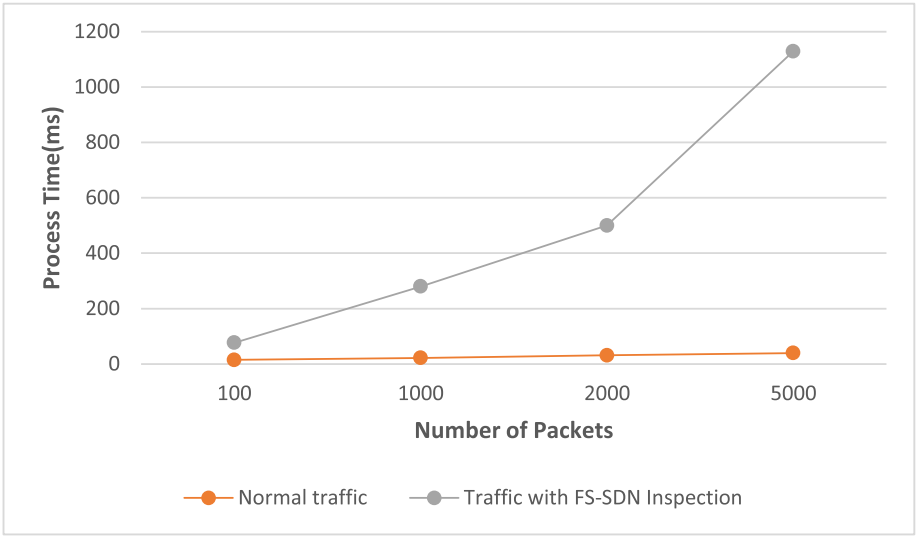


Fig. 13. Processing of the SD-Switch inspection time varies with the number of packets.

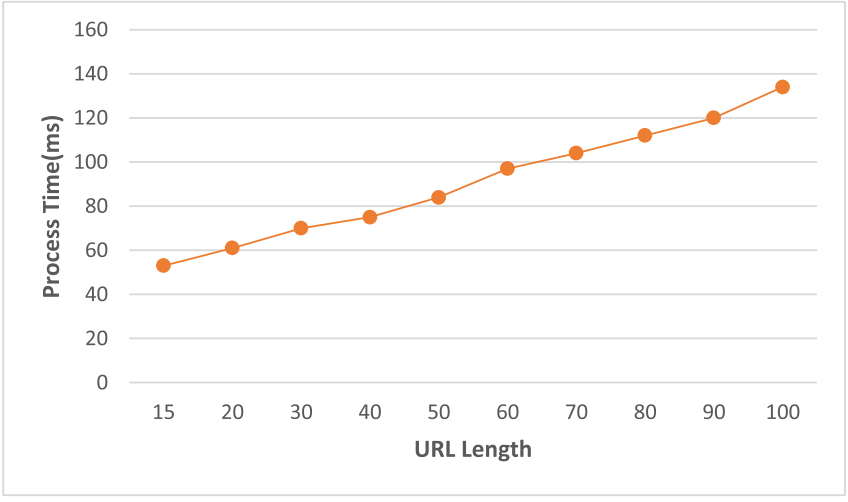


Fig. 14. Relationship between URL lengths and switch process time.

CRediT authorship contribution statement

Raniyah Wazirali: Data curation, Writing – original draft, Software, Validation. **Rami Ahmad:** Conceptualization, Methodology, Supervision, Writing – original draft. **Ashraf Abdel-Karim Abu-Ein:** Writing – review & editing, Investigation.

Declaration of Competing Interest

None.

References

[1] X. Xiao, D. Zhang, G. Hu, Y. Jiang, S. Xia, CNN–MHSA: a convolutional neural network and multi-head self-attention combined approach for detecting phishing websites, Neural Netw. 125 (2020) 303–312, https://doi.org/10.1016/j.neunet.2020.02.013.

[2] A.K. Jain, B.B. Gupta, A survey of phishing attack techniques, defence mechanisms and open research challenges, Enterp. Inf. Syst. 00 (00) (2021) 1–39, https://doi.org/10.1080/17517575.2021.1896786.

[3] "PhishTank," *Cisco Talos Intelligence Group*. https://www.phishtank.com/ (accessed May 21, 2021).

[4] "Spam and phishing in Q1 2021," *Kaspersky*. https://securelist.com/spam-and-phishing-in-q1-2021/102018/ (accessed Jun. 30, 2021).

