Ransomware Detection using Machine and Deep Learning Approaches

Ramadhan A. M. Alsaidi[1], Wael M.S. Yafooz[2], Hashem Alolofi[3], Ghilan Al-Madhagy Taufiq-Hail[4], Abdel-Hamid M. Emara[5], Ahmed Abdel-Wahab[6]

Department of Mathematics, College of Science and Arts in Gurayat, Jouf University Gurayat, Saudi Arabia[1]
Department of Computer Science, College of Computer Science and Engineering, Taibah University, Medina 42353, Saudi Arabia[2]
Bremen University, Bremen, Germany[3]
College of Business (COB), University of Buraimi (UOB), Al Buraimi Governorate, Oman[4]
Department of Computer Science, College of Computer, Science and Engineering, Taibah University, Medina 42353, Saudi Arabia, Department of Computers and Systems Engineering, Faculty of Engineering, Al-Azhar University, Cairo 11884, Egypt[5]
Faculty of Computer Studies, Arab Open University, Riyadh, Saudi Arabia, Department of Computers and Systems Engineering, Faculty of Engineering, Al-Azhar University, Cairo 11884, Egypt[6]

***Abstract*—Due to the advancement and easy accessibility to computer and internet technology, network security has become vulnerable to hacker threats. Ransomware is a frequently used malware in cyber-attacks to trick the victim users to expose sensitive and private information to the attackers. Consequently, victims may not be able to access their data any longer until they pay a ransom for stolen files or data. Different methods have been introduced to overcome these issues. It is evident through an extensive literature review that some lexical features are not always sufficient to detect categories of malicious URLs. This paper proposes a model to detect Ransomware using machine and deep learning approaches. This model was introduced as a novel feature for classification using the idea that starts with "https://www." This feature was not considered in the earlier papers on malicious URLs identification. In addition, this paper introduced a novel dataset that consists of 405,836 records. Two main experiments were carried out utilizing malicious URL features to defend Ransomware using the proposed dataset. Moreover, to enhance and optimize the experimental accuracy, various hyper-parameters were tested on the same dataset to define the optimal factors of every method. According to the comparative and experimental results of the applied classification techniques, the proposed model achieved the best performance at 99.8% accuracy rate for detecting malicious URLs using machine and deep learning.**

Keywords*—Machine learning; ransomware; URL classification; malicious URLs; deep learning

I. Introduction

Over the past few years, the growth of ransomware has become an uncontrolled cyber problem and highly profitable criminal business. Ransomware attacks are primarily performed using malicious Uniform Resource Locators (URLs) [1]. On May 7, 2021, Colonial Pipeline [2], an American oil pipeline system that originates in Houston, Texas, suffered a ransomware cyberattack that hit computerized facilities managing the pipeline.

In response, Colonial Pipeline Company stopped all of the pipeline's operations to contain the attack. Colonial Pipeline paid the demanded ransom (75 bitcoin or $4.4 million) within several hours after the attack. The attackers then sent Colonial Pipeline a software application to restore their network, although it was slow. This example illustrates how problematic Ransomware can be. Therefore, it becomes a necessity to know more about Ransomware [3].

There are essentially two types of ransomware: locker and crypto-ransomware. Locker ransomware works by blocking the victim from arriving at their files by denying access to computing resources, such as locking the desktop or preventing the victim from logging in and, demanding a ransom to regain access to the system. Crypto-Ransomware encrypts all data on the target system until the victim pays the ransom via the Bitcoin currency and obtains the decryption key from the hacker. Some kinds of crypto-ransomware can progressively erase files or release them to the public if the victim declines to pay the ransom on time. Recent ransomware families are essentially based on this type. It can have disastrous effects, primarily on corporate and government agencies, if they do not have a backup to restore the system to the state before the attack. Following the installs of the crypto-ransomware installs on the target system, it can noiselessly search for essential files based on their extensions and starts encrypting them. Then the ransomware will search for files on the local hard disk, the external hard disk, and in the network shares. Many crypto-ransomware users make sure that the files cannot be restored from the backup by totally deleting the backup. Usually, there are three main ways that ransomware can attack a system. They can get it infected, then takes over the files.

The first common way is through browsing the web. The malicious websites usually install the exploit kit to take over the machine and install additional malicious software such as ransomware. The other even more common way is through emails. Emails become a very demanding business tool. Email, more specifically, has two different vectors within itself. It may contain a malicious web link. That is sent within the email message that people click assuming that they are directed to a safe web page location. The second common way

in which an email can affect us is with malicious attachments sent directly in the email body. Once it opens, it can take control over the entire system. The third common way is through free software or games. Most of the free software found on the Internet is malicious software developed by hackers, disguised as legitimate software to access systems. Based on the said, it is necessary to notify the user that some URLs direct users to a malicious website before accessing it. Though numerous cybersecurity techniques defences are developed against Ransomware URLs, the nature of the security still requires further research to improve the entire system. As a remedial response to Ransomware threats, the present paper proposes a model to detect and track malicious URLs using machine learning classifiers and deep learning approaches. To enhance and secure the efficiency of our model, a novel dataset called Ransomware Detection Dataset (RDD) has been introduced. RDD is available for the public on the git hub website [4].

The proposed model achieved a high performance in terms of accuracy using both MCLs and DL, securing 99% and 99% respectively. In addition, through the discussion and analysis of numerous identity algorithms, malicious URLs detection systems were used to identify different forms of known phishing and to examine the appearance of dangerous URLs in websites.

