

Comparison of the efficiency of machine learning algorithms for phishing detection from uniform resource locator

Ahana Nandi Tultul¹, Romana Afroz¹, Md Alomgir Hossain²

¹Department of Computer Science and Engineering, College of Engineering and Technology (CEAT), International University of Business Agriculture and Technology (IUBAT), Dhaka, Bangladesh

²Department of Computer Science and Engineering, Faculty of Computer Science and Engineering, International University of Business Agriculture and Technology (IUBAT), Dhaka, Bangladesh

Article Info

Article history:

Received Feb 9, 2022

Revised Jun 4, 2022

Accepted Jul 22, 2022

Keywords:

Comparing among KNN

Decision tree

Deep neural decision forest

Machine learning

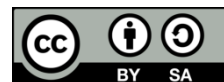
Random forest

Support vector machine

ABSTRACT

We are using cyberspace for completing our daily life activities because of the growth of Internet. Attackers use some approaches, such as phishing, with the use of false websites to collect personal information of users. Although, software companies launch products to prevent phishing attacks, identifying a webpage as legitimate or phishing, is a very difficult and these products cannot protect from attacks. In this paper, an anti-phishing system has been introduced that can extract feature from website's URL as instant basis and use four classification algorithms named as K-Nearest neighbor, decision tree, support vector machine, random forest on these features. According to the comparison of the experimental results from these algorithms, random forest algorithm with the selected features gives the highest performance with the 95.67% accuracy rate. Then we have used one deep learning algorithm as enhanced of our experiment named as deep neural decision forests which have given performance with the 92.67% accuracy rate. Then we have created a system which can extract the features from raw URL and pass the features to our deep neural decision forest trained model and can classify the URL as Phishing or legitimate.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Ahana Nandi Tultul

Department of Computer Science and Engineering, College of Engineering and Technology (CEAT)

International University of Business Agriculture and Technology (IUBAT)

Dhaka, Bangladesh

Email: ahananondi6@gmail.com

1. INTRODUCTION

We are using internet for completing our daily life activities such as electronic banking, social media communications etc. because of the growth of Internet. Attackers use different techniques, such as phishing, by using phishing websites to collect personal information of users like username, password, and Credit card number. Phishing attack is a type of cyber-attack which targets general users fooling them to give personal information like username, credit card number, national identify card number, debit card number, password etc. Every year, general users loss billions of moneys for this phishing attacks [1]. Crackers trick the users by sending uniform resource locator of a similar website as a legitimate website to get personal information. As like as the general phishing where fisherman give bait to catch fish, crackers send this type of websites and website's URL as bait and waits to anyone would enter and give personal information. Sometimes, they also attack general mobile users for increasing their efficiency [2]. For example, the URL for google.com, they replace 'oo' with 'ooo' to trick with end-user and set phishing site. Instead of entering google.com when end-users enter google.com and enter their google account those end-users' credentials will go to attackers.

Anti-phishing working group (APWG) is a non-profit organization that inspect phishing attacks reported by its member companies such as Kaspersky Lab, Iron-Key, and Internet Identity. According to the latest APWG (June 2021) Q1 report, APWG saw that there were 222,127 attacks in June 2021, which was the third-worst month in the history of APWG's reporting [3]. Often end-users become the victim of phishing attacks because of five causes such as, end-users don't have overall understanding about phishing-URLs, they don't understand whether a website is trustworthy or not, it is not possible for them to see the full URL of the website because of the structure of Hypertext Markup Language, they have less time for analyzing URLs and they cannot differentiate the appearance of phishing website from the legitimate website [4]. But aware of these five reasons cannot protect them to fall in phishing attacks [5]. Thousands of email users are getting fake emails from attackers, as the attackers are hoping that certain numbers of user would enter to that URL [6]. Spear Phishing attacks are done by attackers by targeting an individual group of people or community as a part of an organization or a company [7]. End Users get messages from attackers over social media or any other communication media by pretending to be a familiar face of the company, they make communication and ask for sensitive data related to that company [8]. There are many ways to create anti-phishing techniques or tool. These can be categorized mainly in four categories, such as List-based techniques: There are two types of lists used in list-based anti-phishing technique, mainly used by browsers. Those are: first one is whitelist that contains legitimate URLs as a list which can be visited by anyone who wants to visit, by the browsers, and second one is blacklist which contains harmful URLs. The browser accessed and download only those websites whose URL is stored in the whitelist as legitimate website's URL [9].

