A Hybrid approach combining blocklists, machine learning and deep learning for detection of malicious URLs

Bronjon Gogoi
National Informatics Centre
Guwahati, Assam, India
asm-bronjon@nic.in

Tasiruddin Ahmed
National Informatics Centre
Guwahati, Assam, India
asm-tasir@nic.in

Arabinda Dutta
National Informatics Centre
Guwahati, Assam, India
dutta.a@nic.in

2022 IEEE India Council International Subsections Conference (INDISCON) | 978-1-6654-6601-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/INDISCON54605.2022.9862909

***Abstract*—Malicious URLs are one of the most commonly used methods of distributing malware and ransomware, launching phishing attacks, sending spam, and defacement of websites. Attackers send malicious URLS via email, advertisement, or embed them in legitimate websites. Unsuspecting users click malicious URLs and become targets of attackers. Detecting malicious URLs and alerting the user can prevent some of the attacks and hence detection of malicious URLs is an important part of cyber security. In this paper, a hybrid approach, combining traditional, machine learning, and deep learning methods for detecting malicious URLs is proposed. The traditional approach is a signature or a blocklist based approach that can detect existing malicious URLs. The signature-based approach is augmented with a shallow and a deep learning based approach that can detect new malicious URLs for which no signatures exist. The machine learning and deep learning based approach is trained and tested on a large dataset consisting of 2.5 million malicious and benign URLs. The combined system of traditional blocklist, shallow learning and deep learning approach achieves a precision, recall and f1- score of greater than 0.97.**

Keywords—Machine Learning, Deep Learning, Malicious URL Detection, Cyber Security

I. INTRODUCTION

Malicious URLs are used by attackers for launching phishing, spam and defacement attacks and also as a method of distribution of malware, ransomware, virus, trojans and other PUPs (potentially unwanted programs). Attackers target unsuspecting users by sending malicious URLs using different channels like email, SMS, as an embedded link in a legitimate website, and social media. An unsuspecting user on clicking a malicious URL can face financial loss, leakage of sensitive information, data loss, credentials loss among others. According to Forbes, malware attacks have increased by 358% in 2020, and there are more than 2,145,013 verified phishing sites in the Internet and a majority of these attacks use malicious URLs as a delivery mechanism[1]. In light of increasing cyber attacks, quick detection and classification of malicious URLs according to the type of attack they are used for can go a long way in limiting the number of such attacks.

To protect unsuspecting users from cyber attacks delivered via malicious URLs, many malicious URL detection schemes have been proposed and deployed with varying degree of success. Classification of URLs as malicious or benign can be performed using static or dynamic analysis. Dynamic analysis executes the page and then tries to find indicators of the page being malicious. If certain malicious features like malicious JavaScript is found, the page is flagged as malicious else it is marked as benign. The main drawback of dynamic analysis is that it is time consuming. Static analysis on the other hand analyses the features of the page without executing the page and is faster compared to the dynamic approach. Static analysis has the disadvantage that it may miss certain features because it does not execute the page and has to rely only on the lexical features of the page. Both static and dynamic analysis requires extracting the content of the page and analyzing it for detecting whether the page is malicious or not and this increases the time required to detect whether an URL is malicious. Users today expect faster response times and any system that takes a significant amount of time will not be preferred by users. An alternative to detecting whether an URL is malicious is using the URL itself. J. Ma, L. K. Saul, S. Savage, and G. M. Voelker in [1] has shown that a malicious URL itself has certain features that can be used to detect whether it is malicious or benign.

The method of classifying an URL based only on the information contained in the URL can be classified as, the traditional blocklist based approach and the newer machine learning approach. Both the traditional blocklist as well as the ML (Machine Learning) based approaches have their own advantages and disadvantages. A comparison of the two approaches is shown in Table 1. Blocklist based approach has the main disadvantage that they cannot detect new malicious URLs. ML based approaches has the main drawback that the accuracy of the model depends on the dataset that is used for training and depending on the dataset, such models may have significant false positive and false negative rates. Moreover, as new samples of malicious URLs are discovered, the ML models need to be retrained which is a time consuming task. ML models also takes time to make a prediction since it has to calculate the probability of the URL being a malicious URL by manipulating the input feature vector and the learned model. By using a blocklist based approach along with a ML model, the time to predict known malicious URLs can be reduced.

