National College of Ireland

# Enhancing Phishing URL Detection by Leveraging Machine Learning and Deep Learning Models

MSc Research Project
MSc.in.Cyber Security


Charan Deep Chinthalapalli
Student ID: x23231866

School of Computing
National College of Ireland


Supervisor: Michael Pantridge

**National College of Ireland**
**Project Submission Sheet**
**School of Computing**

| | |
|---|---|
| **Student Name:** | Charan Deep Chinthalapalli |
| **Student ID:** | x23231866 |
| **Programme:** | MSc.in.Cyber Security |
| **Year:** | 2024-2025 |
| **Module:** | MSc Research Project |
| **Supervisor:** | Michael Pantridge |
| **Submission Due Date:** | 12/12/2024 |
| **Project Title:** | Enhancing Phishing URL Detection by Leveraging Machine Learning and Deep Learning Models |
| **Word Count:** | 7000 |
| **Page Count:** | 19 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

**ALL** internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

| | |
|---|---|
| **Signature:** | Charan Deep Chinthalapalli |
| **Date:** | 29th January 2025 |

## PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies). | ☐ |
| **Attach a Moodle submission receipt of the online project submission**, to each project (including multiple copies). | ☐ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | ☐ |

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Enhancing Phishing URL Detection by Leveraging Machine Learning and Deep Learning Models

Charan Deep Chinthalapalli
x23231866
x23231866@student.ncirl.ie
National College of Ireland

**Abstract**

Phishing is one of the most popular types of cybercrime; it deceives the users to surrender some personal information through the help of fake sites that mimic the real ones. Phishing URLs are difficult to detect and are currently one of the biggest issues because of the growing complexity of these attacks and the inefficiency of the measures. This paper aims to compare the efficiency of multiple machine learning and deep learning techniques for the detection of phishing URLs with an emphasis on the impact of feature engineering. The work compares several machine learning models such as Random Forest, AdaBoost, Logistic Regression, LSTM, and TabNet using a labeled set of URLs comprising benign and phishing URLs. To enhance the classification performance, a few properties like the URL length, special characters in the URL, the use of the HTTPS protocol, and several subdomains are extracted. The research evaluates model performance based on the evaluation parameters including accuracy, precision, recall, and F1-score, and addresses issues like class imbalance and dataset complexity. The results reveal that the model with the highest accuracy was XGBoost with 88.93%, while deep learning models such as LSTM and TabNet were slightly lower. However, Random Forest and XGBoost enhance the performance in detecting phishing URLs, and the traditional machine learning methodologies are sufficient for detecting phishing URLs.

## 1 Introduction

Phishing attacks become one of the most popular and dangerous types of threats in the digital world. Phishing attacks are designed to trick people into providing information such as passwords, financial information, and identification information. Such attacks are normally conducted with fake websites that are very similar to the real sites, while the URLs look rather trustworthy. Many of the obtained phishing URLs contain several manipulations, including homographic attacks, domain imitating, the use of shortened links, and so on, making it significantly more challenging to identify such links for a typical user. These attacks start to become more complex, and simple solutions such as blacklisting or using heuristic rules fail most of the time. Consequently, the development of fully automated, effective, and large-scale anti-phishing solutions that can identify phishing URLs and prevent user exposure to these threats is required.

## 1.1 Motivation

Phishing attacks in particular have caused a surge in activity, and are enhanced by the fact that attackers continue to develop more advanced methods of infiltrating these security systems. Most recent methods utilize blacklists or basic string searching and they are inadequate in the identification of new or emerging phishing sites. The current approaches have some limitations, which can be overcome by using machine learning and deep learning to identify the patterns in URLs and detect the differences between legitimate and phishing URLs. But issues like class imbalance (where several legitimate URLs far exceed the number of phishing URLs) and the fact that there are diverse types of methods used in phishing, make it impossible to get a high level of accuracy and at the same time very low false positive rates. Therefore, additional techniques to counter the new trends in phishing strategies are required. As such, there is an opportunity to enhance the identification of phishing attacks using feature-extracted machine learning and deep learning models in categorizing URLs.

