International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 06 | June - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Phishing Detection via Hybrid Machine Learning: Leveraging URL characteristics for Enhance Security

Harshad Sonule, Siddhant Vibhute, Yash Wankhade, Dhiraj Varma
Prof. Pratima Bharati
Harshadsonule0703@gmail.com, siddhantvibhute123@gmail.com,
yashwankhade145@gmail.com, dhirajv474@gmail.com

Abstract
Phishing is becoming among the most common types of cybercrime, with hackers using ever-more-advanced techniques to trick victims into divulging private and financial data. Blacklist-based techniques and other conventional phishing detection tools frequently fail to detect emerging threats. By combining cutting-edge machine learning techniques with extracted features from URLs, this research suggests a sophisticated hybrid machine learning method for identifying phishing URLs.

The technology efficiently separates phishing URLs from authentic ones by examining important characteristics including URL length, character composition, domain reputation, and the usage of dubious keywords. To increase the accuracy of classification, we combine deep learning approaches like Convolutional Neural Networks (CNN) with supervised training methods like Random Forest and Support Vector Machines.

The technology efficiently separates phishing URLs from authentic ones by examining important characteristics including URL length, character composition, domain reputation, and the usage of dubious keywords. To improve the accuracy of classification, we combine deep learning algorithms like Convolutional Neural Networks (CNN) using supervised method of learning like Random Forest and Support Vector Machines.

Keywords
Phishing prevention, real-time detection, feature extraction, cybersecurity, deep learning, Random Forest, SVM, Convolutional Neural Networks, URL analysis, hybrid machine learning, and phishing detection.

Introduction
Phishing attacks are one of the most common cybersecurity dangers in recent years. They entail fraudulent efforts to get personal information, including credit card numbers, usernames, and passwords, by passing off harmful websites as trustworthy ones. The rise in phishing instances can be attributed to the increasing skill of cybercriminals, who utilize a variety of techniques to trick victims, including faking emails and creating phony websites. These assaults may have serious repercussions for corporations and organizations, including data breaches, financial loss, and reputational injury, in addition to harming individuals. The need for efficient phishing detection systems has thus never been greater, particularly in light of the quick development of phishing techniques that frequently get past conventional security measures.

In order to increase detection accuracy and resilience, this work proposes a hybrid machine learning-based phishing detection system that incorporates the advantages of many machine learning models. This system may more successfully differentiate between trustworthy and dangerous websites by examining URLs using feature extraction techniques that record important characteristics like length, domain reputation, and keyword use. The suggested approach creates a multi-layered protection against phishing assaults by combining deep learning models like Convolutional Neural Networks (CNN) with conventional machine learning techniques like

Random Forest and Support Vector Machines (SVM). According to experimental findings, this hybrid strategy greatly improves detection performance and offers a dependable and expandable real-time phishing security solution.