In this paper, a hybrid approach that combines both traditional blocklist and ML methods is proposed. The

1.https://www.forbes.com/sites/chuckbrooks/2021/03/02/alarming-cybersecurity-stats-------what-you-need-to-know-for-2021/?sh=fc4fbca58d3d

978-1-6654-6601-1/22/$31.00 ©2022 IEEE

TABLE. I. COMPARISON OF BLACKLIST AND ML APPROACHES

Approach	Advantages	Disadvantages
Blacklist	– Easy to implement – Detection rate for existing malicious URLs is 100 percent – Time to detect a malicious URL is fast as it just has to look up an URL in the blacklist.	– Cannot detect new malicious URLs. – Difficult to maintain an ever-growing list of blacklists. – Time to detect is higher than a blacklist approach
Machine Learning	– Can detect new and zero-day malicious URLs. – Detection rate of known malicious URLs depends on training dataset.	– Difficult to implement and requires expertise in AI technologies. – The performance depends on the training dataset.

traditional model will be used to detect known malicious URLs and the machine learning method will be used to detect new and unknown malicious URLs. The main contributions of this paper are –

- An approach combining the advantages of both traditional blocklist and modern machine learning models.
- The proposed approach uses an in-memory database for storing the blocklist, thereby reducing the time required to detect a known malicious URL.
- Use of a large dataset of around 2.5 million URLs for training the ML models.
- Use of both shallow and deep learning ML models allowing flexibility in deployment depending on the needs.

The remainder of the paper is organized as follows – Section II discusses related work, Section III discusses proposed approach, Section IV discusses Evaluation and finally, Section V discusses conclusions.

II. RELATED WORK

Currently, the traditional blocklist and ML learning methods are the most common methods used for detection of malicious URLs. Lately, deep learning machine learning methods are also being used for detection of malicious URLs. Blocklist approaches maintain a database of malicious URLs and prevents access to any URL present in the blocklist. The disadvantage of blocklist approach is that it is difficult to update the blocklist with new URLs frequently and quickly the size of the blocklist increases to a large size. There exists many publicly available malicious URL blocklist like Phishtank, URLHaus which can be used to develop a blocklist module for detection of malicious URLs. Among machine learning and deep learning many approaches have been proposed. But a majority of the approaches uses a very small dataset for training. Machine learning algorithms perform better when the number of training samples is high. The results reported by the majority of approaches are from experiments that uses a dataset of size less than 100000. M. Mamun, M. Rathore Et. al used shallow ML methods achieving an accuracy of 0.99 but their dataset was small in [2]. A. S. Manjeri, R. Kaushik, A. Mnv, and P. C. Nair and proposed a shallow ML based approach but the dataset used was very small consisting of 1750 URLs and their features included host-based features [3]. S. Egan and B. Irwin proposed a malicious URL detection approach based only on URL based features using Perceptron and Confidence Weighted Algorithm in [4] and again the dataset used for evaluation is too small and the dataset is not publicly available. G. Chakraborty and T. T. Lin combined SVM with Pareto GA and LASSO for classification using lexical and host-based features in [5] and used a dataset of around 45000 URLs. P. L. Indrasiri, M. N. Halgamuge, and A. Mohammad used an ensemble method in [6] and achieved an accuracy of 98.27 percent using a combination of lexical features, host-based, HTML/JavaScript and abnormality features on a moderate dataset of size around 175000 URLs. A. D. Gabriel, D. T. Gavrilut, B. I. Alexandru, and P. A. Stefan used a large dataset of around 2000000 URLs but the method achieved a low accuracy of around 82 percent and high False positive rate of 15 percent in [7]. F. Alkhudair, M. Alassaf, R. Ullah Khan, and S. Alfarraj used shallow ML model of Random Forests and achieved an accuracy of 96% using host-based and lexical features but the dataset size is not mentioned [8]. A. Das, A. Das, A. Datta, S. Si, and S. Barman used a deep learning ML model of CNN, LSTM and CNN-LSTM with CNN-LSTM achieving the highest accuracy of 93.59 precent but the dataset used is small for a deep learning approach and consisted of 194798 URLs [9]. M. Darling, G. Heileman, G. Gressel, A. Ashok, and P. Poornachandran used J48, Naïve Bayes, LR, Bayesian LR, KNN and achieved 99.3 AUC with J48 in [10]. Y. Ma, Q. Guan, F. Guo, and G. Zhang used Sparrow Search Algorithm and achieved a accuracy of 91.4 percent in [11]. W. Yang, W. Zuo, and B. Cui proposed a deep learning based approach based on Gated Recurrent Units (GRU) in [12] but the URLs are targeted at exploiting server side vulnerabilities like SQL injection, directory

```
Algorithm1: Predict if an URL is Malicious
Input: URL
Output: Malicious or Benign Prediction
hash = SHA1(URL)
if hash in BLACKLIST_DATABASE
   prediction = MALICIOUS
   return prediction
end if
if hash not in BLACKLIST_DATABASE
   if model == SHALLOW:
      features = extract_features(URL)
      prediction = predict_shallow(features)
      return prediction
   end if
   if model == DEEP:
      prediction = predict_deep(URL)
      return prediction
endif
```