The contribution of this paper can be briefed as follows:

- Proposed a model to detect and track malicious URLs using machine and deep learning approaches.
- Proposed a new feature used in identifying malicious and benign classification using the idea starting with "https://www.
- Introduced a novel dataset containing 405,836 rows of a URL to classify webpages as Malicious or Benign called Ransomware Detection Dataset (RDD) [4]
- Compared performance of the proposed model with that of MLCs and DL approaches.
- Proposed detection algorithms that enhance accuracy in classifying malicious and benign website applications via optimizing URL features and selecting optimal factors for every method.
- Examine three loss functions (Binary Crossentropy,Hinge, and Mean Squared Error respectively) on the RDD using three optimizers (Adam,SGD, and ADAgard).

The rest of this paper is structured as follows: Section II has to do with a brief review of related works; Section III describes the research approach, feature extraction and classifications algorithms; methods and materials will be explained in Section IV; experiments, results and discussion will be presented in Section V; Section VI concludes the paper and provides suggestions for future work.

II. Related Work

To improve accuracy and faster detection, numerous methods have been introduced for malicious URLs detection. Recently, research has shown that many detection methods can resist the increasingly diverse hackering threats. Popular techniques used for malicious URL detection are machine learning/deep learning-based detection, blacklist-based detection and, heuristic rules-based detection. Blacklist based approach utilizes a massive list of malicious URLs that can be created from numerous sources manually or programmatically. Many commercial products build blacklists utilizing user proprietary and their feedbacks mechanisms to detect malicious URLs [5] and [6], like WOT Web of Trust [7], McAfee's SiteAdvisor [8], Cisco IronPort Web Reputation [9], and Trend Micro Web Reputation Query Online System [10]. On the other hand, diverse websites supply blacklist applications, such as: PhishTank [11], jwSpamSpy [12] and DNS-BH [13]. Although blacklist technique gives faster and low false positive detection, it has some drawbacks. The newly generated malicious URLs are not detected, so the database of blacklisted URL requires timely and frequent updates. As the result, researchers using the heuristic rule-based detection method to overcome this issue, in which generalized rules are used for malicious URL detection. These rules are based on features extracted from the datasets [14]. The limitations of this method are the mapping of weightage to the rules and, the threshold value may be modified based on datasets, it becomes unclear how to fix threshold value for each rule [15].

Later, the machine learning approach has been used in several methods to classify malicious URLs. It operates on two stages: first, extraction of the dataset features and training the model; then, training is used for testing. The best advantage of this technique is that the generated model can automatically assign the weightage of selected features and detect the malicious URL, including newly generated malicious URLs. A survey on malicious URL recognition via machine learning was presented by Doyen et al. in [16]. It scans the cons and pros of various methods used in malicious URLs Authors Suppressed Due to Excessive Length identification literature. Practically, the features utilized in machine learning approach are classified into four broad categories: host-based features, popularity features, content-based features and lexical features. Author in [17] presented a multiclass classification method to identify the malicious URL. It considers 18 hos-based features, 34 content related features and 63 lexical features. Recently, Djaballah et al. [18] presented a new work to recognize the malicious URL in social network such as twitter. In that work, three machine-learning techniques were used namely Random forest, SVM and Logistic Regression for the experiments, giving 5.51%, 93.43% and 90.28% accuracy rate respectively. In [19], the researchers suggested a technique for malicious website detection which utilizes various feature selection methods and diverse machine learning techniques with PCA such as RF, NN, bagging, KNN, NB, and SVM were used for experiments. In [20], the authors presented a method of choosing the optimal features using Chi-Square and ANOVA F-value to detect phish URL. The light weighted technique that uses only a fewer number of lexical features for malicious URL detection was introduced in [14].

III. Proposed Approach

This paper aims to explore how a URL is identified as malicious or benign by using only its associated URLs lexical features quickly and accurately. This work works on build a URL classifier to predict whether a URL is malicious or benign with a very high degree of recall and precision. When the user sends a DNS (Domain Name System) request, it will be sent first to our URL classifier. If the URL is Benign, then the DNS request will be sent to the DNS server. And in case the URL is malicious, then DNS requests will be dropped and not forwarded to the DNS server.

IV. Methods and Materials

This section presents the methods and phases of the proposed model. There are four phases in the presented detection architecture. They are data collection, feature extraction, classification, and model evaluation as shown in Fig. 1.

A. Data Collection

This is the input phase on the mode. Processing and preparing datasets are critical stages. However, getting sizeable balanced data is one of the potential challenges. The efficiency and performance of the system essentially depend on them. The importance of this issue appears when the model deals with imbalanced data that will cause a bias towards predicting the larger class, and on the other hand, may ignore the smaller class altogether. Therefore, one of the challenges that this work is handling a sizeable balanced dataset. Consequently, the dataset was collected from six different Kaggle resources [21, 22, 23, 24, 25] to create an extensive balanced dataset. This effort acquired a total of 405,836 URLs, 202918 benign and 202918 malicious. At the end of this phase, the dataset has been built.

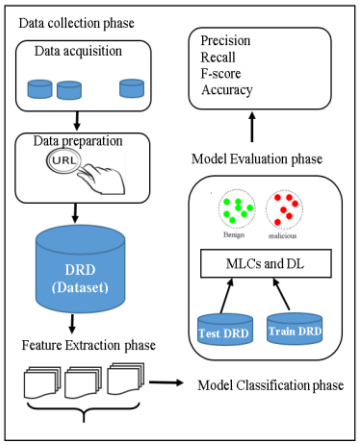

Fig. 1. Model Architecture.