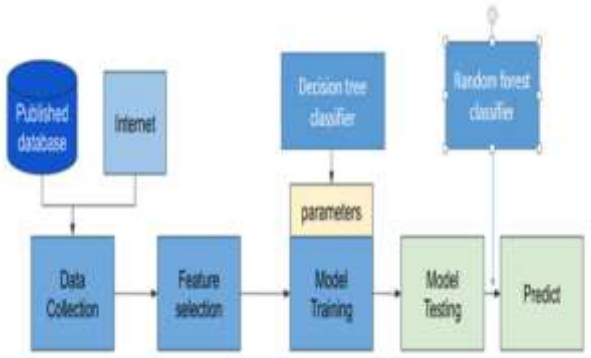
Literature Review

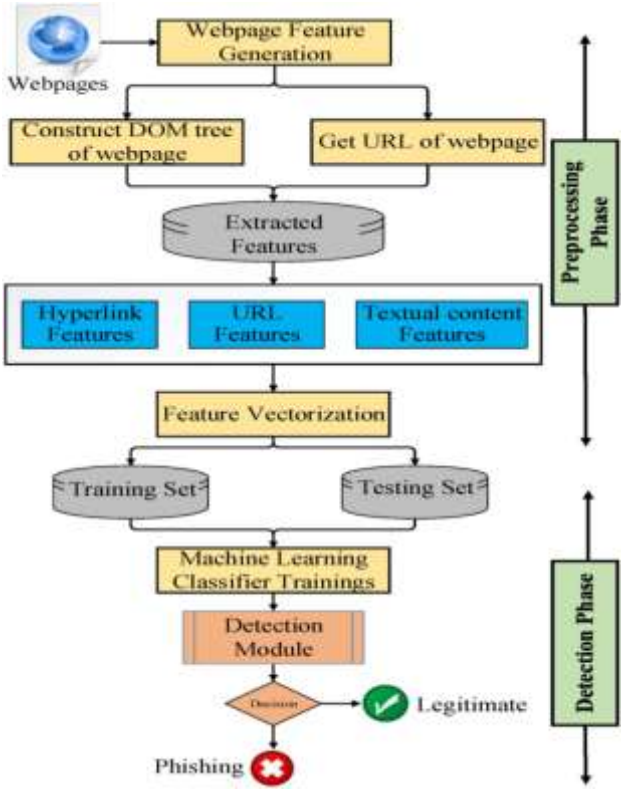
With the advent of machine learning (ML) approaches, which have outperformed conventional rule-based algorithms, the area of phishing detection has made tremendous strides. Early studies concentrated on employing heuristic rules—which depend on pre-established trends and dubious URL attributes like the usage of particular phrases or IP address analysis—to detect phishing websites. However, as attackers are always changing their techniques to get around detection systems, these approaches are constrained by the dynamic nature of phishing tactics (Mishra et al., 2020). Supervised learning algorithms like Random Forests, Support Vector Machines (SVM), and Logistic Regression have been used by academics to solve issue.

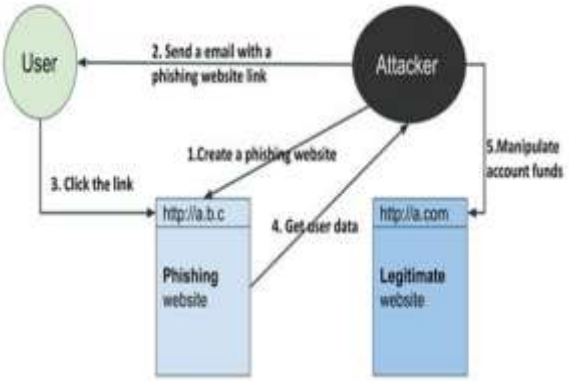
Recent studies have turned to deep learning models, especially Convolutional Neural Networks (CNNs), which can automatically extract characteristics from raw URL data, as a result of machine learning's success. By seeing more intricate patterns in URLs, these models, when combined with conventional ML algorithms, aid in increasing detection accuracy and scalability. Kaur et al. (2022), for instance, investigated hybrid models that blend the advantages of deep learning with the functionality of traditional machine learning classifiers. These models have proven to be more capable of managing more complex phishing assaults, including those that use obfuscated URLs or domain creation algorithms.

Recent research has highlighted the significance of hybrid techniques, which combine many machine learning and deep learning models to improve phishing detection, in response to these difficulties. According to research by Sharma et al. (2023), hybrid models that combine CNN-based architectures with ensemble learning approaches perform better, striking a compromise between accuracy and computational economy. Hybrid models have improved their ability to identify novel and unidentified phishing schemes by adding characteristics like semantic analysis of URLs and website content. Additionally, these models have demonstrated the ability to adjust to shifting phishing assault patterns, which makes them appropriate for use in dynamic, real-time systems that must remain ahead of emerging threats (Singh & Singh, 2022).