Visual similarity-based Approach: In this anti-phishing technique, a comparison has to be made between suspicious website image with legitimate website image database to get similarity ratio, the scores of ratio will be used for the classification of suspicious websites. When the similarity score is greater than a certain threshold, the website will be classified as phishing else it will be legitimate [10], [11]. But a problem is that it takes a huge storage for storing images. Heuristic-based techniques: In this technique, features are to be collected from the phishing website based on some rules or characteristics of the phishing attacks, phishing website or phishing website URL [12]-[14]. If any phishing website don't have same features, then this mechanism gives poor result as detection rate. Machine learning-based techniques: Recently, researchers are researching on the use of machine learning algorithms (ML) to detect phishing attacks by applying them into extracted features from the websites [15]. Heuristic method and machine learning are combined here where heuristic method is used to extract information from website and its URL to make dataset, then dataset is feed to machine learning algorithm for detection and classification.

2. BACKGROUND STUDY

As our proposed system is based on ML based techniques, we will discuss some of the papers that represents existing ML techniques which are used in the detecting phishing websites. Zhang *et al.* [16] in this paper, they proposed a system name as CANTINA which looks the content of a web page. For this purpose, it uses time frequency-inverse document frequency (TF-IDF) algorithm. When someone is locating any webpages, they can use the URL following a common structured. If the resources that one URL is locating, can't be found in the web, then one can search the signature terms by a browser to find out the website. CANTINA collects major terms by using the term frequency-inverse document frequency algorithm and then those terms were searched by browsers. If it shows the websites, it was noted to be a legitimate website. But this system is limited in English language.

Guang *et al.* CANTINA+ (2011) [17]. In this paper, they proposed an extended version of CANTINA. CANTINA+ is a comprehensive feature-based way where they have applied two filters. First one is a filter based on hash which compares a webpage against known phishing websites and second one finds out the HTML for login form. If the webpage is not detected as duplicated nearly to any other webpage, but it has login-form, then they extract 15 features from that webpage. Six machine learning algorithms named as logistic regression, decision forest, random forest, support vector machine, Bayesian network, Adaboost are used here. The limitation of this papers is small dataset around 8,118 phishing and 4,883 legitimate web pages. Though the system gives 92% accuracy, it gives a high false positive rate. Le, *et al.* (2011) [18].

In this paper, a system has been proposed name as 'PhishDef' which use lexical features and external features to identify phishing sites. They have taken 6,083 malicious URLs and 8,155 legitimate URLs from different sources. Their external features have been taken from WHOIS and Team Cymru. They have used different machine learning algorithm. Among them, adaptive regularization of weights (AROW) has work best and they work with it. Jeeva and Rajsingh [19], in this paper, they have introduced a system which is based on rule mining. They have obtained rules by interpreting features that are commonly seen or common characteristics for phishing URL. The proposed method has been consisted of two phases: i) URL search phase in which one request for URL if the URL is in legitimate URL's directory, then the URL is

identified as trustworthy URL. If not, then the URL goes to next phase and ii) feature extraction in which 14 heuristics features has been extracted from that URL and stored for applying rule to find out the URL as legitimate or phishing URL. Here, feature extraction phase includes features such as length of host, number of slashes, dots of host name, IP address etc. By using apriori algorithm, rule mining was established on the gathered features. Their experimental result shows 93% of accuracy of phishing detection.

Their main limitation is that the classification depends on quality of rules and they use 1,200 phishing URLs and 200 legitimate URL which is a limited dataset Jain and Gupta [20]. In this paper, they introduced a machine-learning based phishing detection way which collect features from client-side webpage content and page source code without using any third-party sources. They have used 4,059 phishing and legitimate websites as dataset. They have used a web crawler to obtain the feature from websites source code. This system downloads the total website including its URL and source code, then they extract URL features like number of dots, special symbol, suspicious keyword present in URL and also extract HTML source code features like presence of fake login form, and number of hyperlinks random forest (RF), support vector machine (SVM), neural networks, logistic regression and Naïve Bayes algorithm has been used in this system. Among these algorithm, random forest has worked most effectively and is used for trained the ultimate model. Their main limitation can be point out as this system need to download the whole webpage html source code and also content. Sundaram [21] in this paper, an efficient feature-based machine learning framework has been introduced. This study compares accuracy, f1 scores, guessing and remembering of Naïve Bayes (NB) and random forest (RF) to detect criminal messages to steal credential information. Here they have used a small dataset of 11,055 URLs with 6,157 cases of identity theft and 4,898 legal cases named as 'Phishing Websites Dataset' from the UCI Machine study library.