B. Features Extraction

This is the second phase of the model. It receives input from the first phase. The reliable and discriminative feature is significant for achieving high-performance machine learning classification. The criteria of attributes are essential to present the crucial characteristics of the malicious URL. It should be noted that in the Feature extraction phase, there are three main steps: the length of the feature, count feature, and binary feature. These steps are performed using the work developed in the open-source code, which Siddharth Kumar provides in [23]. The output of this phase is the features extracted from the URLs. These features are used in the next phase which is classification.

What is a URL? To understand attackers' strategy, firstly, the reader must know the basic components of a URL. The basic structure of a URL is depicted in a URL is typically made up of six or seven components. First is the Protocol, which is used to access resources on the Internet. It could be Hypertext Transfer Protocol Secure (HTTPS) or the less secured HTTP. The other components are subdomain, domain name, top-level domain, path, query, and parameters, as shown in Fig. 2.

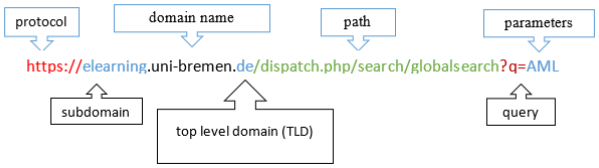

Fig. 2. URLs Components.

Features Score Initially, this work specified 17 most common lexical features. Since all extracted features are not always suitable for classification, the quality and effectiveness of each lexical feature was examined by testing and training the classifier with each feature group separately. The results are illustrated in Table I.

TABLE I. Features Score

Feature	Score	Feature	Score
Start with https://www	0.9978527	count=	0.546040
count-https	0.980527	count?	0.545931
count-www	0.924533	path length	0.516274
ld length	0.723060	use of ip	0.513758
hostname length	0.681407	count-digits	0.510565
count-http	0.666719	count@	0.510015
count.	0.605214	count%	0.498175
count dir	0.552626	fd length	0.484810
count-	0.549475	count-letters	0.473917

A compelling feature selection will enhance the classifier ability of the model with low computational complexity that reduces the storage consumption and execution time.

Ranking analysis help to find the optimal feature. The effect of prominent features on recognizing accuracy will be addressed in the following:

- Counting HTTPS The first and most crucial feature found was the count-https. The count of HTTPS was calculated in all URLs in the dataset. It has been observed that all benign URLs (202918) use the HTTPS protocols. On the other side, most of the malicious URLs (195014) do not use HTTPS protocols, making this feature one of the best features for training our classifier.
- Counting www The second most essential feature is count-www in which the frequency of www was

counted in all URLs in the dataset. It has been observed that almost all benign URLs (202739) use www whereas most of the Malicious URLs (172290) do not have www.

- Start with https://www Counting https is an essential feature since most malicious URLs do not use HTTPS. However, the problem with this feature in previous work is that it just counts how much time it acquires. It does not take into account the location of the HTTPS. While testing, it was noticed that the result got many false positives because many malicious URLs used HTTPS in the middle of the URL. To solve this problem, this research introduced a new feature called Start with https://www. It is a binary feature that returns one if the URL starts with https://www. And returns zero otherwise. This feature will enhance the classifier ability of the model (i.e., reach an accuracy of 99.78%) with low computational complexity that reduces the storage consumption and execution time. This feature is newly selected and extracted in this paper.

To represent URLs detection issues mathematically. Let us define a function space [26, 27] as:

$F : u \rightarrow \{0, 1\}$

Where u represent the URLs. Note that the function is assumed to take values in {0, 1}. Meanwhile, g(u) represents the values of every feature.

$$(F \circ g)(u) = F\big(g(u)\big) = F([f1, f2, \dots, f19])$$

$$= \{1\ malicious\ \ 0\ Benign\}$$

where g(u) has three operations $\{counting\ computelength\ Boolean\}$ and [f1, f2, ..., f19] represents the value of every features.

C. Classification Process

In the third phase, the classification process is carried out by using the most common MLCs and DL approaches used scholarly [28, 29]. In MCLs, the most common classifiers used are the Random Forest, Gradient Boosting Classifier, Neural Net,poly SVM, Decision Tree, AdaBoost, Nearest Neighbors, Stochastic Gradient Descent (SGD), Naive Bayes, and QDA. The general overview of the machine learning classification process is shown in Fig. 3. In deep learning models architecture as shown in Fig. 4, two deep neural networks ANN and LSTM mode are applied. In both experiments, the dataset is divided 30% for training and 70% for testing.

The model performance has been evaluated based on the most common methods using a confusion matrix. For all experiments, the unified performance indicators such as accuracy, precision, recall, and F1 were used to evaluate the performance of the model as represented in the following mathematical formula.

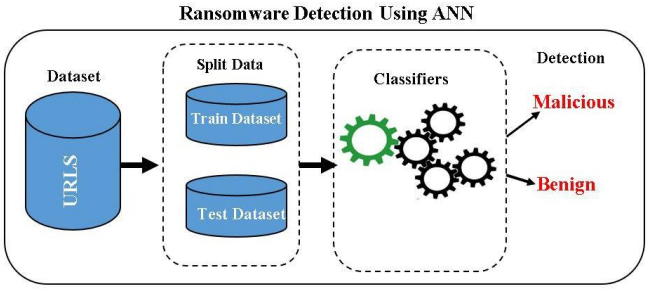

Fig. 3. Overview of Machine Learning Classification Process.