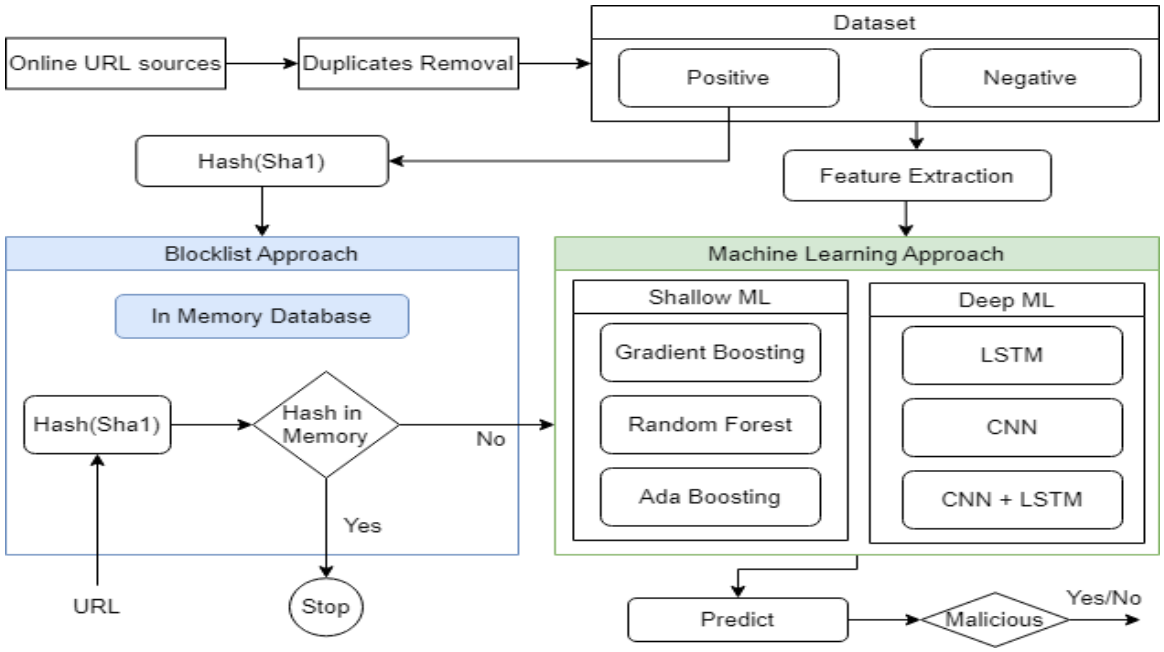

Fig. 1. Architecture of the proposed System

browsing etc. rather than attacking clients side browsers. T. Manyumwa, P. F. Chapita, H. Wu, and S. Ji used ensemble learning algorithms of XGBoost, AdaBoost, CatBoost and LightGBM and achieved the highest accuracy of 96.86 using XGBoost using a dataset of 126983 URLs in [13].

