IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ICRTTC '23

International Conference on Recent Trends and Technologies in Computing

Department of Computer Science and Engineering, Arasu Engineering College

Vol. 12, Special Issue 1, April 2023

Hybrid Algorithm Based Framework For Phishing Website Detection

M.Rajeshkanth[1], S.Sameer Mohammed [2], R.Saravana kumar[3]

Department of Computer Science & Engineering, M.I.E.T. Engineering College, Trichy[1,2,3]

Abstract: Cyber attackers share phishing links through e-mail, text messages, and social media platforms. They use social engineering skills to trick users in to visiting phishing websites and entering personal information and they steal our personal information and it is used to defraud the trust of regular websites or financial institutions to obtain illegal benefits. With the development and applications of machine learning technology, many machine learning-based solutions for detecting phishing have been proposed. Some solutions are based on the features extracted by rules and some of the features need to rely on third-party services, which will cause instability and time-consuming issues in the prediction service. In this paper, we propose a deep learning-based hybrid framework for detecting phishing websites. We have implemented the framework as a browser plug-incapable of determining whether there is a phishing risk in real-time when the user visits a webpage and give a warning message. The real-time prediction service combines multiple strategies to improve accuracy, reduce false alarm rates, and reduce calculation time, including white list filtering, blacklist interception, and machine learning (ML) prediction. In the ML prediction module, we compared multiple machine learning models using several datasets. From the experimental results, the LSTM-XGBOOST model obtained the highest accuracy of 99.18%, demonstrating the feasibility of the proposed solution.

Index terms: Phishing detection, machine learning, deep learning, LSTM-XGBOOST, Web browser extension.

I. INTRODUCTION

Internet services have brought tremendous changes to people's lives. Most online services manage users through a membership system, and individual users need to register and log in to obtain these personalized services. Therefore, people need to provide personal information when enjoying these convenient and efficient services. In a secure network environment, the transmission and storage of information are protected by network security technology. However, there are many cybercriminals who use various methods to attack and steal personal information.

Phishing is one of the cyber attack methods that simulate a regular website to trick users into providing personal information. Since the emergence of phishing attacks more than ten years ago, network security experts have been using technical methods to intercept attacks. Attack technology and anti-attack technology are constantly changing and improving. Unfortunately, there is no effective technology that can completely prevent phishing attacks. From the network security reports in recent years, we can see that the economic losses caused by phishing attacks are huge [2]. According to the phishing activity report from APWG, there are more than 100,000 phishing link several month, and it has been a growing trend in the past year [3].The 2020 annual report from the Internet crime complaint centre showed that the economic loss caused by phishing attacks was over $54million [4].

Commonly used means of spreading phishing links are e-mails, text messages, and social media platforms. The content of the copy edited by the attacker through social engineering means that the user is very eager to click on the phishing link after receiving the information. Therefore, when the phishing link is accessed in the browser, detecting the risk through network security technology, and alerting the user is a very effective anti-attack technology to prevent the user from leaking personal information. The core of traditional methods of detecting phishing URLs is based on rules. The generation of these rules can be summarized as parsing features from URL and web an empirical threshold. It can be seen from the academic research report that the number of effective rules is within 100 [5], [6].

In this paper, we propose a deep learning-based frame work to detect phishing links in a real-time web browsing environment. We developed a browser plug-in to receive client information, call the background prediction service, and show the prediction results to users. When the URL of the current tab of the browser is predicted to be a phishing link, the current page will receive an obvious warning prompt. The prediction result is obtained by the core prediction service calling a trained machine learning model. We introduced multiple models with multiple data sets for comparison and backup. It is concluded from the experimental results that the RNN-GRU model obtains the highest accuracy rate of 99.18, which is better than SVM, Logistic Regression, Random Forest. The contributions of this paper are: A deep learning-based framework for detecting phishing URLs. We trained and tested the models using seven custom datasets generated from four existing data sources, and we achieved the highest accuracy of

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ICRTTC '23
International Conference on Recent Trends and Technologies in Computing
Department of Computer Science and Engineering, Arasu Engineering College
Vol. 12, Special Issue 1, April 2023

99.18% with the RNN-GRU model. A prototype implementation of the proposed framework as a Chrome browser extension. We organized the rest of the paper as follows: Section II summarizes the related work focusing on deep learning models and real-time frameworks. Section III presents the design and architecture of the proposed framework. Section IV discusses the prototype implementation, including some open source frameworks, services, and tools that we have utilized. Experimental results and analysis are reported in section V. Finally, Section VI concludes the paper and offers ideas for future work.