## 1.2 Objectives

The primary goal of this research study is to assess and compare different machine learning and deep learning algorithms for the classification of phishing URLs. This means creating a system that can categorize URLs into safe or phishing based on features derived from the URLs. The study aims to identify which models perform best for this task and assess the importance of feature engineering in improving classification accuracy. By comparing machine learning models such as Random Forest, AdaBoost, and Logistic Regression with deep learning models like LSTM and TabNet, the research will provide insights into the effectiveness of each approach for phishing URL detection.

## 1.3 Research Question

How can machine learning and deep learning models effectively detect phishing URLs, and what role does feature engineering play in improving their performance?

In response to this research question, the study will assess the effects of feature engineering on the performance of both machine learning and deep learning models in identifying phishing URLs. First, the database of benign and phishing URLs will be constructed and ready for feature selection. Such features include an indicator of URL length, character set, usage of the HTTPS protocol, and several subdomains, which have been proven to provide good results for phishing detection. A comparison of different machine learning algorithms such as Random Forest, AdaBoost, and Logistic Regression, will be done based on the accuracy of the classification of URLs. Thus, further evaluation of the models, LSTM and TabNet, will be performed to establish if they have an edge in identifying more intricate patterns in the data set. This research will consider how feature selection, hyperparameter optimization, and model architecture affect the capacity of these models to differentiate between 'safe' and 'phishing' URLs.

## 1.4 Scope of the Study

This research aims to identify phishing URLs using a dataset consisting of benign and phishing URLs. Different machine learning and deep learning models have to be implemented during the study and feature engineering has to be used to enhance the model's

performance. The results will be assessed with accuracy, precision, recall, and F1-score, the identification of phishing URLs, and at the same time the efficiency of the model on a huge amount of data. Also, this study aims to establish the effects of hyperparameters tuning and model selection towards performance. Finally, this research aims to make the following contributions towards strengthening the identified aspects to enhance the effectiveness of the phishing URL detection system.

# 2 Literature Review

## 2.1 Introduction

Phishing attacks are a large threat to cyber security as they use various tricks to get users private information (Do et al.; 2022; Ozcan et al.; 2023). The existing technologies that are used for detection, like blacklists and heuristic-based systems are inadequate to address the current and even more complex and aggressive kind of phishing attacks. As a result of this, research into ML and DL techniques for detecting phishing has gained prominence because such techniques are capable of improving the accuracy of detection as more data is processed (Divakaran and Oest; 2022; Tamal et al.; 2024; Sahoo et al.; 2017; Balantrapu; 2023). This review presents a comprehensive analysis of the ML (machine learning) and DL (deep learning) based methods with an indication of the opportunity and the challenges that come with it.

## 2.2 Machine Learning-Based Approaches

Phishing detection is a well-suited problem for machine learning approaches because of the possibility of updating the model and the fact that the data is structured. Many papers have used traditional classifiers comprising of Random Forest, Logistic Regression, Gradient Boosting, and XGBoost among others. These models are specifically proficient in the use of the URL length, subdomain, and the use of special characters to distinguish between the two; phishing and benign URLs (Rani et al.; 2024; El-Metwaly et al.; 2024; Tamal et al.; 2024).

showing the versatility of ML algorithms in their survey Sahoo et al. (2017), and explaining the dynamic nature of phishing URLs as a problem domain Rani et al. (2024); Tamal et al. (2024). Moreover, El-Metwaly et al. (2024) showed that XGBoost shows a high level of accuracy in addressing the class imbalance, a problem typical for phishing datasets with the —nature of phishing URLs. Rani et al. (2024) outlines the significance of feature engineering in optimizing phishing detection, while Tamal et al. (2024) proposed an advanced feature vectorization algorithm to improve classification performance. Similarly, El-Metwaly et al. (2024) shows that XGBoost performs well in handling class imbalance, a common issue in phishing datasets.

Some recent works have proposed the use of more complex architectures to improve the capabilities of ML. Adebowale et al. (2023) examined ensemble techniques that combined ML algorithms with feature optimization to enhance the detection rates. This aligns with the study by Sahingoz et al. (2024) who put forward an ML-deep learning-based pipeline, that identifies dangerous URLs fast while incorporating context featurization strategies to enhance predicting accuracy. This aligns with the findings of Sahingoz et al. (2024), who proposed a deep-learning-enhanced ML pipeline that efficiently detects malicious URLs by integrating context-aware features.