III. PROPOSED APPROACH

The architecture of the proposed approach is shown in Figure 1. Malicious and benign URLs are collected from various online sources including Phistank2, and GitHub repositories. The ISCX-URL2016[1] is also used which contains 35000 benign URLs and 69000 malicious URLs. The data set in repository[2] contains 2000000 malicious URLs and 500000 benign URLs. The URLs collected are then passed through a deduplication process to eliminate duplicates. Following this, the URLs are divided into positive and negative samples. The complete dataset of positive and negative URLs is used by the machine learning module for training the classifiers. The positive set is used by the blocklist module. The positive URL set contains malicious URLs. The URLs are hashed and stored in an in-memory database. When a URL is presented to the system, it is passed to the blocklist module. The URL is hashed and checked if the hashed value is present in the in-memory database. If the hash of the encountered URL is present in the HASH, the URL is flagged as a malicious URL. If the hash of the URL is not determined to be malicious by the blocklist module, it is passed to the machine learning module. The machine learning module first extracts the features from the URL and then classification is performed. Based on the prediction of the classifier, the URL is flagged as either malicious or benign. The Algorithm 1 is used for detecting if a given URL is malicious or benign. The backlist module and the machine learning modules are discussed in section III(A) and III(B).

1. https://www.unb.ca/cic/datasets/url-2016.html
2. https://github.com/cysecmldev/cybersec datasets

A. *Blocklist Module*

The backlist module uses an in-memory database of malicious URLs. When an URL is presented to the system, it hashes the URL and performs a lookup in the in-memory database. If the URL is present, it is quickly identified as malicious and the system can immediately respond. The ML module requires time to extract feature from the URL and then make a prediction. The blocklist module saves this time when the URL is a known malicious URL. The database used by the blocklist module is periodically updated with new malicious URL as when they are discovered. The steps for creation of the blocklist modules in-memory database is shown in Figure 1. Labelled URLs are collected from various online sources. Duplicate URLs can interfere with the learning process of the machine learning models so a duplication removal process is run on the collected URLs. From the labelled dataset, the positive only set is used by the blocklist module. The blaclist module takes an URL from the positive set and hashes the URL using SHA1 hashing algorithm. SHA1 is used because it is a very fast algorithm and the approach needs to hash a large number of URLS.The hased URL is stored in an in-memory datadase that provides very quick read and write operations. This process of taking an URL, hashing and storing in the in-memory database is repeated for all the URLs in the positive set. When all URLs are hased and stored, the blocklist database is ready to used by the blocklist module. Now when an URL is presented, the blocklist module hashes the incoming URL and performs a lookup on the in-memory database. If the URL is found, it is flagged as malicious and if the URL is not found in the blocklist module, it is passed on to the ML module for prediction.

B. *Machine Learning Module*

New malicious URLs are discovered and reported every day and such it is not possible to immediately update the blocklist module to include the newly discovered malicious URL. The machine learning module is used to classify a URL

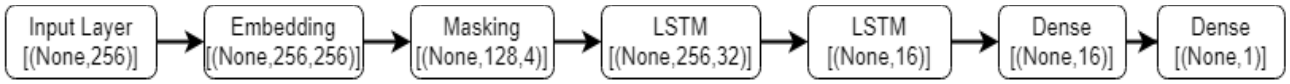

Figure 2. LSTM network architecture

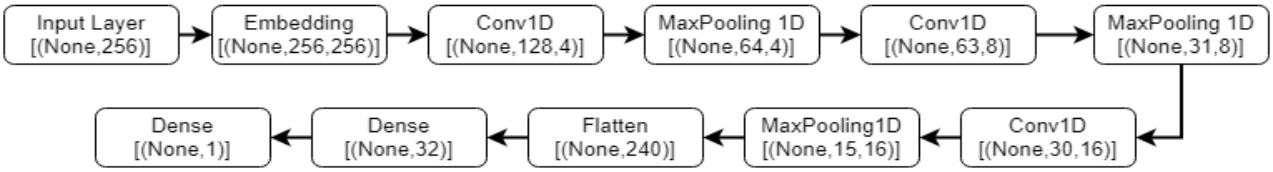

Figure 3. CNN network architecture

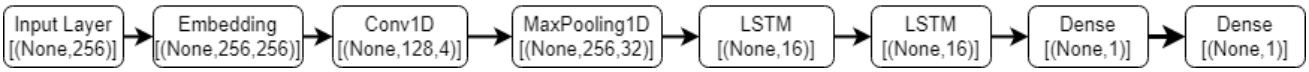

Figure 4. CNN-LSTM network

as either malicious or benign when the blocklist module cannot identify that URL as malicious. The ML module can be configured to use either shallow learning models or deep learning models. The shallow learning models can be trained quickly but requires feature extraction for obtaining the best results. The deep learning models requires time to be trained