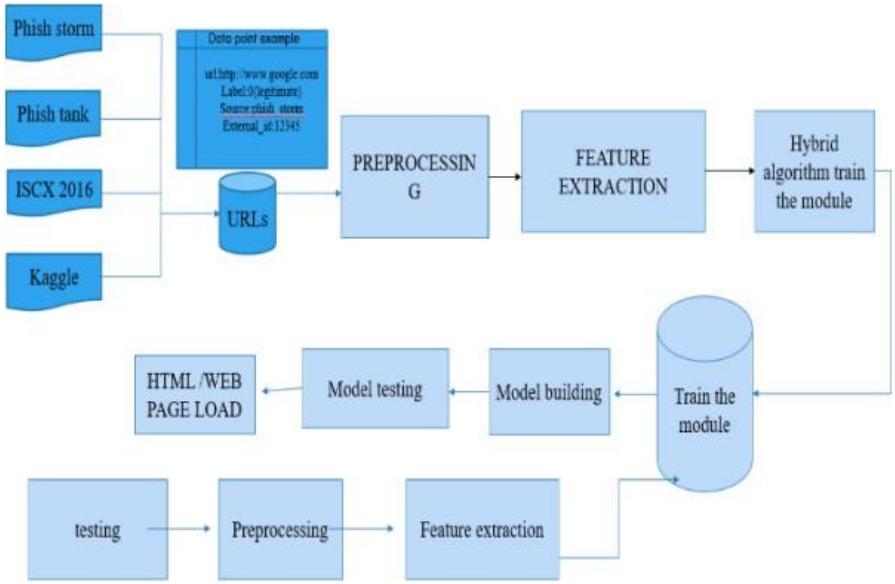

FIGURE1. Architecture of the deep learning-based hybrid framework for detecting phishing URLs.

II. RELATEDWORK

Phishing attacks represent a serious problem, and the technology for detecting and intercepting phishing attacks is constantly evolving. It is the most accurate and fast way to filter good URLs through the white list and block phishing URLs through the blacklist. However, the list method cannot detect new phishing links, and because of the low cost of creating a phishing URL, the attacker does not rely on using the same phishing link multiple times. Many research reports based on machine learning have been published, and high accuracy results have been obtained in experiments. However, in the actual network environment, there are still many victims of phishing attacks every year, causing economic losses. There is still a certain gap between the experimental data results and the real network security solutions. Therefore, it is very important to study anti- phishing solutions in a real-time environment. We divide the related work into two parts:

(1) deep learning-based methods for detecting phishing websites
(2) frameworks with prototype implementations.

III. DEEP LEARNING-BASED METHODS

In this part, we reviewed some state-of-the-art deep learning-based solutions for phishing websites detection. Buand Cho [11] proposed a deep auto encoder model to detect zero-day phishing attacks and obtained 97.34% accuracy. They extracted character-level features from URL strings and executed experiments on three different datasets collected from Phish Storm [2], ISCX-URL-2016 [12], and Phish Tank [13]. They used receiver-operating characteristic curve analysis and N-fold cross-validation to evaluate the experimental results. Comparing the root mean square error(RMSE) in the reconstruction phase between legitimate URLs and phishing URLs, they found the RMSE increased significantly for the phishing URL. Some ha *et al.* [14] introduced deep learning models for detecting phishing websites only using ten features extracted from HTML and a third-party service. They compared three deep learning models and calculated 18 features weights. The experimental results demonstrated that the Long Short-Term Memory (LSTM) model achieved the highest accuracy of 99.57%. However, they only use done published dataset with 3526 instances. The dataset is obviously too small for deep learning training. The high accuracy rate in the experimental results may be due to the uneven distribution and poor diversity of the test data.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ICRTTC '23
International Conference on Recent Trends and Technologies in Computing
Department of Computer Science and Engineering, Arasu Engineering College
Vol. 12, Special Issue 1, April 2023