## 2.3 Deep Learning-Based Approaches

The methods of deep learning are implemented because they learn the hierarchical representations and can distinguish subtle patterns within URLs.For instance, LSTM networks, in particular, have provided good results in identifying sequential relationships that exist in the components of URLs, which are usually related to phishing attempts (Remya et al.; 2024; Ullah et al.; 2024). demonstrated the ability of deep learning models to outperform classical models in identifying hidden patterns associated with phishing URLs.

Another category of work has investigated models that prioritize feature importance during training such as TabNet. In the same line, Trinh et al. (2022) proposed to use DL-based image classifiers for detecting phishing sites based on the visual similarity analysis. Building on this, Ujah-Ogbuagu et al. (2024) developed a hybrid deep learning system that integrates feature engineering into the DL framework for real-time detection.

However, limitations are present to applying the DL models for phishing identification For instance, Tamal et al. (2024) demonstrated that DL approaches need more data to outperform other methods. Tajaddodianfar et al. (2020) developed a character/word-level DL model for detecting phishing URLs, which has been reported to have better performance on smaller datasets. However, a few weaknesses are also notable; computational complexity and the non-interpretable nature of DL models are still issues in its applicability (Bauskar et al.; 2024; Sameen et al.; 2020; El-Metwaly et al.; 2024).

## 2.4 Relevance to Current Research

The reviewed literature shows an improvement in the implementation of Machine learning and Deep learning in Phishing URL detection. Though DL methods are capable of identifying complex feature interactions, approaches such as Random Forest and XGBoost are still remarkable on regular feature datasets. This is in line with research conducted by Sahingoz et al. Sahingoz et al. (2024) and Rani et al. Rani et al. (2024) which indicate that the real-world application of ML is very stable. ML models like Random Forest and XGBoost show competitive performance, especially for structured datasets Sahingoz et al. (2024) and Rani et al. (2024), which shows the robustness of ML techniques in real-world scenarios. This current research extends these ideas further by employing a more sophisticated feature extraction process and different types of models in ML and DL. The results highlight the need to choose the right models and fine-tune the hyperparameters that guarantee a high detection rate.

# 3 Methodology

This study employs a structured approach to identify phishing URLs to apply both ML (machine learning) and DL (deep learning). The steps followed in the study are data collection, feature extraction, model training, and model evaluation. The objective is to explore the effectiveness of various models in identifying malicious URLs This section seeks to achieve is to present the research process, method, instruments, approaches, and overall methodology of this study.

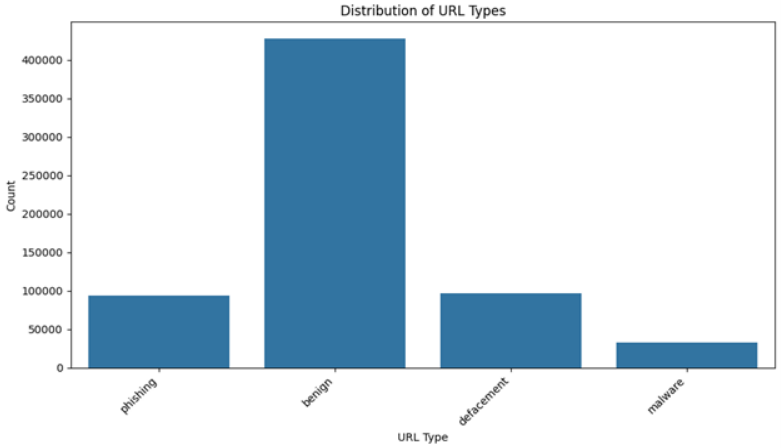



Figure 1: Various URL distributions in the dataset

## 3.1 Data Collection and Preprocessing

The dataset used in this study consists of 651,191 labeled URL entries, categorized into four types: defacement, benign, phishing, and malware. These were collected from the public domain and the sources included a broad range of web traffic data, both genuine and fake. The next step of data preprocessing was to clean the dataset to remove duplicate records and manage missing values. Several characteristics of the URLs were also derived after preprocessing: IP, subdomain, HTTPS usage, and URL length, among others. These features were selected because of their possible relation to the task of malicious URL identification.