3. METHOD

3.1. Features extraction and data processing in proposed model

In this proposed system, feature extraction has been done from raw URL as input. In order to identify the relevant features, analyzation has been done on phishing websites dataset published: 24 September 2020| Version [22]. From the data source, after analyzing the dataset, we have taken 24748 data as training data set and 10,607 data as testing data set of dataset_small.csv. Based on the heuristics, twenty-three features have been selected for determining website as phishing or legitimate sites. We can divide them into three categories.

3.1.1. Third-party services based features

First seven features have been collected from Python's third-party services which we are calling as Third-party Service Based Features. They are as follows:

- Feature-1: TLS/SSL Certificate: Transport layer security (TLS) is a protocol that provides communication between two computers which is encrypted on the communication path or internet. It checks the credential to identity of the right destination server or computer so that hackers or cracker cannot intercept and get any data. Secure Sockets Layer or SSL is the previous version of TLS.
- Feature-2: Time to Live Hostname (ttl_hostname): Time to live refer to how long DNS setting are supposed to be cached before they are automatically refreshed.
- Feature-3: Number of Resolved Nameservers (qty_nameservers): A nameserver is a server that is responsible to identify address of a domain through the internet by translating and mapping human readable domain name to corresponding IP address of the corresponding web server to find a website, communicate across the network and return to the desired web address. One website can be connected to many nameserver.
- Feature-4: Number of resolved Ips: Resolved IP means mapping IP address to the domain name. one domain name can have multiple IP addresses resolved to it.
- Feature-5: Time (in days) of Domain Activation: Domain activation means domain name has been added to zone file and name server has been added to domain name. Here time of domain activation in days means, from present day how many days has it been passed after the domain has been activated.
- Feature-6: Time (in days) of Domain Expiration: After a limited time, activated domain will come to its expiration limit and if it is not renewed the domain name cannot be used. Here time of domain expiration in days means from the present days how many days left to expire the domain.
- Feature-7: AS Number (or ASN): An autonomous system number (ASN) is a unique number assigned to an autonomous system (AS) by the Internet Assigned Numbers Authority (IANA).

3.1.2. URL based features

Next thirteen features have been taken from URL which is called as 'URL Based Features'. These features are: quantity of dot in URL, quantity of hyphen in URL, quantity of underline in URL, quantity of slash in URL, quantity of question mark in URL, quantity of equal in URL, quantity of at in URL, quantity of and in URL, quantity of space in URL, quantity of tilde in URL, quantity of plus in URL, quantity of dollar in URL and length of URL. These features have been taken from the structure of URL. These features are important to detect phishing URL properly.

3.1.3. Domain based features

Last three features have been taken from Domain which is called as 'Domain Based Features'. These features are: quantity of dot in Domain, quantity of hyphen in Domain and length of domain. These features have been taken from the structure of domain of URL. These features are important to detect phishing URL properly. The algorithm for extracting feature from a URL in Algorithm 1.

Algorithm 1: The algorithm for extracting feature from a URL

Start:

Step-1: Call the Function Feature Extraction (URL):

Step-2: Find the domain name of the url

Step-3: If the url start with "https" then ttl_ssl_certificate=1 else ttl_ssl_certificate=0

Step-4: Find ttl_hostname by using dns.resolver.query() in domain

Step-5: Find qty_nameserver by using dns.resolver.query() with domain and "NS" as parameter

Step-6: Find qty_ip_resolved by using socket.gethostname_ex() with domain as parameter

Step-7: Find time activation and expiration by using WHOIS

Step-8: Find asn_ip by using Net()

Step-9: URL based feature by use URL.count(feature_symbol) and len(URL)

Step-10: Domain based feature by use domain.count(feature_symbol) and len(domain)

3.2. Used algorithms

We have used four algorithms as K-nearest neighbor, decision tree, support vector machine, random forest and compare their work efficiency based on their accuracy level. Based on their performance, we have used Deep neural decision forest, which is a new comer deep learning algorithm in this field, in our model. Now, we are going to discuss these five algorithms.