IV. FRAMEWORKSANDSYSTEMS

When detecting whether a webpage is at risk of phishing attacks, the core service is a prediction service based on machine learning. The response time of predictive service is the most important indicator to measure the feasibility of this real-time system. Atimorathanna *et al.* [16] introduced an anti-phishing protection system, which consists of a web browser extension, an e-mail detection plug-in, filters, and a machine learning based phishing detecting server. The browser extension is used to extract the current URL, capture a screenshot, and store the user's visit history as a profile on the client-side. The server mainly uses the following processes to detect phishing links:(1) using the black list and white list of third-party services to filter new URLs; (2) using a machine learning model based on 13 features to predict whether the URL is a phishing link; (3) using computer vision technology to detect website logos and comparisons the similarity of screenshots of web pages.

Abiodun*etal.* [20] developed a website to verify a link is a phishing URL or not. The detector was implemented by JAVA programming language and a library named JSoup HTML Parser (JHP). This solution is mainly divided into three stages. The first is to use JSoup to parse the DOM structure of the website to be detected. The second is to analyze the number of link tag <a>from the DOM structure and analyse the attribute ''href'' value. The attribute value is classified as an empty link, external links and internal links. Third, the link calculator figured out an indicator, which has a value between 0 and 1. When the value exceeds 0.8, the URL to be verified is considered a phishing link. Since no machine learning model is introduced, there is no training process. In the experiment, the authors used 300 URLs to test the performance of the link calculator. The testing results showed they achieved 99.97% accuracy and a 0.03 false-negative rate. They will need to use a larger test data set to verify this solution in the future. From the analysis, it is a misjudgement to judge the phishing risk by analysing the characteristics of the link tag from the website source code alone, and it is easy for attackers to use this rule to circumvent these rules.

FRAMEWORKDESIGN:

Figure 1 depicts the architecture of the components of our proposed framework. There are four modules in terms of data collection tasks, machine learning (ML), cloud application, and web browser extension. The data collection module is an independent scheduled task application. The ML module is used for training modules. The web browser extension is a client-side product. The cloud application is built to deal with false alarms and phishing URLs reported by users from the web browser extension. The orange lines with arrows in the figure show the data interaction process. The core process of this framework is mainly divided into the following six steps: the first is to collect and integrate data from various data sources; these condition is to combine different datasets for machine learning model training, and store the trained model in a file system; the third is that the interface for predicting phishing risk calls the trained model to make predictions; the fourth is that the browser extension calls the prediction interface to perform real-time detection and display the detection results; the fifth is that users can submit real-time feedback when they disagree with the detection results, such as misjudgement, missed alarm; finally, the report submitted by the user is verified through manual review and automatic review strategy, and the verification result is synchronized to the dataset.

DATA COLLECTIONTASKS:

Data is the core of the field of machine learning. The quality and quantity of data significantly impact the performance of machine learning-based modules [22]. The data collection module is the foundation of this system. A data collecting task is divided into two parts, obtaining data from different data sources, then analysing and storing data. We collected data from different open sources shown in Table1. The Phish Storm [2] dataset contains 96,018 URLs: 48,009 legitimate URL sand 48,009 phishing URLs.