## 3.2 Model Selection and Training

Different machine learning and deep learning algorithms are selected to analyze the performance of each model for detecting phishing URLs. The selected machine-learning models are AdaBoost, RandomForest, DecisionTree, Logistic Regression, Gradient Boosting, and XGBoost. These models were chosen because they are suitable for structured tabular data and they are widely used in classification problems. In addition, some deep learning models such as custom deep learning architecture, TabNet, LSTM, etc., were employed to learn dependencies in the data.

The models were trained with a supervised learning algorithm, the extracted features used as input, and the URL type: benign or malicious as the response variable. Data was divided into a training set with 80%, a validation set of 10% from training data, and a test set of 20%. This division made it possible to avoid bias in the model's evaluation. Optimizations such as hyperparameters optimization through grid search and cross-validation were also made in the subsequent steps.

## 3.3 Evaluation Methodology

To evaluate the models, four key metrics were used: accuracy, precision, recall, and F1 score. Accuracy was a general measure of the models performance while precision and recall were a vital measure of how well the models performed in correctly assigning the URL as phishing. Due to the nature of the data set which is highly unbalanced, recall was given priority so that all the phishing URLs would be captured. To assess the trade-off

between precision and recall, the F1-score is reported given that it is a harmonic mean of both metrics. confusion matrices indicate true positives, true negatives, false positives, and false negatives. These matrices further improved the understanding of the models' discriminative ability between benign and phishing URLs. The evaluation was made with the test set to analyze the generalization capability of every model. A comparison between the two approaches of machine learning and deep learning to find out which is more suitable for this classification problem.

In conclusion, the methodology used in this research offers a comprehensive and structured approach to identifying phishing URLs. The further application of both machine learning and deep learning models, and a thorough examination of each complemented by efforts to address such issues as class imbalance, make it possible to provide a comprehensive comparison of different approaches towards the detection of phishing URLs.

# 4 Implementation

## 4.1 About Dataset

The dataset for phishing URL detection comprises 651,191 entries, each labeled into one of four categories: benign, defacement, phishing, and malware. It contains 428103, benign URLs the most numerous class while the second class contains 96457 defacement URLs, the third class contains 94111 phishing URLs, and the last class contains 32520 malware URLs. This class distribution simply depicts the real-world scenario where benign URLs are more common than malicious types. The variability of the dataset can offer a background for analyzing the characteristics of various sorts of URLs, including the fine details of the malicious ones. It becomes useful for developing and evaluating machine learning models for classifying and identifying malicious activities in Web URLs.

## 4.2 Feature Engineering

New features were extracted from the URLs to enrich the dataset with attributes that will be useful in the classification of benign and malicious URLs. These features are as follows:

1. **use_of_ip**: There is a binary feature representing the presence of an IP address presence in the URL.
2. **use_of_at**: A binary feature that checks the presence of the "@" symbol which is always used in the phishing URLs.
3. **url_length**: The length of the malicious URL is usually longer.
4. **url_redirection**: A binary feature indicating the URL contains more than one redirection indicated by // after the protocol.
5. **prefix_suffix**: A binary feature that looks for hyphens ("-") in the domain, regularly employed by phishing scams.
6. **https_token**: A binary feature that is 1 if the URL contains HTTPS, otherwise 0.
7. **digits_present**: A binary feature that flags the presence of numeric characters in the URL.

8. **subdomain_present**: A binary feature that marks the URLs that have more than two dots which indicates subdomain.

9. **shortening_service**: URL expansion service detection feature that looks for such services as bit.ly or tinyurl.com.

10. **word_count**: The sum of all the words in the URL.

11. **char_count**: The sum of the number of characters within the URL string.

12. **abnormal_url**: A binary feature that marks URLs with words such as 'malicious' or 'phish'.

13. **count_http** : A binary features of counts the number of times "http" appears in the URL.

14. **hostname_length**: It is the number of characters in the hostname of the URL.