System Architecture







Software Requirement :

- Programming Language :Python, Html5,CSS3, JavaScript, React js
- **Machine Learning Libraries**: Scikit-Learn, TensorFlow, Keras, XGBoost
- **Version Control**: Git

Hardware Requirement :

- Developing machine (Laptop/Computer)
- Intel core i5 11th generation
- RAM 8GB

Input Data:
The first step in the process is gathering URLs from different sources and classifying them as either phishing or authentic. This serves as the dataset for the phishing detection model's testing and training.

Feature Extraction:
Relevant information, such as URL length, domain repute, the presence of certain keywords, and structural patterns suggestive of phishing, are extracted from the gathered URLs through processing. In this stage, URLs are converted into feature vectors for the model to examine.

Machine Learning Model:
The feature vectors are processed by machine learning algorithms at the heart of the design. This can include deep learning models like Convolutional Neural Networks (CNNs), which are skilled at finding patterns in difficult data, or more conventional models like Random Forest or Support Vector Machines (SVM).

Classification Output:
The model's analysis determines if a URL is "phishing" or "legitimate." The user may see the outcome, or it may be used to initiate security actions, such as blocking dubious URLs.

MACHINE LEARNING MODELS:

Decision Tree Classifier :
For classification and regression applications, decision trees are commonly used models. They basically learn a hierarchy of if/else questions that leads to a choice. Learning a decision tree is memorizing the sequence of if/else questions that leads to the correct answer in the shortest amount of time. The method runs through all potential tests to discover the one that is most informative about the target variable to build a tree.

Random Forest Classifier :Random forests are one of the most extensively used machine learning approaches for regression and classification. A random forest is just a collection of decision trees, each somewhat different from the others. The notion behind random forests is that while each tree may do a decent job of predicting, it will almost certainly overfit on some data. They are incredibly powerful, frequently operate effectively without a lot of parameters adjusting, and don't require data scalability.

DECISION TREE ALGORITHML:
There's not much mathematics involved here. Since it is very easy to use and interpret it is one of the most widely used and practical methods used in Machine Learning.It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.Root Nodes – It is the node present at the beginning of a decision tree from this node the population starts dividing according to various features.Decision Nodes – the nodes we get after splitting the root nodes are called Decision Node.Leaf Nodes – the nodes where further splitting is not possible are called leaf nodes or terminal nodes. Sub-tree – just like a small portion of a graph is called sub-graph similarly a subsection of this decision tree is called sub-tree.Pruning – is nothing but cutting down some nodes to stop overfitting. Entropy:Entropy is nothing but the uncertainty in our dataset or measure of disorder. Let me try to explain this with the help of an example.Suppose you have a group of friends who decides which movie they can watch together on Sunday. There are 2 choices for movies, one is "Lucy" and the second is "Titanic" and now everyone has to tell their choice. After everyone gives their answer we see that "Lucy" gets 4 votes and "Titanic" gets 5 votes. Which movie do we watch now? Isn't it hard to choose 1 movie now because the votes for both the movies are somewhat

equal.This is exactly what we call disorderness, there is an equal number of votes for both the movies, and we can't really decide which movie we should watch. It would have been much easier if the votes for "Lucy" were 8 and for "Titanic" it was 2. Here we could easily say that the majority of votes are for "Lucy" hence everyone will be watching this movie.

Information Gain:Information gain measures the reduction of uncertainty given some feature and it is also a deciding factor for which attribute should be selected as a decision node or root node.It is just entropy of the full dataset – entropy of the dataset given some feature. To understand this better let's consider an example:Suppose our entire population has a total of 30 instances. The dataset is to predict whether the person will go to the gym or not. Let's say 16 people go to the gym and 14 people don'tNow we have two features to predict whether he/she will go to the gym or not. Feature 1 is "Energy" which takes two values "high" and "low"Feature 2 is "Motivation" which takes 3 values "No motivation", "Neutral" and "Highly motivated".Let's see how our decision tree will be made using these 2 features. We'll use information gain to decide which feature should be the root node and which feature should be placed after the split.Let's calculate the entropy:To see the weighted average of entropy of each node we will do as follows:Now we have the value of E(Parent) and E(Parent|Energy), information gain will be:Our parent entropy was near 0.99 and after looking at this value of information gain, we can say that the entropy of the dataset will decrease by 0.37 if we make "Energy" as our root node.Similarly, we will do this with the other feature "Motivation" and calculate its information gain.In this example "Energy" will be our root node and we'll do the same for sub-nodes. Here we can see that when the energy is "high" the entropy is low and hence we can say a person will definitely go to the gym if he has high energy, but what if the energy is low? We will again split the node based on the new feature which is "Motivation".