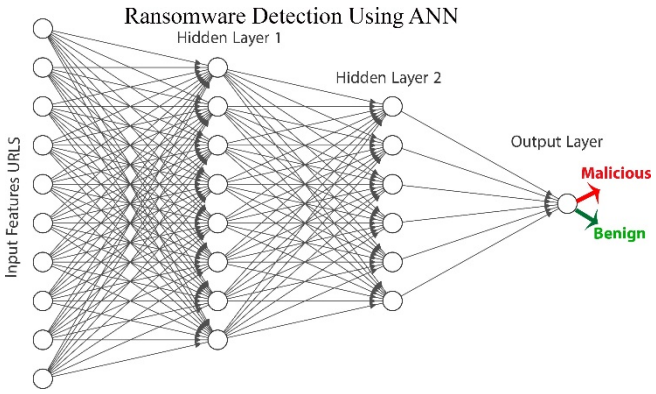

Fig. 4. Deep Neural Networks Architecture.

D. Model Evaluation

The model evaluation has been used in the most popular accuracy measure called confusion matrix to calculate accuracy, precision, recall and, the F1 Score that use in [30].

- Accuracy: measures how much of the data are labeled correctly

$$Accuracy = \frac{T\,P\ + T\,N}{T\,P\ + T\,N + F\,P\ + F\,N}$$

- Precision (specificity): it estimates how many identified targets are indeed relevant (real targets)

$$Specificity = \frac{T\,P}{T\,P\ + F\,P} = \frac{true\ positive}{no.of\ predicted\ positive}$$

- Recall (Sensitivity): it measures how good it is at identifying the positive cases.

$$Sensitivity = \frac{T\,P}{T\,P\ + F\,N} = \frac{true\ positive}{no.of\ actual\ positive}$$

- F1 Score: F1 is the function of Precision and Recall.

$$F1 = 2\,x\,\frac{Precision\ x\ Recall}{Precision + \ Recall}$$

V. Results and Discussion

This section discusses the experimental results of the proposed model which were used to detect malicious URLs. These experiments are categorized according to their processing architecture; the machine learning experiments and deep learning approach experiments.

A. Machine Learning Experiments

This section presents and demonstrates the results of the experiments of most popular classifiers used in our model [28, 29]. The experiments evaluated in this section used a confusion matrix to show the detection performance of the models. After creating the large balance dataset and extracting features, the implementation of the model is completed in two stages. In the first stage, all the features were used to train the models. In the second stage, we tested the adoption of the presented model with new lexical feature, namely, Start with https://www. This feature enhanced the classifier ability of the model (i.e., reaching an accuracy of 99.78%) with low computational complexity that reduced storage consumption and execution time. For all experiments, 75% of the dataset was utilized for training and 25% for testing. It should be noted that for Preprocessing, Classification, Dimensional reduction, Model selection, and evaluation Scikit-learn (sklearn) library has been used. To find the best parameter values, the GridSearchCV, provided by sklearn, has been used. The results of the experiments are shown in Tables I and II.

TABLE II. EVALUATION METRICS

Sr. No.	Classification Algorithm	Accuracy	precision	recall	f1-score
1	Random Forest	0.998115	0.9999	0.996	0.9981
2	GradientBoosting Classifier	0.99809	0.9997	0.997	0.9981
3	Neural Net	0.997967	0.9996	0.996	0.998
4	poly SVM	0.997893	0.9997	0.996	0.9979
5	Decision Tree	0.997893	0.9996	0.996	0.9979
6	AdaBoost	0.997893	0.9998	0.996	0.9979
7	Nearest Neighbors	0.997856	0.9994	0.996	0.9979
8	Stochastic Gradient Descent (SGD)	0.997647	0.9998	0.996	0.9976
9	Naive Bayes	0.995232	0.9953	0.995	0.9952
10	QDA	0.935577	0.9944	0.876	0.9316

B. Deep Learning Experiment

This section presents the results of the deep neural network experiment built on the basis of a Multi-layer Perceptron. Two Deep learning models were applied. They are an artificial neural network (ANN) and long short-term memory (LSTM). All the experiments were executed using Keras by Jupyter platform. In the ANN experiment, the model consists of five layers of one input layer, three hidden layers, and one output layer. The input layer consists of seventeen input dimensions. The features and activation function is "Relu" and the output are 256 neurons. The three hidden layers consist of 256, 128, 64 neurons, and the activation function is "Relu" with a drop out of 30%, while the output layer consists of a single layer with "sigmoid" as the activation function. The loss function used in the testing stage was binary crossentropy while three optimizers were used. They are: Adam, ADAgard, and SGD in which the learning rate was recorded at 0.01, 0.001 and 0.0001. Both training and testing for the models were applied, and the dataset was divided into 70% for training, and 30% for testing utilizing Adam optimizer with the three categories of the learning rate. In training and validation, the model achieved a high-performance rate of 99.16% in terms of accuracy while for validation it achieved 98.10%. Twenty epochs were included, and the batch size was 64. The loss function noticeably decreased. The model accuracy of ANN for learning rates of 0.01, 0.001, and 0.0001 are shown in Fig. 5 to 7 for training and validation.