and also requires larger datasets than the shallow models but feature extraction process is simplified. The shallow learning models and the deep learning models have comparable performance in terms of classification accuracy and either model be used. The shallow and deep learning models also periodically retrained as new URLs are discovered and added to the system.

a) Shallow ML models: The shallow ML models used in the proposed system includes Gradient Boosting and Random forests algorithms. Gradient Boosting methods are the best performing shallow machine learning algorithms for tabular data. To train the shallow ML models, feature extraction is performed on the positive and negative datasets. Commonly used features for detection of maclious URLs are host based features and lexical features. In the proposed approach, host based features are not used because to obtain the host based features, information related to the host of the URL like domain information, whois information needs to be queried. The proposed system aims to quickly detect if a URL is malicious and hence, the host based features are not used. The following categories of Lexical features are extracted from the URLs –

- *Number based* – The number-based features include number of digits in the URL, number of parameters, number of fragments, number of encoded characters, number of subdirectories, URL length, path length, URL host length and number of periods.
- *Miscellaneous Features* – The miscellaneous features include URL scheme, is host IP address, has port in URL, is URL encoded, URL string entropy.
- *Keyword Features* - The keyword-based features are Boolean features which indicate whether any of the keywords viz. admin, login, server, login. These are some common keywords used in Phishing URLs.

Following feature extraction, the machine learning models of Gradient Boosting, Ada Boosting, and Random Forest are trained. Gradient Boosting, Ada Boosting and Random forests are ensemble algorithms. Ensemble algorithms perform better than single algorithms and are also robust. Classification of URLs into malicious or benign is inherently an unbalanced classification task because the number of malicious URLs will always be less than the number of benign URLs. Ensemble methods are also better suited for such unbalanced datasets.

b) Deep ML models: The deep learning models used in the proposed system includes CNN, LSTM and CNN-LSTM. The advantage of the deep learning models over shallow learning is that the deep learning models does not require extensive feature engineering or feature extraction techniques. In the proposed approach, the deep neural network models are character based. The URLs are first integer encoded and then passed to an embedding layer. The embedding layer then transforms the integer encoded strings to dense vectors which can be used for training the deep neural models. The data repsesentation for each of the models CNN, LSTM and CNN-LSTM are different and are discusses next.

1) CNN: The proposed approach uses an 1D convoluational neural network for classifying the URLs as either malicious or benign. 1D convolutaional networks are widely used for text classification. The input to the CNN is the character embeddings obtained after transforming the URLs into integer encoder vectors.The arhictecture of the CNN used in the proposed approach is shown in Figure 2. The output shape of each layer is shown below the layer name. The CNN network contains one Embedding Layer, three pairs of 1D Convolutional layer and Max Pooling Layer, followed by a Flatten and two Dense Layers. The final Dense layer uses a Sigmoid layer which outputs and value in the rage 0.0 to 1.0. All outputs with a value greater than equal to 0.5 is considered malicious and all values smaller than 0.5 is considered as benign.

2) LSTM: The LSTM neural network based module has 7 layers including the input layer. The architecture of te network is shown in Figure 3. LSTM network require inputs in form of sequences. In the proposed approach, the input URLs are converted into fixed length dense vectors of length 256 by the embedding layer which is then passed to the LSTM layer. The masking layer is used to mask the zero values which are used for padding when the length of an URL