3.2.1. K-nearest neighbor (KNN)

In the training phase, KNN learns the similarity of the input labels for each output labels. In testing phase, we give our test data, according to its learning pattern. It measures the distance of the similarity of the new datapoint and put the new data to that category which is most similar or closer of the available categories [23]. Here, we use Minkowski distance between two features (X) and labels (Y) which can be defined as:

$$(\sum i = 1n |Xi - Yi|^p)^{1/p} \quad (1)$$

here i is the data and if $p=1$, then it is Manhattan distance or if $p=2$, then it is Euclidian Distance.

3.2.2. Decision tree

It is a tree like structure which tree can be explained by two parts as decision node and leafs where decision nodes split data and leaf are the final outcomes or outputs [24]. The data is continuously divided basis on the certain parameter in this algorithm. We have used "Gini Impurity Measure" method for splitting out data. Here Gini impurity tells us that what is the probability of misclassification of an observation. Here, only one tree structure flowchart will be produced basis on the impurity checking and final leaf will be either phishing or legitimate label. The decision tree can be normalized in decision rules as:

- If condition1 and condition2 and condition3 then outcome.
- If we consider as D our dataset where k is total classes, $p(i)$ is the probability of getting any class of a data then we can write Gini impurity as (2).

$$\text{Gini}(D) = 1 - \sum_{i=1}^k p(i)^2 \quad (2)$$

3.2.3. Support vector machine (SVM)

It is another supervised machine learning algorithm which place each features considering as a data point in n -dimensional space where n is the total features number. Here we have 23 features, with the value of each feature being the value of each co-ordinate. In the training data, the algorithm plots the data and features as coordinate and find out the right-hyper plane or the best fitted line to separate or distinguish between phishing characteristics and legitimate characteristics to classify and predict phishing and legitimate URL in the test dataset [25].

3.2.4. Random forest

Random forest refers to as an ensemble learning which is a technique to combine many classification algorithms to provide solution of difficult situation and improve accuracy of machine learning algorithm [26]. A random forest is the combination of many decision trees. The training data is fed to random forest algorithm and it produces a number of decision tree. The final output of random forest is done by summing the maximum number of decision tree gives which is result. Here, in this system, the output is chosen by checking the majority of the decision trees gives which result.

3.2.5. Deep neural decision forest

Deep neural decision forest is an algorithm which combines properties from representation learning known as deep architectures with divide-and-conquer principle from decision trees [27]. Here, in this one, in the upper level it has deep convolution network and it lower-level it has number of decision tree or in a word random forest. Deep neural decision forests combines the classification trees and representation learning functionalities known from deep convolution neural network. It has differentiable trees with the initial representation learning conduct in the deep neural decision forest. This is different from convolutional neural network because here the final prediction has been made on the decision trees.

3.3. Proposed model

This system take URL as input. We extracted features in three categories as Third-party based features, URL based features and Domain based features. Then we feed that data to previously trained model of machine learning algorithm and, detected and labeled as ‘Phishing’ or Legitimate. This is illustrated as Figure 1.

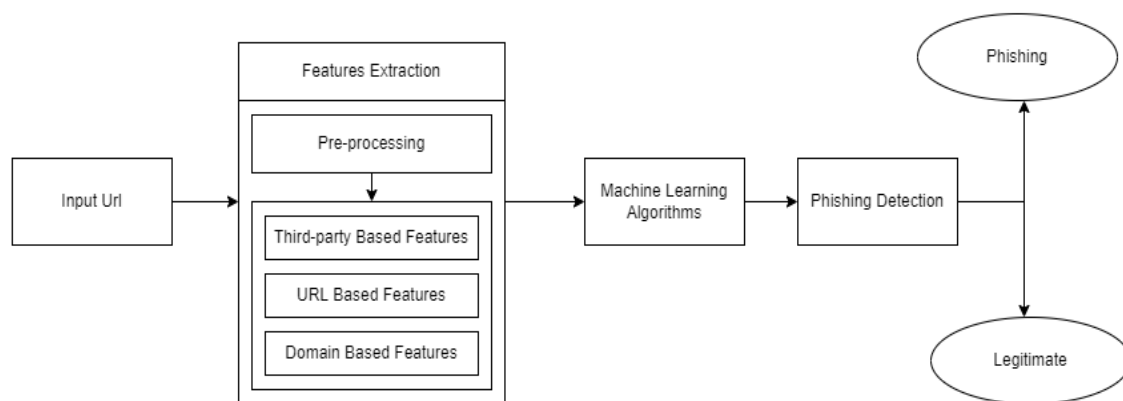


Figure 1. Block diagram of ‘comparison of the efficiency of machine learning algorithms for phishing detection from URL’

In this system, machine learning algorithms are trained by twenty-four thousand data in a dataset and we test those learned model by more than ten thousand data. Our four algorithms named as decision tree, random forest, SVM, KNN have trained and tested on our dataset.