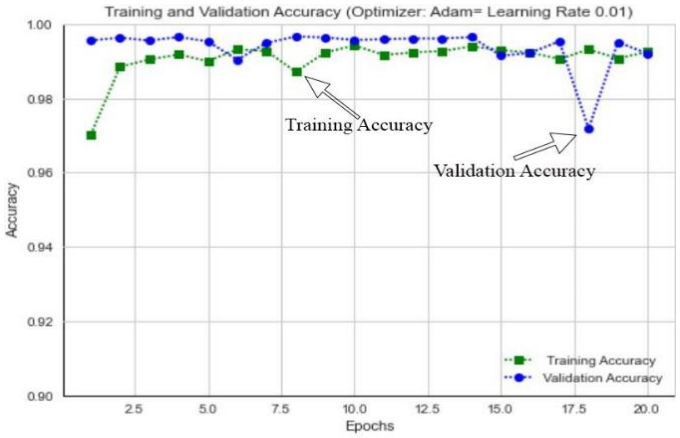

Fig. 5. Model Accuracy for Validation and Testing using Adam Optimizer (LR 0.01).

While loss function for testing and validation is shown in Fig. 8 to 10. The best accuracy rate has been recorded using a learning transfer of 0.0001.

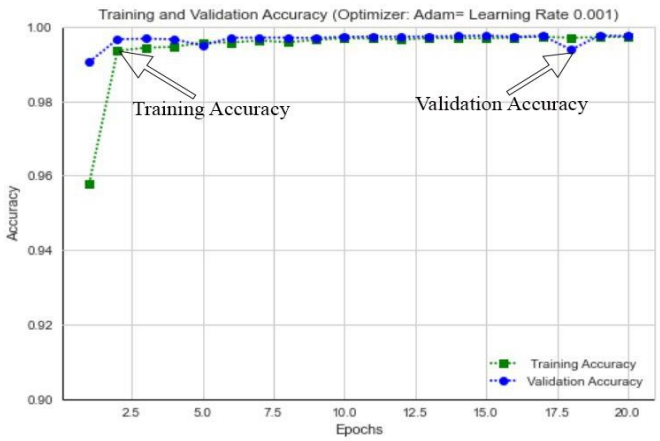

Fig. 6. Model Accuracy for Validation and Testing using Adam Optimizer (LR: 0.001).

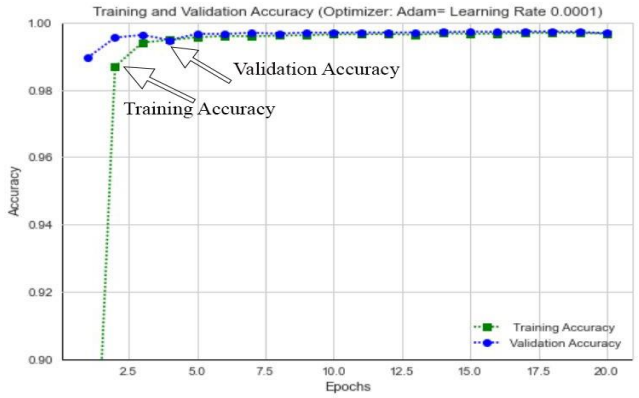

Fig. 7. Model Accuracy for Validation and Testing using Adam Optimizer (LR: 0.0001).

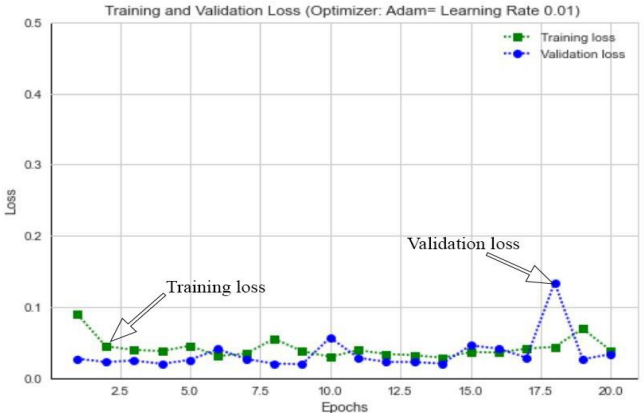

Fig. 8. Loss of Validation and Testing using Adam (LR0.01).

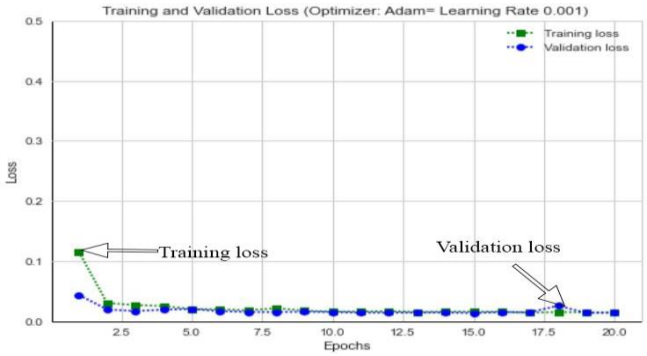

Fig. 9. Loss of Validation and Testing using Adam (LR: 0.001).

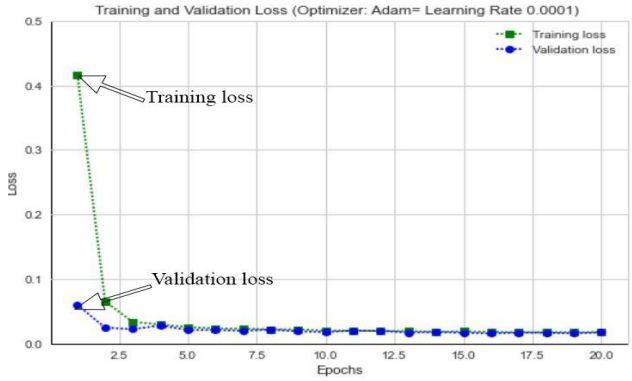

Fig. 10. Loss of Validation and Testing using Adam (LR: 0.0001).