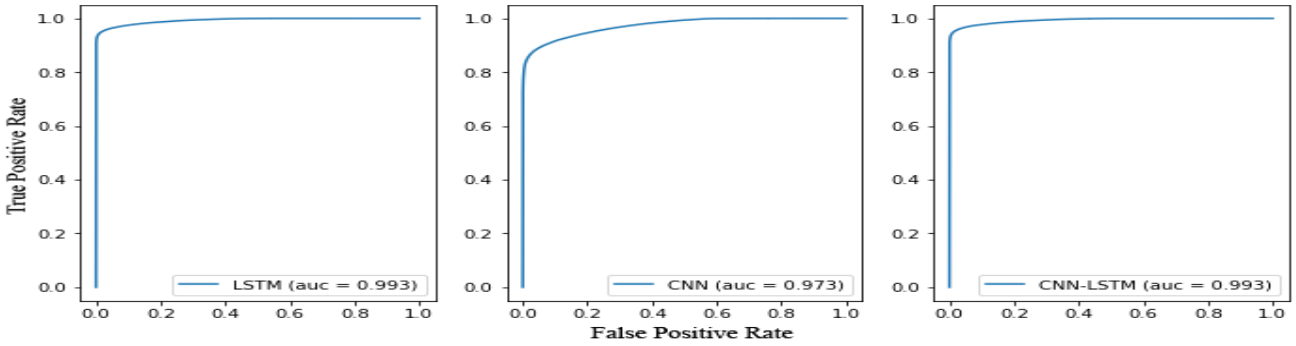

Fig. 5. Deep Learning Models – LSTM, CNN, CNN-LSTM AUC-ROC curve

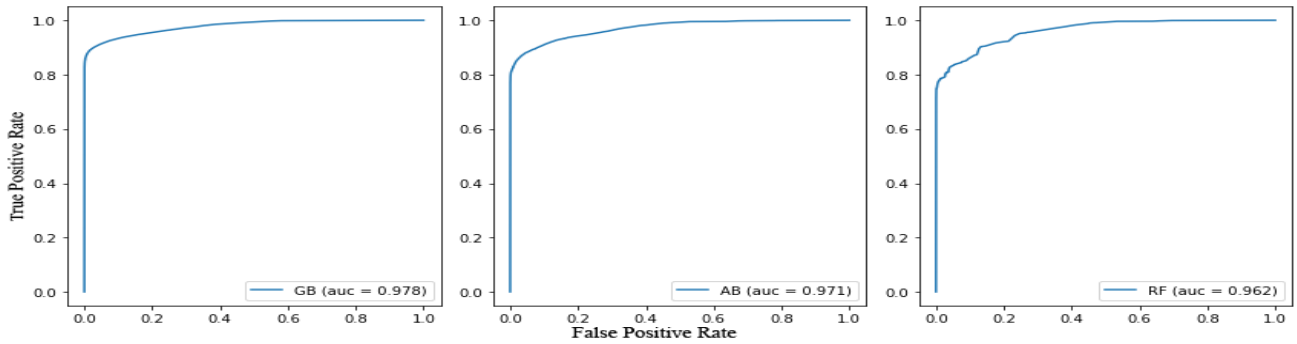

Fig. 6. Shallow Learning Models – Gradient Boosting, Ada Boosting, Random Forest AUC-ROC curve

is less than 256. The LSTM layers are followed by two dense layers. The final layer is Dense layer with a Sigmoid activation function that outputs a value in the range 0.0 to 1.0. Values greater than equal to 0.5 are considered Malicious and values less than 0.5 are considered benign.

3) CNN-LSTM: The final model in the ML module is the CNN-LSTM model which is a combination of 1D convolutional network and an LSTM network. The architecture of the CNN-LSTM model is shown in Figure 4. Followed by the Embedding Layer, the model contains an 1D convolutional layer followed by a 1D Maxpooling Layer. These two layers are followed by 2 LSTM layers. The input format is similar to the CNN model discussed above. The final layers include two dense layers. The final dense layer contains a single neural with a Sigmoid activation function which outputs values in the range 0.0 to 1.0. Similar to other models, any value greater than equal to 0.5 is considered malicious and any values less than 0.5 is considered benign.

The machine learning models' performances are dependent on hyper paramters. The hyper parameters used for the shallow and deep learning models are shown in Table II and Table III respectively. The deep learning models were implemented using Tensorflow and Keras deep learning frameworks while the shallow ML models were implemented using the sklearn framework.