[5] T. Chin, K. Xiong, C. Hu, Phishlimiter: a phishing detection and mitigation approach using software-defined networking, IEEE Access 6 (2018) 42513–42531, https://doi.org/10.1109/ACCESS.2018.2837889.

[6] Y. Zhang, J.I. Hong, L.F. Cranor, Cantina: a content-based approach to detecting phishing web sites, in: Proceedings of the 16th International World Wide Web Conference WWW2007, 2007, pp. 639–648, https://doi.org/10.1145/1242572.1242659.

[7] S. Sheng, B. Wardman, G. Warner, L.F. Cranor, J. Hong, C. Zhang, An empirical analysis of phishing blacklists, in: Proceedings of the CEAS 2009 6th Conference Email Anti-Spam, 2009.

[8] S.W. Lin, K.C. Ying, C.Y. Lee, Z.J. Lee, An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection, Appl. Soft Comput. J. 12 (10) (2012) 3285–3290, https://doi.org/10.1016/j.asoc.2012.05.004.

[9] M. Al-Janabi, E. De Quincey, P. Andras, Using supervised machine learning algorithms to detect suspicious URLs in online social networks, in: Proceedings of the 2017 IEEE/ACM International Conference Advance Society Networks Analysis Mining, ASONAM 2017, 2017, pp. 1104–1111, https://doi.org/10.1145/3110025.3116201.

[10] M.A. Adebowale, K.T. Lwin, M.A. Hossain, Deep learning with convolutional neural network and long short-term memory for phishing detection, in: Proceedings of the 2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), 2018, pp. 1–8, https://doi.org/10.1109/SKIMA47702.2019.8982427. Aug. 2019, no. March.

[11] Z. Rustam, S.A.A. Kharis, Comparison of support vector machine recursive feature elimination and kernel function as feature selection using support vector machine for lung cancer classification, J. Phys. Conf. Ser. 1442 (1) (2020), https://doi.org/10.1088/1742-6596/1442/1/012027.

[12] X. Lu, D. Han, L. Duan, Q. Tian, Intrusion detection of wireless sensor networks based on IPSO algorithm and BP neural network, Int. J. Comput. Sci. Eng. 22 (2–3) (2020) 221–232, https://doi.org/10.1504/IJCSE.2020.107344.

[13] "Feature selection," *Scikit-Learn*. https://scikit-learn.org/stable/modules/feature_selection.html#univariate-feature-selection (accessed Jul. 03, 2021).

[14] M. Alweshah, S. Alkhalaileh, D. Albashish, M. Mafarja, Q. Bsoul, O. Dorgham, A hybrid mine blast algorithm for feature selection problems, Soft Comput. 25 (1) (2021) 517–534, https://doi.org/10.1007/s00500-020-05164-4.

[15] V.N. Vapnik, An overview of statistical learning theory, IEEE Trans. Neural Netw. 10 (5) (1999) 988–999, https://doi.org/10.1109/72.788640.

[16] O.A. Khashan, R. Ahmad, N.M. Khafajah, An automated lightweight encryption scheme for secure and energy-efficient communication in wireless sensor networks, Ad Hoc Netw. 115 (2021), 102448, https://doi.org/10.1016/j.adhoc.2021.102448. Apr.

[17] R. Wazirali, R. Ahmad, A. Al-Amayreh, M. Al-Madi, A. Khalifeh, Secure watermarking schemes and their approaches in the iot technology: an overview, Electron. (Basel) 10 (14) (2021) 1744, https://doi.org/10.3390/electronics10141744. Jul.

[18] "Software-Defined Networking (SDN) Definition," *ONF*. https://www.opennetworking.org/sdn-definition (accessed Jun. 08, 2021).

[19] 15 K. Archana Janani, V. Vetriselvi, R. Parthasarathi, G. Subrahmanya VRK Rao, et al., in: S. Smys, R. Bestak, J.I.Z. Chen, et al. (Eds.), An Approach to URL Filtering in SDN, Springer Singapore, Singapore, 2019, pp. 217–228, 15.

[20] OpenDaylight, The Linux Foundation Project, 2021. https://www.opendaylight.org/(accessed. Jun. 08.

[21] "Open Network Operating System (ONOS)," *onosproject*. https://docs.onosproject.org/ (accessed Jun. 07, 2021).

[22] R. Ahmad, E.A. Sundararajan, A. Khalifeh, A survey on femtocell handover management in dense heterogeneous 5G networks, Telecommun. Syst. (2020), https://doi.org/10.1007/s11235-020-00718-1. Oct.