The ISCX-URL 2016[12] dataset contains 35378 legitimate URLs and 9965 phishing URLs. We loaded around 350,000 benign URLs from an open Kaggle project[23]. In addition, we initially collected 400,000 data and regularly grabbed new data from the Phish Tank platform[13]everyday. We analysed the basic structure of the URL and parsed out basic information such as protocol, domain, sub-domain, top level domain, and path [24]. Table 2 presents the major fields of a table named URL. We stored data in a relational database, as it is flexible and efficient for providing data services by reading based on SQL. These data services can combine multiple data sets. For example, select 20,000 phishing links from phish tank and 20,000 good links from Kaggle, and combine them into a balanced data set with 40,000instances.

MACHINE LEARNING:

The machine learning module is mainly responsible for model training and model testing. In this frame work, the data of the training model is updated regularly, and the training and testing processes of all models are automatically and regularly triggered. The system will record each run's parameters and data collection types and dump the model to the file storage system. It is flexible to add new models to the ML module. This research developed six machine learning models, namely Logistic Regression, Support vector machines(SVM), Random Forest, RNN, RNN-GRU, and RNN-Long short-term memory(LSTM).

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ICRTTC '23
International Conference on Recent Trends and Technologies in Computing
Department of Computer Science and Engineering, Arasu Engineering College
Vol. 12, Special Issue 1, April 2023

PARAMETER CONFIGURATION:
The parameter configuration process initializes the model parameters according to the configuration file. The configuration file includes a parameter grid corresponding to each model, and each parameter has a discrete number of values. In the model training process, one of the permutations and combinations of these parameter values will be selected for each training. When all the combinations are applied to the model and the training is completed, the optimal parameter combination can be obtained by comparing the accuracy of the model.

DATA LOADING:
The data set used for model training is obtained from the database through the data service. The data service supports the flexible selection of different data source combinations and data sets of varying data volumes. Each data instance contains a URL string and a label that signs the URL is a phishing link or legitimate link. The label values are normalized as 1 and 0.

FEATURE EXTRACTION:
We treat the URL string as a document containing semantics and apply the Natural language processing (NLP) technology to extract features. The feature extraction process converts a collection of text documents to a matrix of token counts, and each token stands for one word. In classical machine learning models, the tokenization process converts a URL string to a list of words. Therefore, the number of features equals the vocabulary size found by analysing the data. In deep learning models, the tokenization process parses a URL string to a list of characters(Character-level tokens).The characters in the URL come from the ASCII character set. We chose the most common 100 characters as the character set dictionary for this study. Figure 2 shows all the arranged characters and the corresponding index. The maximum length of a URL is 2083 characters [24]. Because of the calculation time of the deep learning model and the analysis of the statistical data of the existing data set, we set the maximum number of URL characters to 200. Therefore, each URL can be transformed into a 200*100 matrix. The position of the dictionary corresponding to each character is marked as 1, and the remaining values are 0.

MODELLING:
It is a solution to treat a URL as a document and use character separators to parse words as features. However, many words in URLs also lack semantics. Moreover, the analysis of word level results in an extensive dictionary will slow the calculation time. Therefore, we choose to analyse with character level and the characters of the entire URL as a sequence. The recurrent neural network (RNN) is a feedback neural network that stores temporary states. It's suitable for training sequence data[25]. Figure4 shows a regular RNN architecture that consists of an input layer, several hidden layers and an output layer. Compared to the feed forward artificial neural networks (ANN), RNNs have a unique architecture with a connection function between neurons in hidden layers. The figure shows that the current hidden state is related to the previous hidden state and the current input. The current hidden state's functional form can be represented as Eq. (1) and (2). The tanh is an on linear function, W represents the weights between the neurons, and b is the bias vector of the setting. The soft max calculates the output value as an activation function, as shown in Eq. (3), and the model prediction value is related to the current hidden state.

The scenario that detects the phishing link is a many-to-one task type, the input is character-level sequence data, and the output is a category. Figure 5 shows the structure of one hidden layer. Before model training, the structure of the model is fixed, and activation function is established, so the process of model training is the process of optimizing the weights parameters by calculating each error. First, randomly initialize the weights matrix, then calculate the difference between the actual value and the predicted value, then use the optimized algorithm to find the optimal solution to minimize the difference, and finally adjust each weight by calculating the step each time.