IV. Evaluation

The proposed system's performance is evaluated in terms of the performance of the machine learning module since the blocklist module is a traditional rule-based module and its performance depends on the size of the blocklist. The performance of both the shallow ML models and the deep ML models are measured in terms of the metrics of accuracy, precision and recall. The values of precision, recall and f1-score for the shallow ML models are shown in Table IV. Among the shallow models, Gradient Boosting has the highest performance. The AUC-ROC curve also provides vital information regarding the performance of the classifier. The AUC-ROC curve plots the TPR (True Positive Rate) along y-axis and the false positive along the x-axis. The AUR-ROC curve for the shallow models is shown in Figure 6. The greater the AUC (area under the curve), the better the

TABLE. II. Hyper paramters of SHallow Models

Hyper parameter	Value
Number of Estimators	300
Learning Rate	0.3
Max Depth	2

TABLE. III. Hyper parameters of deep models

Hyper Parameter	Value
Number of Epochs	20
Optimizing Algorithm	Adam
Learning Rate	0.001

TABLE IV. Performance metrics of shallow models

Model	Precision	Recall	F1-Score
Gradient Boosting	0.97	0.97	0.97
Ada Boosting	0.97	0.97	0.97
Random Forest	0.96	0.96	0.96

TABLE V. Performance metrics of deep models

Model	Precision	Recall	F1-Score
LSTM	0.99	0.99	0.99
CNN	0.97	0.97	0.97
CNN-LSTM	0.99	0.99	0.99

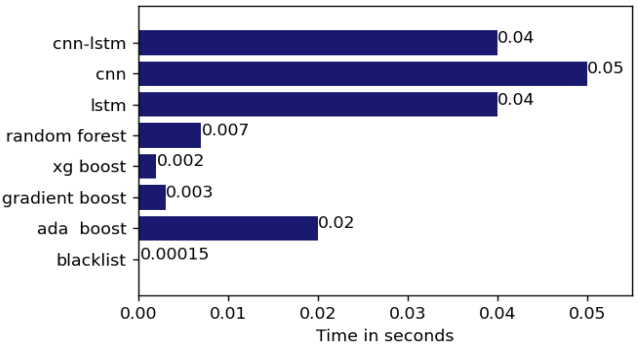

Figure 7. Time taken for detection in seconds

performance of the classifier. The shallows models of Gradient Boosting, Ada Boosting and Random Forest classifier achieves an AUC of 0.97,0.96 and 0.97 respectively. The AUC-ROC curve indicates that the false positive rate is quite low for the shallow models and there is no overfitting or underfitting.

The values of precision, recall and f1-score achieved by the deep ML models are shown in Table V. The AUC-ROC curves for the deep models are shown in Figure 5. The LSTM, CNN and CNN-LSTM deep ML models achieves an AUC of 0.99,0.97 and 0.99 respectively. The AUC-ROC curves indicates that the models are able to generalize well and there is no overfitting or underfitting. The false positive rate is also quite low indicating the model can accurately classify majority of the samples correctly. The time taken by various models to predict whether a given URL is malicious is shown in Figure 7. From figure 7, it can be seen that the blocklist approach can detect if a URL is malicious with a millisecond. The shallow ML models are the second fastest group and deep learning models are the slowest. Depending on the response or prediction time requirement, the blocklist and any one of the machine learning models can be combined for implementing the detection module.

V. Conclusion

This paper proposes a hybrid approach combining the traditional blocklist approach and modern machine learning approach for detection of malicious URLs. Blocklist module detects known malicious URLs quicky whereas the ML module detects unknown URLs. This hybrid approach of detection offers the benefits of both the traditional blocklist approach and modern blocklist approach. The ML models are configurable and the system can be configured to use either shallow ML models or deep ML models depending on the requirement. The shallow ML and deep ML models achieved an accuracy of 0.97 and 0.98 respectively and given the size of the dataset, it is an impressive accuracy. The experimental results obtained also indicates that with proper feature extraction, shallow machine learning algorithms can achieve accuracy comparable to deep learning algorithms for the purpose of detection of malicious URLs. In the proposed approach, inclusion of more features can improve the performance of the shallow models. The future roadmap includes extending the proposed approach to include more state-of-the-art deep learning algorithms and improved feature extraction for shallow ML models and development of a plugin-based system deployable as a browser extension or as a reverse proxy-based system.