The working process of this system starting from dataset is taken as input then we plot and analyze the features of that dataset. Then we select the most effective features and create a new dataset. We find the unique value of each feature of that dataset and remove negative values. Then we split that dataset as train and test dataset. We train model for four classification algorithms and test their performance. As decision tree and random forest give us the highest accuracy we use deep neural decision forest, is applied for the testing and training those data. After that module has been created which take raw URL as input, it extracts data from that URL it have fed that extracted data to trained model of deep neural decision forest and give us output as “Phishing” or “legitimate”. In this module, three types of extraction have been done as our trained and test dataset. First one is third-part based features extraction, second one is URL based features annd third one is domain based features. This is illustrated as Figure 2.

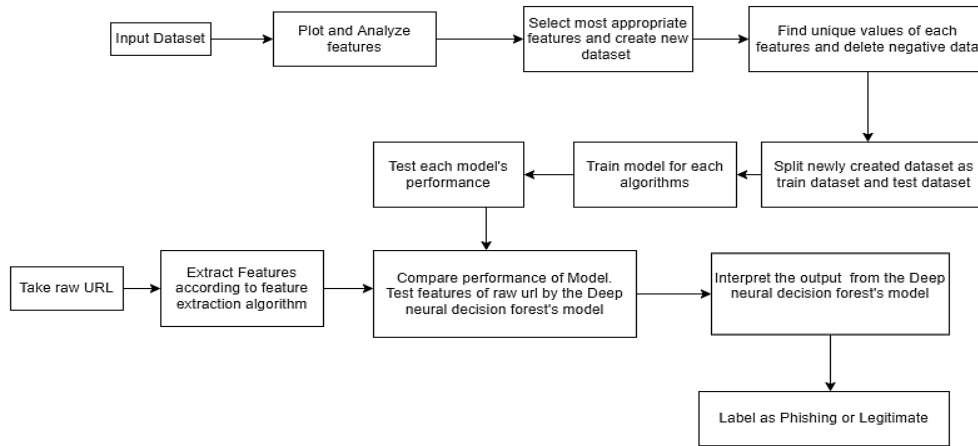


Figure 2. The working procedure of the system

4. RESULTS AND DISCUSSION

“Comparison of the efficiency of machine learning algorithms on phishing detection from URL” is a supervised machine learning classification type problem. Four popular machine learning algorithms and one deep learning algorithm have been applied into our trained dataset, they create models then our test dataset has been fed to those trained model and make the prediction. decision tree and random forest give us the higher accuracy than KNN and support vector machine. That’s mean, tree-structured algorithms are working better with our datasets. The performance of four algorithms is such that KNN as 75.10%, SVM as 90.64%, decision tree as 93.34% and Random Forest as 95.67%. This is shown in Table 1.

According to their performance in terms of how accurate the algorithms are predicting; we have expanded our work and we use deep neural decision forest, which is given 92.67% accurate prediction. The confusion matrix for these four algorithms is illustrated in Figure 3. After trained the model of deep neural decision forest, the Feature_Extraction_and_result () function takes the URL as input, extract the feature and the trained model is fed that features or data.