These engineered features are important to understand the framework and characteristics of the URLs to improve classification with machine learning models.

### 4.3 Data Transformation

To pre-process the data for the machine learning models, feature scaling was done using the Min-Max scaling method. The transformation process involved two key steps:

1. **Fitting the Scaler:** On the training data (`X_train`), the Min-Max scaler is used to retrieve the minimum and maximum values of each feature.

2. **Transforming the Data:** With the fitted scaler, the training data (`X_train`), validation data (`X_val`), and test data (`X_test`) were scaled to make sure that all of the subsets are scaled similarly.

This approach assists in preventing data leakage by deriving scaling parameters from the training set. Normalization helps increase the speed of the execution of the machine learning models and also helps in enhancing models precisions, especially for models that are known to be very sensitive to the range of features.

## 5 Machine Learning

### 5.1 AdaBoost Classifier

The AdaBoost Classifier was selected to build an ensemble of weak classifiers with the help of a weighted distribution that targets samples, which are difficult for classification. This kind of adaptive nature makes it good for imbalanced datasets where phishing and other related links are compared to harmless ones. Through the use of higher weights misclassified samples during each cycle, AdaBoost also performs efficiently and captures features that other simple classifiers could easily overlook. This makes it a strong solution to identify the malicious URLs.

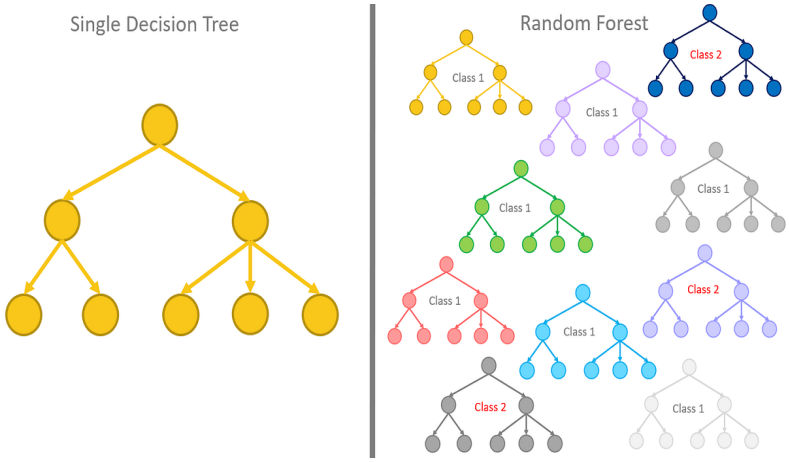



Figure 2: Architecture of Decision tree and Random forest

## 5.2 Random Forest Classifier

The Random Forest Classifier was selected because it is an ensemble learning method that constructs multiple decision trees and integrates the results of the trees to make predictions. This method eliminates overfitting that could occur with unique decision trees and enhances the models ability to generalize. The Random Forest algorithm is for large data sets with high-dimensional attributes, as in our research, where many URL-derived variables has to be examined. It can learn higher-order interactions between features and can therefore be a very effective model for classifying malicious and benign URLs.The Decision Tree was added due to its readability and compactness. and characteristics of the URLs, aiding in effective classification using machine learning models.

## 5.3 Decision Tree Classifier

The Decision Tree was included for its interpretability and simplicity. Decision trees each feature leads to a particular decision-making process which is essential. This model was used to provide a point of comparison to see how well the features in the dataset could classify URLs on their own. Its capability of dealing with non-linearity means it remains capable of learning patterns in benign and malicious URLs without much fine-tuning.

## 5.4 Logistic Regression

Logistic regression is known for its efficiency in binary class problems and offers probability predictions. Perhaps the speed and simplicity of this model make it suitable as a benchmark model. Logistic regression provides information about significance of each attribute by assessing the coefficients weights. This interpretability assists in elucidating the role of various URL-derived features including the length of a URLs, and the presence of sub domains when differentiating between safe and unsafe URLs.