[23] R. Wazirali, R. Ahmad, S. Alhiyari, SDN-openflow topology discovery: an overview of performance issues, Appl. Sci. 11 (15) (2021) 6999, https://doi.org/10.3390/app11156999. Jul.

[24] S. Lim, J. Ha, H. Kim, Y. Kim, S. Yang, A SDN-oriented DDoS blocking scheme for botnet-based attacks, in: Proceedings of the 2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN), 2014, pp. 63–68, https://doi.org/10.1109/ICUFN.2014.6876752. Jul.

[25] M. Wang, Y. Lu, J. Qin, A dynamic MLP-based DDoS attack detection method using feature selection and feedback, Comput. Secur. 88 (2020), https://doi.org/10.1016/j.cose.2019.101645.

[26] S. Ustebay, Z. Turgut, M.A. Aydin, Intrusion Detection System with Recursive Feature Elimination by Using Random Forest and Deep Learning Classifier. Int. Congr. Big Data, Deep Learn. Fight. Cyber Terror. IBIGDELFT 2018 - Proc, 2019, pp. 71–76, https://doi.org/10.1109/IBIGDELFT.2018.8625318.

[27] R. Ahmad, R. Wazirali, Q. Bsoul, T. Abu-Ain, W. Abu-Ain, Feature-selection and mutual-clustering approaches to improve dos detection and maintain wsns' lifetime, Sensors 21 (14) (2021) 4821, https://doi.org/10.3390/s21144821. Jul.

[28] N.Ö. Özcan Şimşek, A. Özgür, F. Gürgen, A novel gene selection method for gene expression data for the task of cancer type classification, Biol. Direct 16 (1) (2021), https://doi.org/10.1186/s13062-020-00290-3.

[29] R. Wazirali, R. Ahmad, Machine learning approaches to detect DoS and their effect on WSNs lifetime, Comput. Mater. Contin. 70 (3) (2022) 4922–4946, https://doi.org/10.32604/cmc.2022.020044.

[30] R. Vinayakumar, M. Alazab, K.P. Soman, P. Poornachandran, A. Al-Nemrat, S. Venkatraman, Deep learning approach for intelligent intrusion detection system, IEEE Access 7 (c) (2019) 41525–41550, https://doi.org/10.1109/ACCESS.2019.2895334.

[31] R. Wazirali, R. Ahmed, Hybrid feature extractions and CNN for enhanced periocular identification during Covid-19, Comput. Syst. Sci. Eng. 41 (1) (2022) 305–320, https://doi.org/10.32604/csse.2022.020504.

[32] W. Wei, Q. Ke, J. Nowak, M. Korytkowski, R. Scherer, M. Woźniak, Accurate and fast URL phishing detector: a convolutional neural network approach, Comput. Netw. 178 (April) (2020), https://doi.org/10.1016/j.comnet.2020.107275.

[33] L. Alzubaidi, et al., Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions, 8, Springer International Publishing, 2021.

[34] M. Miao, B. Wu, A flexible phishing detection approach based on software-defined networking using ensemble learning method, ACM Int. Conf. Proc. Ser. (2020) 70–73, https://doi.org/10.1145/3407947.3407952.

[35] J. Yuan, G. Chen, S. Tian, X. Pei, Malicious URL detection based on a parallel neural joint model, IEEE Access 9 (2021) 9464–9472, https://doi.org/10.1109/ACCESS.2021.3049625.

[36] P. Yang, G. Zhao, P. Zeng, Phishing website detection based on multidimensional features driven by deep learning, IEEE Access 7 (2019) 15196–15209, https://doi.org/10.1109/ACCESS.2019.2892066.

[37] R.S. Rao, T. Vaishnavi, A.R. Pais, CatchPhish: detection of phishing websites by inspecting URLs, J. Ambient. Intell. Humaniz. Comput. 11 (2) (2020) 813–825, https://doi.org/10.1007/s12652-019-01311-4.

[38] K.L. Chiew, C.L. Tan, K.S. Wong, K.S.C. Yong, W.K. Tiong, A new hybrid ensemble feature selection framework for machine learning-based phishing detection system, Inf. Sci. 484 (2019) 153–166, https://doi.org/10.1016/j.ins.2019.01.064.

[39] E. Zhu, Y. Chen, C. Ye, X. Li, F. Liu, OFS-NN: an effective phishing websites detection model based on optimal feature selection and neural network, IEEE Access 7 (2019) 73271–73284, https://doi.org/10.1109/ACCESS.2019.2920655.