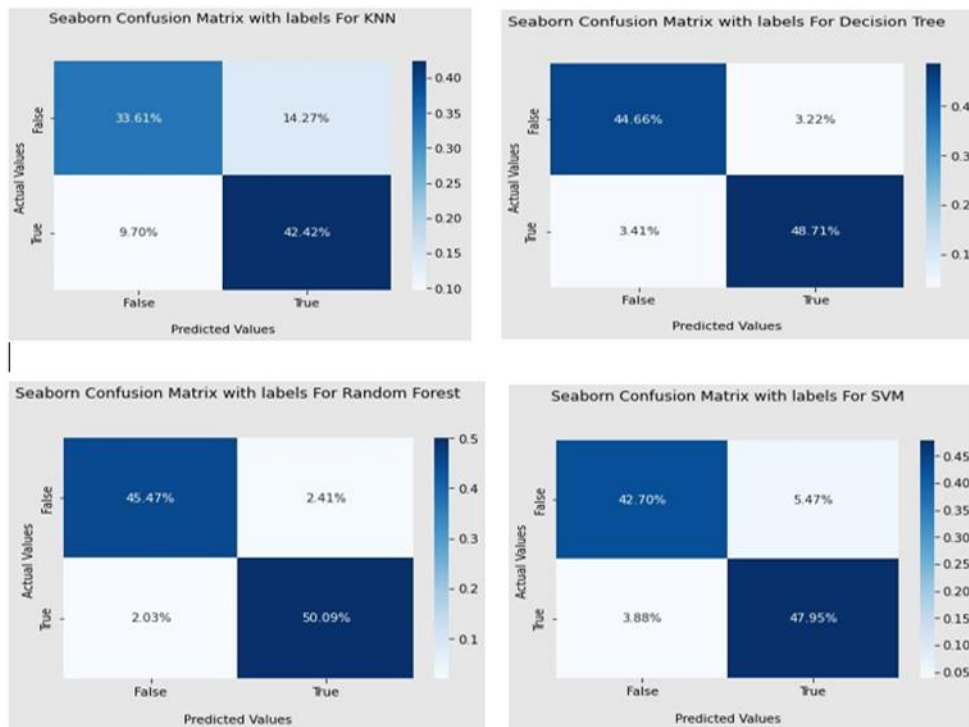


Figure 3. Confusion matrix

The extracted data from the URL as a list, the tensor that has been made to feed the Deep Neural Decision Forest model, the third-party features result one by one and at last the label as “Legitimate” illustrated in the Figure 4 for the first URL and “Phishing” illustrated in the Figure 5 for the second URL which is correct result for the given URL.

After analyzing result of each algorithm, it is proved that Decision Tree and Random Forest works better than other two algorithms and tree-structured algorithm is working better with our dataset and our selected features. After expanding the possibility, we have used Deep Neural Decision Forest which has given us the accuracy as 92.67% in prediction with our test-dataset output label.

```

features_Extraction_and_result("https://colab.research.google.com/drive/1GV09C_4ev9FRQ8htP_fdpLM9JPSNOjgs#scrollTo=c8Tp5EpaAIxD")
colab.research.google.com
1
300
4
(['173.194.216.113', '173.194.216.139', '173.194.216.138', '173.194.216.102', '173.194.216.100', '173.194.216.101'],)
6
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:19: DeprecationWarning: please use dns.resolver.resolve() instead
8895 2426
1997-09-15 2028-09-13
2022-01-22
8895 2426
15169
[[1, 300, 4, 6, 8895, 2426, '15169', 3, 0, 2, 4, 0, 1, 0, 0, 0, 0, 0, 95, 3, 0, 25]]
1
[1]
[<tf.Tensor: shape=(1,), dtype=float32, numpy=array([1.], dtype=float32)>, <tf.Tensor: shape=(1,), dtype=float32, numpy=array([300.], dtype=float32)>, <tf.Tensor:
[[0.0, 1.0]]
Legitimate
    
```

Figure 4. Output of the system labeled as 'Legitimate' for Legitimate URL

```

features_Extraction_and_result('http://bid.openx.net/json?amp;cid=oxpv1:1-1-1-2&f=0.1&pid=6816e307-0a4f-774f-9a04-0656e
bid.openx.net
0
900
4
(['35.244.159.8', '34.98.64.218'],)
2
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:19: DeprecationWarning: please use dns.resolver.resolve() instead
0 0
15169
[[0, 900, 4, 2, 0, 0, '15169', 8, 9, 1, 9, 1, 6, 0, 5, 0, 0, 0, 292, 2, 0, 13]]
1
[1]
[<tf.Tensor: shape=(1,), dtype=float32, numpy=array([0.], dtype=float32)>, <tf.Tensor: shape=(1,), dtype=float32, numpy=array([900.], dtype=float32)>, <tf.Tensor:
[[1.0, 0.0]]
Phishing
    
```