Using the same experiment setting and same model hyperparameter "SGD" and "ADAgrad" optimizers were applied. The model performance was calculated as shown in Fig. 11 and Fig. 12 while the loss for the validation and accuracy is presented in Fig. 13 and Fig. 14. Additionally, three loss functions were used. The results as shown in Table III, IV, and V, Binary Crossentropy, Hinge, and Mean Squared Error respectively.

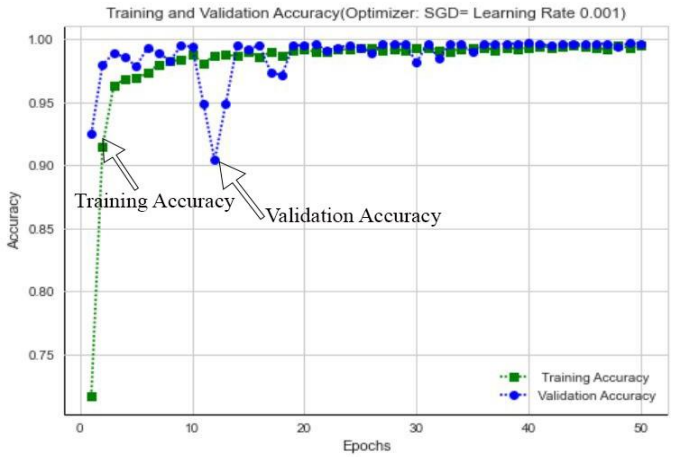

Fig. 11. Model Accuracy for Validation and Testing using SGD Optimizer.

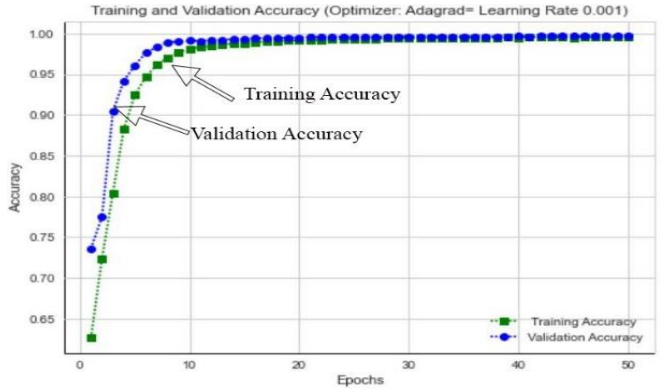

Fig. 12. Model Accuracy for Validation and Testing using ADAgard Optimizer.

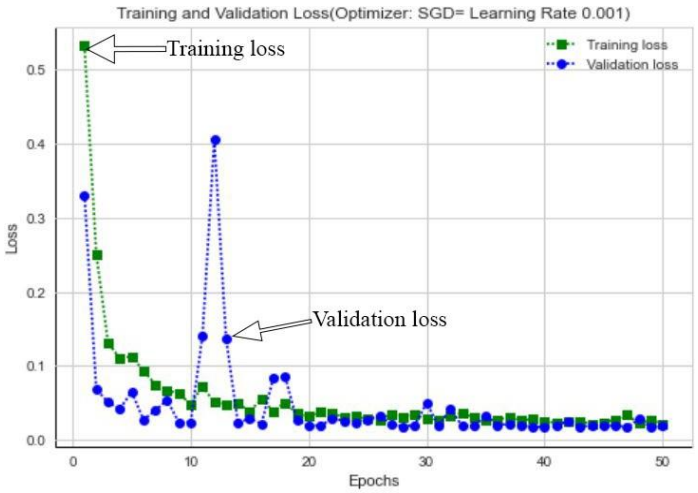

Fig. 13. Loss of Validation and Testing using SGD Optimizer.

TABLE III. ADAM OPTIMIZER WITH THREE LOSS FUNCTIONS

Loss function	TR	Loss	Acc.TR	Acc.VAL	Acc.TE
Binary Crossentropy	0.01	0.0203	0.9961	0.9965	0.997
	0.001	0.0161	0.9969	0.9973	0.997
	0.0001	0.0175	0.9967	0.9974	0.9967
Hinge	0.01	0.9985	0.5008	0.4997	0.499
	0.001	0.5117	0.9875	0.9933	0.993
	0.0001	0.5035	0.9959	0.9968	0.9968
Mean Squared Error	0.01	0.1887	0.8108	0.8858	0.885
	0.001	0.0028	0.9970	0.9969	0.996
	0.0001	0.0034	0.9964	0.9969	0.9969

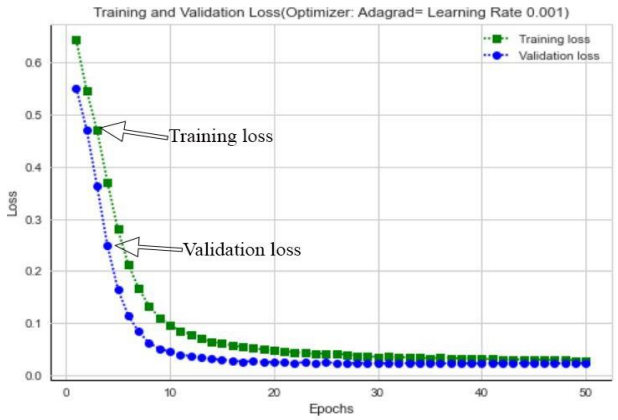

Fig. 14. Loss of Validation and Testing using ADAgard Optimizer.