Depending on the architecture of RNN and activation functions used, the basic RNN architecture does not perform well for handling inputs for long sequences because of the vulnerability to gradient vanishing or exploding problems [26]. To address these, Hochreiter and Schmidhuber introduced a gradient-based model named long short-term memory (LSTM) in 1997 [27]. They invented a long short term memory unit instead of tanh function to compute hidden states. The LSTM unit consists of three gates and two memory cells. Cho *et al.* proposed a novel model with a hidden unit, which was motivated by LSTM in 2014 [28].Sincethehiddenunitcontainstwogatestocontrolandcalculatethehidden state, this model is also named Figure 6 demonstrates the architecture of the gate units. It can be said that long short-term memory network (LSTM) are two enhanced versions of RNN. Many studies and experimental data show that for sequence data training, the LSTM and XGBOOST architecture can achieve better performance than the basic RNN architecture[29]–[31].

OPTIMIZER AND LOSS FUNCTION:
In the model training process, it is also essential to choose a suitable optimizer and loss function. Among many optimization algorithms, we have selected Adam, which is a popular and effective optimization algorithm for deep learning [32]. Since the problem is a binary classification problem, we used the cross-entropy loss function [33], which is also called log loss function.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ICRTTC '23
International Conference on Recent Trends and Technologies in Computing
Department of Computer Science and Engineering, Arasu Engineering College
Vol. 12, Special Issue 1, April 2023

According to the scenario of the current problem, the output predicted value is a floating-point number between 0 and 1. Cross-entropy loss increases as the predicted probability diverge from the actual label. It is believed that it will converge quickly in the initial stage of training with the same learning rate. The loss value of each epoch is calculated by calculating the average loss value of all data points.

DUMP MODEL TO FILE SYSTEM:

After the model training is completed, the system will save the model to the specified file directory. Each file has a different version number, and the file name is related to the version. When the system is deployed to the cloud, it will use the Google file system to manage these files. The prediction service can load these models directly and perform real-time prediction for single or multiple URL links, and each call takes less than 200 ms.

WEB BROWSER EXTENSION:

We have developed a Chrome browser extension. Since users installed the extension in the Chrome browser, the extension will automatically detect whether the newly opened URL is at risk of phishing. If there is a risk, the user will be interactively prompted through a popup box, and the entrance feedback error detection will be provided. The extension will call the HTTP interface of the prediction service to obtain the detection result and save the detection result in Chrome's storage.

PROTOTYPE IMPLEMENTATION:

The prototype implementation of the entire frame work is divided into three independent applications. The browser extension is independently packaged and uploaded to the Chrome browser according to the extension development specifications of the Chrome browser and will be reviewed and released by the Chrome platform. Chrome browser plug-in development uses three web front-end development languages: HTML, JavaScript, and CSS. The data collection application is based on Python as the main development language, using scheduled tasks to manage the collection tasks of each data source. Among them, phish tank crawls information from web pages, using the most used package named Beautiful Soup.

Model training, prediction services and product official website are integrated into one application. This application also uses Python as the primary language and imports Flask as the web framework. Model training is managed by timed tasks as well. After the training is completed, the core performance indicators are written into the MySQL database in real-time, and the model is dumped into the file system. The prediction service is a RESTful API that provides clients with real-time POST requests to obtain detection results. The core function of the official website is to accept the suspected phishing link submitted by the user and determine the link risk by manual and automatic verification.

WEB FRAMEWORK:

We used Python as our core language, which is a modern high-level programming language in the field of data mining and machine learning. There are various frameworks and libraries for the Python language. In our system, data collection, data storage, model training, websites, and HTTP services are all supported by mature libraries and frameworks. In addition, the access and use of these packages are very simple and convenient. Considering the usage scenarios and read and write performance, we chose the MySQL relational database. First, the website has user management, report management, model version management and other functions, which require a relational database. In addition, the data set used for model training is acquired dynamically .It is very flexible to combine different data sources and data volumes to form a new dataset for model training. For example, we obtained 200,000 phishing URLs from Phish tank and 340,000 legitimate URLs from Kaggle. A balanced data set with 40,000 URLs can be flexibly combined, including 20,000 phishing URLs and 20,000 legal URLs. We imported the Flask as a web framework to provide HTTP service and maintain the official website. Flask is a light weight web framework and easy to extend [34]. For example, the flask-user package provides user authorization services.