Figure 5. Output of this system labeled as "Phishing" for Phishing URL

Table 1. Algorithm’s performance

Serial No	Algorithm’s Name	Accuracy
1	K-Nearest Neighbor (KNN)	75.10%
2	Support Vector Machine (SVM)	90.64%
3	Decision Tree	93.34%
4	Random Forest	95.67%

5. CONCLUSION AND FUTURE WORK





We are introducing a comparative and effective phishing detection from URL by Machine Learning system which is a system for detecting phishing URL by observing URL structure. By this system, phishing websites URL can automatically be detected and say whether it is involves to phishing attacks or not. In this system, we have used twenty-three features from URL structure to detect the phishing site’s URL. We are using, in total, five algorithms for detecting. We use first four algorithm as decision tree, K-nearest neighbor (KNN), support vector machine (SVM) and Random Forest and compare their performance. Among these

four, random forest gives best performance. Then, we apply deep learning algorithm which is deep neural decision forest. After that, a system has been created that take a URL as input, extracted features from that URL and then that extracted data has been fed to the model of deep neural decision forest and it give us the output as phishing or legitimate label. This system works well. Machine learning based model is developing the better security against cybercrime like phishing attack. Now we will work in the future on the system that we are working on for improving the working procedure and feature extraction part. We will apply other deep learning algorithm like wide, deep and cross network, gated residual and variable selection network and compare their accuracy and performance.