[40] J. Mao, et al., Detecting phishing websites via aggregation analysis of page layouts, Proc. Comput. Sci. 129 (2018) 224–230, https://doi.org/10.1016/j.procs.2018.03.053.

[41] D. Harris, S. Harris, Digital Design and Computer Architecture, 2nd ed., Morgan Kaufmann, 2013.

[42] C. Cheadle, M.P. Vawter, W.J. Freed, K.G. Becker, Analysis of microarray data using Z score transformation, J. Mol. Diagn. 5 (2) (2003) 73–81, https://doi.org/10.1016/S1525-1578(10)60455-2.

[43] X. Huang, L. Zhang, B. Wang, F. Li, Z. Zhang, Feature clustering based support vector machine recursive feature elimination for gene selection, Appl. Intell. 48 (3) (2018) 594–607, https://doi.org/10.1007/s10489-017-0992-2.

[44] A. Aljofey, Q. Jiang, Q. Qu, M. Huang, J.P. Niyigena, An effective phishing detection model based on character level convolutional neural network from URL, Electron. (Basel) 9 (9) (2020) 1514, https://doi.org/10.3390/electronics9091514. Sep.

[45] K. Eckle, J. Schmidt-Hieber, A comparison of deep networks with ReLU activation function and linear spline-type methods, Neural Netw. 110 (2019) 232–242, https://doi.org/10.1016/j.neunet.2018.11.005.

[46] J. Nagi, et al., Max-pooling convolutional neural networks for vision-based hand gesture recognition, in: Proceedings of the IEEE International Conference Signal Image Processing Applications ICSIPA, 2011, pp. 342–347, https://doi.org/10.1109/ICSIPA.2011.6144164.

[47] G. Çınarer, B.G. Emiroğlu, R.S. Arslan, A.H. Yurttakal, Brain tumor classification using deep neural network, Adv. Sci. Technol. Eng. Syst. 5 (5) (2020) 765–769, https://doi.org/10.25046/AJ050593.

[48] "5000 BEST WEBSITES." http://5000best.com/websites (accessed May 21, 2021).

[49] "Mininet." http://mininet.org/ (accessed Jul. 11, 2021).

[50] "sklearn.feature_selection.SelectFromModel," *Scikit-Learn*. https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html (accessed Jul. 03, 2021).

[51] O.K. Sahingoz, E. Buber, O. Demir, B. Diri, Machine learning based phishing detection from URLs, Expert Syst. Appl. 117 (2019) 345–357, https://doi.org/10.1016/j.eswa.2018.09.029.
[52] A.C. Bahnsen, E.C. Bohorquez, S. Villegas, J. Vargas, F.A. Gonzalez, Classifying phishing URLs using recurrent neural networks, eCrime Res. Summit, eCrime (2017) 1–8, https://doi.org/10.1109/ECRIME.2017.7945048.
[53] M. Roesch, Snort – lightweight intrusion detection for networks, in: Proceedings of the LISA 13th Systems Administration Conference, 1999, pp. 229–238.

Raniyah Wazirali received the B.Sc. degree in computer science from Taif University, Taif, Saudi Arabia, in 2008, and the M.Sc. degree in Information Technology and the Ph.D. degree in Software Engineering from the University of Technology, Sydney (UTS), in 2013 and 2017, respectively. She is currently an Assistant Professor in Information Technology Department, Saudi Electronic University (SEU). Her current research interests include steganography, cryptography, evolutionary optimization and machine learning techniques. She is currently involved in intrusion detection algorithms, detection anomalies in DoS, person identification, AI algorithms, focusing particularly on machine learning and optimization with a willingness to implement them in a context of decision making and solving combinatorial problems in real-world projects.

Rami Ahmad is an assistant professor at Sebha University, Libya. He received his BSc in Computer Science in 2002 from Al-Albyat University, Jordan and MSc in Intelligent system in 2008 from University of Utara, Malaysia. He obtained his PhD in Computer Science from Networking and Security Lab of the National University of Malaysia (UKM), Malaysia in 2017. His-current research interests include Machine Learning, Computer Network, Network Security, Information Security, and Wireless Networks.

Ashraf Abdel-Karim Abu-Ein received the M.Sc. and PhD degrees in 2007 from the "National Technical University of Ukraine". He is currently working as Associate Professor in Computer Engineering and Computer Networks department at Al-Balqa' University. His-current research interest is including computer networking, image processing, and computer graphics.