In the LSTM experiment, the model consists of three layers. They are the input layer, LSTM layers, and output layer. In the input layer, the model receives a training dataset of features as an input dimension with 'Relu' activation function including 128 neurons.

In the second and third layers of LSTM, the number of neurons is 64, 32 respectively. The activation function is "Relu" for both layers. The dataset was divided into 70% for training and 30% for testing. In the training, the model reached a performance rate of 99.76% in terms of accuracy with twenty epochs using 'Adam' optimizer with a learning rate of 0.0001 and the validation reached to 99.66% is shown in Fig. 15. The loss function of validation and testing is "binary-cross entropy" which decreased with each training epoch is shown in Fig. 16.

TABLE IV. SGD OPTIMIZER WITH THREE LOSS FUNCTIONS

Loss function	TR	Loss	Acc.TR	Acc.VAL	Acc.TE
Binary Crossentropy	0.01	0.1463	0.9584	0.9592	0.959
	0.001	0.1543	0.9585	0.9757	0.975
	0.0001	0.5725	0.6909	0.7331	0.733
Hinge	0.01	0.5487	0.9514	0.9780	0.977
	0.001	0.7697	0.7383	0.7495	0.749
	0.0001	0.9257	0.5611	0.6394	0.639
Mean Squared Error	0.01	0.0251	0.9718	0.7260	0.725
	0.001	0.0984	0.9096	0.9518	0.951
	0.0001	0.2272	0.6247	0.7110	0.710

TABLE V. ADAGARD OPTIMIZER WITH THREE LOSS FUNCTIONS

Loss function	TR	Loss	Acc.TR	Acc.AVL	Acc.TE
Binary Crossentropy	0.01	0.02	0.9968	0.9971	0.997
	0.001	0.091	0.9824	0.9929	0.992
	0.0001	0.707	0.5557	0.6493	0.649
Hinge	0.01	0.504	0.996	0.9957	0.995
	0.001	0.642	0.8932	0.922	0.921
	0.0001	0.922	0.6003	0.689	0.689
Mean Squared Error	0.01	0.004	0.996	0.9967	0.996
	0.001	0.068	0.9389	0.9771	0.977
	0.0001	0.227	0.6314	0.7093	0.709

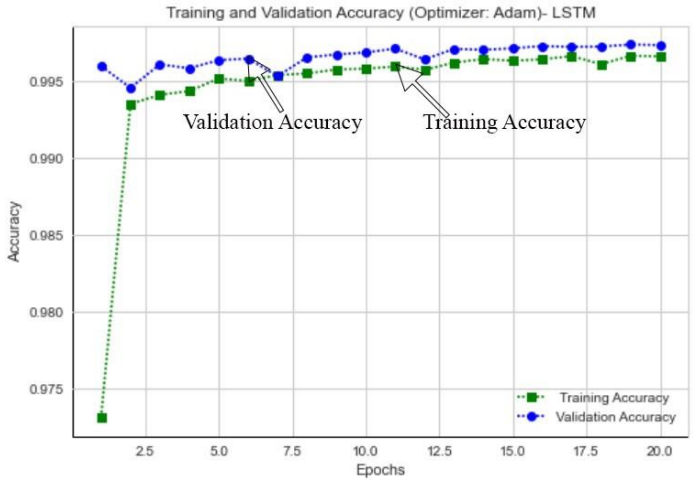

Fig. 15. Model Accuracy for Validation and Testing using LSTM.

Fig. 16. Loss of Validation and Testing using LSTM.

VI. CONCLUSION

The study explored the possibility of distinguishing benign and malicious URLs by using machine learning classification algorithms. A new feature was proposed for classifying and identifying malicious and benign using the idea of start with https://www. Experimental results using this feature were found significant. To evaluate the proposed feature, a sizeable balanced dataset has been created with a total of 405,836 URLs, 202918 benign, and 202918 malicious. Machine learning classifiers and deep learning approaches were applied, particularly ANN and LSTM. The proposed approach

gave a high classification accuracy rate of 99.81%, and a very high degree of classification precision at 99.99%. In the future, more focused research on modeling URL detection can be made to cope with the most sophisticated cyber-attack techniques.

ACKNOWLEDGMENT

Authors would like to thank the Arab Open University for supporting this research paper.

REFERENCES

[1] L. W. Al-Yaseen, "Improving intrusion detection system by developing feature selection model based on firefly algorithm and support vector machine." IAENG International Journal of Computer Science, vol. 46, no. 4, pp. 534-540, 2019.

[2] M. Sigalos, "Colonial Pipeline cyberattack is no cause for panic – here's why," CNBC, May 14, 2021. https://www.cnbc.com/2021/05/14/colonial-pipeline-hack-doesnt-mean-more-ransomware-attacks-critical-infrastructure.htm

[3] S. Sachdeva, R. Jolivot and W. Choensawat, "Android malware classification based on mobile security framework," IAENG International Journal of Computer Science, vol. 45, no. 4, pp. 514–522, 2018.

[4] R. Alsaidi, "Ransomware detection dataset (RDD) dataset," Ransomware detection dataset (RDD) dataset. [Online]. Available: kaggle, https://www.kaggle.com/ramdhanamalsaidi/a-novel-dataset-containing-405836-url, [Accessed November 28, 2021].