EVALUATION RESULTS:

All the experiments were executed on a MacBook Pro 2020 running Quad-Core Intel Core i5 CPU @ 2GHz with mac OS BigSur11.5.2 operating system. The server has a 500 GB storage capacity. In addition, the test data ratio is0.2.We have seven data sets, six models, and each model has different parameters. Our experiment is carried out in the following steps to find the optimal model quickly. First, choose a model that may perform well from the theoretical analysis. Then, we compare the datasets with the best performance from the experimental results. Second, we use the dataset determined in the previous experiment to train different models and compare the results. Finally, we optimize the model hyper parameters for the model with the dataset. The primary method is to enumerate the optional discrete values of the parameters and perform cross-combination to compare the performance of all experimental results. Evaluate whether a machine learning model has high performance, standard statistical metrics with accuracy, recall, and precision[38].These indicators are obtained by simple mathematical calculations of four atomic statistical indicators in terms of the number of correctly identified positive instances (TP), the number of correctly identified negative data points (TN), the number of negative data points predicted by the model are positive (FP), the number of positive

instances labelled as negative (FN). In the article, we use the F1 score to represent the meaning of the recall and precision. In addition, in cyber security detection applications, false alarms can affect the user experience and trust, and leak alarms are likely to directly cause user losses. Therefore, we use accuracy, F1, false-positive rate, false-negative rate to measure the efficiency of models. Furthermore, Average precision (AP) is a widely used metric in evaluating the accuracy of deep learning models by computing the average precision value for recall value over 0 to 1; higher is better. Mean average precision (mAP) is the average of AP. Equation (10) shows the calculation logic. In this scene, the number of classes is two.

LSTM-XGBOOSTWITHDIFFERENTDATASETS:

In this experiment, we compared different datasets fed to the LSTM classifier. The number of epochs is 20, the batch size is 32, KPT stands for the data collected from Kaggle and Phish Tank, and each KPT dataset is a balanced dataset, which consists of the same number of phishing URLs and legitimate URLs. Table 3 shows the core performance indicators of the GRU model with 8 datasets. The ISCX dataset obtained the highest accuracy. However, the F1 score is lower than the other three KPT datasets, and the false-negative rate is high. In other words, more legitimate instances are predicted as phishing URLs during the test data process. This result is probably because there are not enough data points labelled as phishing. Furthermore, we combined false-positive rate and false-negative rate to measure efficiency. We can see that the LSTM-XGBOOST model with the KPT-12 dataset performs best. In KPT datasets, the false rate decreases linearly as the number of data increases shows the accuracy and F1 of each dataset. The KPT-12 dataset obtained 99.18% accuracy and 99.15% F1 score. To quantify how well the LSTM-XGBOOST model performs every class, we calculate the mean of average precision (mAP) of a set of classes. The mean average precision (mAP) in LSTM-XGBOOST modelswithKPT-12 dataset is 0.986. Assessing the efficiency of a machine learning model is in complete, depending on accuracy. In experiments, it is customary to get high accuracy in one dataset and not perform well in another. It is also likely that models with high accuracy will not predict new data accurately in a real-time environment. These situations may be due to the fact that over fitting has already occurred [39]. Over-fitting is a concept in data mining that analyses whether a trained model can efficiently predict unknown new data [40]. In machine learning based classification models, it is common to compare errors in the training process with errors in the validation process to see if there is over-fitting, along with epoch. Figure 9 presents the training loss and validation loss along with epochs in the RNN-GRU model. One of the strategies to avoid over-fitting is early-stopping [40],[41]. As shown in figure12, the epoch equals 6 is the demarcation point between under-fitting and over-fitting.