References

[1] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Beyond blocklists: Learning to detect malicious web sites from suspicious urls,” in *Proceedings of the SIGKDD Conference*, 2009a, pp. 1245–1254.

[2] M. Mamun, M. Rathore, A. Habibi Lashkari, N. Stakhanova, en A. Ghorbani, “Detecting Malicious URLs Using Lexical Analysis”, *International Conference on Network and System Security*,09 2016, vol 9955, bll 467–482.

[3] A. S. Manjeri, R. Kaushik, A. Mnv, and P. C. Nair, “A Machine Learning Approach for Detecting Malicious Websites using URL Features,” *Proc. 3rd Int. Conf. Electron. Commun. Aerosp. Technol. ICECA 2019*, pp. 555–561, 2019, doi: 10.1109/ICECA.2019.8821879.

[4] S. Egan and B. Irwin, “An evaluation of lightweight classification methods for identifying malicious URLs,” *2011 Inf. Secur. South Africa - Proc. ISSA 2011 Conf.*, 2011, doi: 10.1109/ISSA.2011.6027532.

[5] G. Chakraborty and T. T. Lin, “A URL address aware classification of malicious websites for online security during web-surfing,” *11th IEEE Int. Conf. Adv. Networks Telecommun. Syst. ANTS 2017*, pp. 1–6, 2018, doi: 10.1109/ANTS.2017.8384155.

[6] P. L. Indrasiri, M. N. Halgamuge, and A. Mohammad, “Robust Ensemble Machine Learning Model for Filtering Phishing URLs: Expandable Random Gradient Stacked Voting Classifier (ERG-SVC),” *IEEE Access*, vol. 9, pp. 150142–150161,2021,doi: 10.1109/ACCESS.2021.3124628.

[7] A. D. Gabriel, D. T. Gavrilut, B. I. Alexandru, and P. A. Stefan, “Detecting malicious URLs: A semi-supervised machine learning system approach,” *Proc. - 18th Int. Symp. Symb. Numer. Algorithms Sci. Comput. SYNASC 2016*, pp. 233–239, 2017, doi: 10.1109/SYNASC.2016.045.

[8] F. Alkhudair, M. Alassaf, R. Ullah Khan, and S. Alfarraj, “Detecting Malicious URL,” *2020 Int. Conf. Comput. Inf. Technol. ICCIT 2020*, pp. 7–11, 2020, doi: 10.1109/ICCIT-144147971.2020.9213792.

[9] A. Das, A. Das, A. Datta, S. Si, and S. Barman, “Deep Approaches on Malicious URL Classification,” *2020 11th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2020*, pp. 1–6, 2020, doi: 10.1109/ICCCNT49239.2020.9225338.

[10] M. Darling, G. Heileman, G. Gressel, A. Ashok, and P. Poornachandran, “A lexical approach for classifying malicious URLs,” *Proc. 2015 Int. Conf. High Perform. Comput. Simulation, HPCS 2015*, pp. 195–202, 2015, doi: 10.1109/HPCSim.2015.7237040.

[11] Y. Ma, Q. Guan, F. Guo, and G. Zhang, “Malicious URL Classification Model Based on Improved Sparrow Search Algorithm,” *ICEIEC 2021 - Proc. 2021 IEEE 11th Int. Conf. Electron. Inf. Emerg. Commun.*, pp. 21–25, 2021, doi: 10.1109/ICEIEC51955.2021.9463841.

[12] W. Yang, W. Zuo, and B. Cui, “Detecting Malicious URLs via a Keyword-Based Convolutional Gated-Recurrent-Unit Neural Network,” *IEEE Access*, vol. 7, pp. 29891–29900, 2019, doi: 10.1109/ACCESS.2019.2895751.

[13] T. Manyumwa, P. F. Chapita, H. Wu, and S. Ji, “Towards Fighting Cybercrime: Malicious URL Attack Type Detection using Multiclass Classification,” *Proc. - 2020 IEEE Int. Conf. Big Data, Big Data 2020*, pp. 1813–1822, 2020, doi: 10.1109/BigData50022.2020.9378029.