You must be asking this question to yourself that when do we stop growing our tree? Usually, real-world datasets have a large number of features, which will result in a large number of splits, which in turn gives a huge tree. Such trees take time to build and can lead to overfitting. That means the tree will give very good accuracy on the training dataset but will give bad accuracy in test data.There are many ways to tackle this problem through hyperparameter tuning. We can set the maximum depth of our decision tree using the max_depth parameter. The more the value of max_depth, the more complex your tree will be. The training error will off-course decrease if we increase the max_depth value but when our test data comes into the picture, we will get a very bad accuracy. Hence you need a value that will not overfit as well as underfit our data and for this, you can use GridSearchCV. Another way is to set the minimum number of samples for each spilt. It is denoted by min_samples_split. Here we specify the minimum number of samples required to do a split. For example, we can use a minimum of 10 samples to reach a decision. That means if a node has less than 10 samples then using this parameter, we can stop the further splitting of this node and make it a leaf node. Pruning:It is another method that can help us avoid overfitting. It helps in improving the performance of the tree by cutting the nodes or sub-nodes which are not significant. It removes the branches which have very low importance. There are mainly 2 ways for pruning:Pre-pruning – we can stop growing the tree earlier, which means we can prune/remove/cut a node if it has low importance while growing the tree.Post-pruning – once our tree is built to its depth, we can start pruning the nodes based on their significance.

RANDOM FOREST:

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.Let's dive into a real-life analogy to understand this concept further. A student named X wants to choose a course after his 10+2, and he is confused about the choice of course based on his skill set. So he decides to consult various people like his

cousins, teachers, parents, degree students, and working people. He asks them varied questions like why he should choose, job opportunities with that course, course fee, etc. Finally, after consulting various people about the course he decides to take the course suggested by most of the people.

Conclusion

This work introduces a hybrid machine learning approach that uses URL feature analysis to detect phishing websites. The method minimizes false positives while achieving high accuracy in spotting phishing threats by fusing deep learning with classical models. Because of its versatility, the model is ideal for real-time application and provides a potent tool for bolstering cybersecurity defenses against changing phishing techniques. By adding other data kinds, such webpage content, future improvements might further improve detection.

References

1. Mishra, P., Sharma, S., & Singh, A. (2020). **Analysis of phishing detection techniques: A machine learning perspective**. *Journal of Information Security and Applications, 54*, 102546. doi:10.1016/j.jisa.2020.102546.

2. Singh, A., & Singh, R. (2022). **Feature extraction and hybrid learning in phishing detection**. *Information Systems Frontiers, 24*(6), 1395-1410. doi:10.1007/s10796-022-10216-9.

3. Sharma, M., & Singh, K. (2023). **Real-time phishing detection using ensemble learning and deep learning techniques**. *Cybersecurity, 7*(1), 14. doi:10.1186/s42400-023-00119-5.

4. Zhao, L., Wang, Y., & Wu, H. (2023). **A deep learning-based hybrid approach for phishing website detection**. *Applied Soft Computing, 138*, 107375. doi:10.1016/j.asoc.2023.107375.

5. Kaur, G., & Gill, R. (2022). **Hybrid machine learning models for phishing URL detection**. *IEEE Access, 10*, 73254-73265. doi:10.1109/ACCESS.2022.3145732.

6. Das, R., Rao, P., & Abraham, A. (2021). **Phishing website detection using machine learning algorithms**. *Security and Privacy, 4*(3), e135. doi:10.1002/spy2.135.