REFERENCES

- [1] A. N. Shaikh, A. M. Shabut, and M. A. Hossain, "A literature review on phishing crime, prevention review and investigation of gaps," in *2016 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA)*, 2016, pp. 9–15, doi: 10.1109/SKIMA.2016.7916190.
- [2] D. Goel and A. K. Jain, "Mobile phishing attacks and defence mechanisms: State of art and open research challenges," *Computers & Security*, vol. 73, pp. 519–544, Mar. 2018, doi: 10.1016/j.cose.2017.12.006.
- [3] "Unifying the Global Response To Cybercrime" Accessed: Jan. 27, 2022. [Online]. Available: https://docs.apwg.org/reports/apwg_trends_report_q2_2021.pdf
- [4] M. Volkamer, K. Renaud, B. Reinheimer, and A. Kunz, "User experiences of TORPEDO: TOoltip-poweRed Phishing Email DetectiOn," *Computers & Security*, vol. 71, pp. 100–113, Nov. 2017, doi: 10.1016/j.cose.2017.02.004.
- [5] K. Greene, M. Steves, and M. Theofanos, "No phishing beyond this point," *Computer*, vol. 51, no. 6, pp. 86–89, Jun. 2018, doi: 10.1109/MC.2018.2701632.
- [6] B. B. Gupta, N. A. G. Arachchilage, and K. E. Psannis, "Defending against phishing attacks: taxonomy of methods, current issues and future directions," *Telecommunication Systems*, vol. 67, no. 2, pp. 247–267, Feb. 2018, doi: 10.1007/s11235-017-0334-z.
- [7] J. Hong, "The state of phishing attacks," *Communications of the ACM*, vol. 55, p. 74, 2012.
- [8] S. R. Curtis, P. Rajivan, D. N. Jones, and C. Gonzalez, "Phishing attempts among the dark triad: Patterns of attack and vulnerability," *Computers in Human Behavior*, vol. 87, pp. 174–182, Oct. 2018, doi: 10.1016/j.chb.2018.05.037.
- [9] Y. Cao, W. Han, and Y. Le, "Anti-phishing based on automated individual white-list," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 51–59, 2008, doi: 10.1145/1456424.1456434.
- [10] A. Y. Fu, L. Wenying, and X. Deng, "Detecting phishing web pages with visual similarity assessment based on earth mover's distance (EMD)," *IEEE Transactions on Dependable and Secure Computing*, vol. 3, no. 4, pp. 301–311, Oct. 2006, doi: 10.1109/TDSC.2006.50.
- [11] F. Toolan and J. Carthy, "Phishing detection using classifier ensembles," in *2009 eCrime Researchers Summit*, Oct. 2009, pp. 1–9, doi: 10.1109/ECRIME.2009.5342607.
- [12] D. L. Cook, V. K. Gurbani, and M. Daniluk, "Phishwish: A stateless phishing filter using minimal rules," in *Financial Cryptography and Data Security*, vol. 5143 LNCS, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 182–186.
- [13] H. Shahriar and M. Zulkernine, "Trustworthiness testing of phishing websites: A behavior model-based approach," *Future Generation Computer Systems*, vol. 28, no. 8, pp. 1258–1271, Oct. 2012, doi: 10.1016/j.future.2011.02.001.
- [14] R. Srinivasa Rao and A. R. Pais, "Detecting phishing websites using automation of human behavior," in *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*, Apr. 2017, pp. 33–42, doi: 10.1145/3055186.3055188.
- [15] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit on - eCrime '07*, 2007, vol. 269, pp. 60–69, doi: 10.1145/1299015.1299021.
- [16] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: A content-based approach to detecting phishing web sites," in *16th International World Wide Web Conference, WWW2007*, 2007, pp. 639–648, doi: 10.1145/1242572.1242659.
- [17] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, "CANTINA+: A feature-rich machine learning framework for detecting phishing web sites," *ACM Transactions on Information and System Security*, vol. 14, no. 2, pp. 1–28, Sep. 2011, doi: 10.1145/2019599.2019606.
- [18] A. Le, A. Markopoulou, and M. Faloutsos, "PhishDef: URL names say it all," in *2011 Proceedings IEEE INFOCOM*, Apr. 2011, pp. 191–195, doi: 10.1109/INFCOM.2011.5934995.
- [19] S. C. Jeeva and E. B. Rajsingh, "Intelligent phishing url detection using association rule mining," *Human-centric Computing and Information Sciences*, vol. 6, no. 1, p. 10, Dec. 2016, doi: 10.1186/s13673-016-0064-3.
- [20] A. K. Jain and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," *Telecommunication Systems*, vol. 68, no. 4, pp. 687–700, Aug. 2018, doi: 10.1007/s11235-017-0414-0.
- [21] K. M. Sundaram, R. Sasikumar, A. S. Meghana, A. Anuja, and C. Praneetha, "Detecting phishing websites using an efficient feature-based machine learning framework," *Revista Gestão Inovação e Tecnologias*, vol. 11, no. 2, pp. 2106–2112, Jun. 2021, doi: 10.47059/revistageintec.v11i2.1832.
- [22] G. Vrbancić, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," *Data in Brief*, vol. 33, p. 106438, Dec. 2020, doi: 10.1016/j.dib.2020.106438.
- [23] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy K-nearest neighbor algorithm," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 4, pp. 580–585, Jul. 1985, doi: 10.1109/TSMC.1985.6313426.
- [24] A. Navada, A. N. Ansari, S. Patil, and B. A. Sonkamble, "Overview of use of decision tree algorithms in machine learning," in *2011 IEEE Control and System Graduate Research Colloquium*, Jun. 2011, pp. 37–42, doi: 10.1109/ICSGRC.2011.5991826.
- [25] Y. Zhang, "Support vector machine classification algorithm and its application," in *Communications in Computer and Information Science*, vol. 308 CCIS, no. PART 2, 2012, pp. 179–186.
- [26] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
- [27] P. Kotschieder, M. Fiterau, A. Criminisi, and S. R. Buló, "Deep neural decision forests," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, vol. 2015 Inter, pp. 1467–1475, doi: 10.1109/ICCV.2015.172.





BIOGRAPHIES OF AUTHORS

Ahana Nandi Tultul     is student at Department of Computer Science and Engineering, College of Engineering and Technology (CEAT), International University of Business Agriculture and Technology (IUBAT), Dhaka, Bangladesh. Her research area is Machine Learning. She can be contacted at email: ahananondi6@gmail.com.



Romana Afroz     is student at Department of Computer Science and Engineering, College of Engineering and Technology (CEAT), International University of Business Agriculture and Technology (IUBAT), Dhaka, Bangladesh. She currently doing internship in software development with MERN Stack, HTML, CSS, Bootstrap, Reactjs, Node.js, JavaScript. Her research area is Machine Learning. She can be contact through email: 18203042@iubat.edu.



Md. Alomgir Hossain     is working as an Assistant Professor in the department of Computer Science & Engineering at IUBAT-International University of Business, Agriculture & Technology. He has completed Master's Degree in Computer Science from Jahangirnagar University His areas of interest are: Cloud Analysis, IoT, Machine Learning and Data mining. He can be contacted at email: alomgir.hossain@iubat.edu.