[5] H. Choi, B. B. Zhu, and H. Lee, "Detecting malicious web links and identifying their attack types," in 2nd USENIX Conference on Web Application Development, 2011, vol. 11.

[6] Y. Sun Y. Chen Y. P and L. Wu, "Android malware family classification based on deep learning of code images," IAENG International Journal of Computer Science, vol. 46, no. 4, pp. 524–532, 2019.

[7] "Web of Trust community-based safe surfing tool," Website Safety Check and Phishing Protection Web of Trust, 2021. [Online]. Available: https://www.mywot.com.

[8] "Service for reporting the safety of web sites," McAfee SiteAdvisor, 2006. [Online]. Available: http://www.siteadvisor.com.

[9] "IronPort Web Reputation: Protect and Defend Against URL-Based Threats," Ironportstore.com. [Online]. Available: https://www.ironportstore.com/search.php?q=Protect+and+defend+against+URL-based+threat. [Accessed: 25-Nov-2022].

[10] "Web reputation query - online system," TREND MICRO, 2021. [Online]. Available: http://reclassify.wrs.trendmicro.com.

[11] "Free community site for anti-phishing service," PHISHTANK, 2021. [Online]. Available: http://www.phishtank.com.

[12] J. Wein, "jwSpamSpy - E-mail spam filter for Microsoft Windows TM," Jwspamspy.net. [Online]. Available: http://www.jwspamspy.net. [Accessed: 25-Nov-2022].

[13] "Malware Prevention through domain blocking," yumpu.com. [Online]. Available: https://www.yumpu.com/en/document/read/44684944/malware- prevention-through-domain-blocking. [Accessed: 25-Nov-2022].

[14] A. Saleem Raja, R. Vinodini, and A. Kavitha, "Lexical features based malicious URL detection using machine learning techniques," Materials Today: Proceedings, Apr. 2021.

[15] H. Tupsamudre A. K. Singh and S. Lodha, "Everything is in the name–a url based approach for phishing detection," in International symposium on cyber security cryptography and machine learning, 2019, pp. 231–248.

[16] D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL Detection using Machine Learning: A Survey," arXiv [cs.LG], 2017.

[17] D. Patil and J. Patil, "Feature-based malicious url and attack type detection using multi-class classification," in The ISC International Journal of Information Security, 2018, pp. 141–162.

[18] K. A. Djaballah, K. Boukhalfa, Z. Ghalem, and O. Boukerma, "A new approach for the detection and analysis of phishing in social networks: the case of Twitter," in 2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS), 2020.

[19] A. Zamir, H.U. Khan, T. Iqbal, N. Yousaf, F. Aslam, A. Anjum, M. Hamdani, "Phishing web site detection using diverse machine learning algorithms," The Electronic Library, vol. 38, no. 1, pp. 65–80, Jan. 2020.

[20] H. M. Junaid Khan, Q. Niyaz, V. K. Devabhaktuni, S. Guo, and U. Shaikh, "Identifying generic features for malicious URL detection system," in 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), 2019.

[21] Remosin, "Bot_Detection," Bot Detection. [Online]. Available: kaggle, https://www.kaggle.com/datasets/remosin/bot-detection. [Accessed November 24, 2022].

[22] A. Singh, "Dataset of Malicious and Benign Webpages," Dataset of Malicious and Benign Webpages. [Online]. Available: kaggle, https://www.kaggle.com/datasets/aksingh2411/dataset-of-malicious-and-benign-webpages. [Accessed November 24, 2022].

[23] S. Kumar, "Detect Malicious URL using ML," Detect Malicious URL using ML. [Online]. Available: kaggle, https://www.kaggle.com/code/siddharthkumar25/detect-malicious-url-using-ml. [Accessed November 24, 2022].

[24] T. Tiwari, "Phishing Site URLs," Phishing Site URLs. [Online]. Available: kaggle, https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls. [Accessed November 24, 2022].

[25] J. Antony, "Malicious_n_Non-Malicious URL," Malicious_n_Non-Malicious URL. [Online]. Available: kaggle, https://www.kaggle.com/antonyj453/urldataset. [Accessed November 24, 2022].

[26] A. R. Alsaidi, "Impact of the various measures of similarity on the statistic hierarchical neural response method." Journal of Theoretical and Applied Information Technology, vol. 99, no. 21, 2021.

[27] R. A. M. Alsaidi, H. Li, Y. Wei, R. Khaji, and Y. Y. Tang, "Hierarchical sparse method with applications in vision and speech recognition," Int. J. Wavelets Multiresolut. Inf. Process., vol. 11, no. 02, p. 1350016, 2013.

[28] Alhujaili, R. F., & Yafooz, W. M. (2022, May). Sentiment Analysis for YouTube Educational Videos Using Machine and Deep Learning Approaches. In 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI) (pp. 238-244). IEEE.

[29] Alhejaili, R., Alsaeedi, A., & Yafooz, W. (2023). Detecting Hate Speech in Arabic Tweets During COVID-19 Using Machine Learning Approaches. In Proceedings of Third Doctoral Symposium on Computational Intelligence (pp. 467-475). Springer, Singapore.

[30] Alhejaili, R., Alhazmi, E. S., Alsaeedi, A., & Yafooz, W. M. (2021, September). Sentiment Analysis of The Covid-19 Vaccine For Arabic Tweets Using Machine Learning. In 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO) (pp. 1-5). IEEE.