V. CONCLUSION AND FUTURE WORK

Many machines learning-based solutions have been proposed in recent years to deal with phishing attacks, but results have not been verified in live browsing environments, and there is a lack of analysis and research of products for phishing detection. In this paper, we proposed a framework for phishing detection in a real-time browsing environment. The novel features of the frame work are: We utilized closed-loop data to drive better performance of machine learning models. A dataset is fundamental to model training, and high-quality data can improve the performance of a model.

The feedback data from users are high-quality data with advancement, accuracy, and sensitivity. The system is running in a real-time environment without delays. The prediction results are displayed when the webpage is opened. Experimental data can be tracked. The model training process is an automated task, and each execution result is stored in a real-time database. We have developed a browser extension as a client product that every ordinary netizen can use. The implementation of predictive services is extendable, and individual detection services can be combined. For example, you can introduce a blacklist filtering service, computer vision service. The feature extraction process in the deep learning model is independent of third-party services.

In the future, we will deploy the whole system to a cloud platform. Configure machines with NVIDIA GPUs for model training and increase efficiency with parallel computing power. Afterward, users can download the extension through the Chrome Web Store. In addition, we plant implement our framework as a plug-in for other browsers.

REFERENCES

[1] L.Tangand Q.H.Mahmoud,''A survey of machine learning-based solutions for phishing website detection,'' *Mach. Learn. Knowl. Extraction*, vol. 3, no. 3, pp. 672–694,Aug.2021, doi: 10.3390/make3030034.
[2] S.Marchal,J.Francois,R.State,andT.Engel, ''Phish Storm: Detecting phishing with streaming analytics,'' *IEEE Trans.Netw. Service Manage.*, vol. 11, no. 4, pp. 458–471, Dec.2014.
[3] R.M.Mohammad,F.Thabtah,andL.McCluskey,''Predicting phishing websites based on self-structuring neural network,'' *Neural Comput. Appl.*, vol. 25, no. 2, pp.443–458,Nov. 2013,doi:10.1007/ s00521-013-1490-z.
[4] M.A.El-Rashidy,''A smart model for web phishing detection based on new proposed feature selection technique,'' *Menoufia J. Electron. Eng. Res.*, vol. 30, no. 1,pp.97–104, Jan. 2021,doi:10.21608/ mjeer.2021.146286.
[5] B.B.Gupta,K.Yadav,I.Razzak,K.Psannis,A.Castiglione,andX.Chang,''A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment,'' *Comput. Commun.*, vol. 175, pp. 47–57, Jul.2021,doi: 10.1016/j.comcom.2021.04.023.

[6] E. Gandotra and D.Gupta, ‘‘Improving spoofed website detection using machine learning,’’*Cybern.Syst.*,vol.52,no.2,pp.169–190,Oct.2020,doi:10.1080/01969722.2020.1826659.
[7] W. Wang, F. Zhang, X. Luo, and S. Zhang, ‘‘PDRCNN: Precise phishing detection with recurrent convolutional neural networks,’’ *Secur. Commun. Netw.*, vol. 2019, pp. 1–15,Oct. 2019, doi:10.1155/2019/2595794.
[8] M. Sabahno and F. Safara, ‘‘ISHO: Improved spotted hyena optimization algorithm for phishing website detection,’’*Multimedia Tools Appl.*, Mar. 2021, doi: 10.1007/s11042-021-10678-6.
[9] S.-J.BuandS.-B.Cho,‘‘ Deep character-level anomaly detection based on a convolutional auto encoder for zero-day phishing URL detection,’’ *Electronics*, vol. 10, no. 12, p.1492,Jun. 2021,doi:10.3390/electronics10121492.
[10] M. Somesha, A. R. Pais, R. S. Rao, and V. S. Rathour,‘‘Efficient deep learning techniques for the detection of phishing websites,’’ *Sadhan¯a¯*, vol. 45, no. 1, pp. Jun.2020, doi: 10.1007/s12046-020-01392-4.