## 5.5 Gradient Boosting Classifier

The Gradient Boosting Classifier was chosen for its ability to create powerful prediction methods through learning from errors. This model performs well in those problems where the data is split into either a majority or minority class, as this model aims at reducing

the loss function in each iteration. The decision tree technique in Gradient Boosting models features interactions and yields a highly accurate prediction that could qualify it for consideration in detecting phishing and malicious URLs. It can accommodate the specific needs of the dataset by parameters it uses.

### 5.6 XgBoost

The XGBoost model was selected because of its efficiency in handling high-dimension datasets and mitigating overfitting due to tuning parameters. XGBoost is an optimized version of the Gradient Boosting system that constructs well-performing classifiers from decision trees, each dedicated to minimizing the loss function. The self-improvement nature and the capability to conquer high-dimensional datasets and address intersectional non-linear relationships make XGBoost highly effective in identifying phishing URLs where the difference between safe and risky. Moreover, XGBoost has other features including column sub sample, and row sample, which enhances the model from overfitting and improves its generality. Such flexibility, tested characteristics, including accuracy, precision, recall, and F1 score, make it suitable for identifying phishing URLs.

These models were selected due to their interpretability, computational speed, and state-of-the-art performance. These approaches also assist in utilizing the benefits of each model in addressing the problems arising from the dataset and enhancing the general URL classification.

## 6 Deep Learning Models

To complement traditional machine learning approaches, three deep learning models were utilized: Custom Deep Learning Models, TabNet Classifier, and LSTM.

### 6.1 Custom Deep Learning Models

Custom deep-learning models were developed to utilize the interdependencies of the features obtained from the URLs. These models are made of several dense layers of high density with ReLU activation functions that allow the models to identify the non-linear transformation between features and labels. This is possible due to the flexibility that characterizes a custom architecture that can be trained to suit the nature of the dataset. And change in several layers, neurons, and non-linearity by activation functions the model can learn the finer details and dependencies in the dataset which are otherwise not easily recognized by most of the conventional machine learning algorithms. This flexibility makes the custom deep learning models a viable technique for dataset characterization and categorization.

### 6.2 TabNet Classifier

The TabNet classifier was selected due to its high interpretability and performance in the tabular data. TabNet uses an attention mechanism to decide what features to focus during training which is highly beneficial for the URL dataset and has many features. The ability of the model to learn with limited data means that areas of high importance can be enhanced while trivial data is eliminated. The architecture of TabNet is consistent with the tabular data settings and provides information about feature significance while

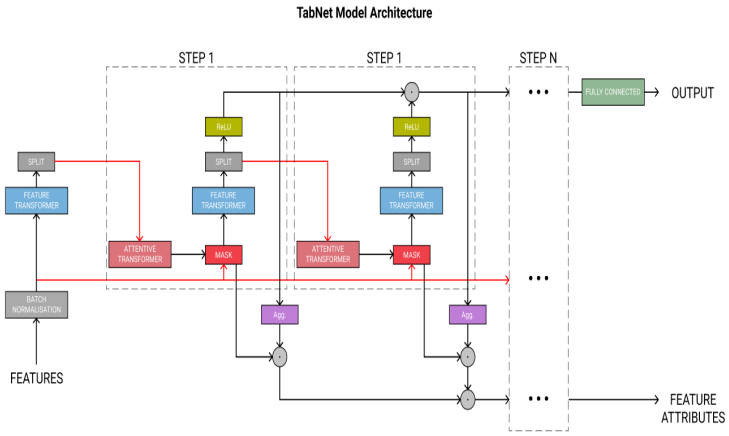



Figure 3: Architecture of TabNet classifier

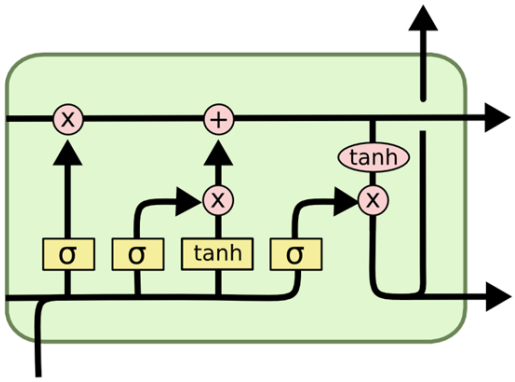



Figure 4: Architecture of LSTM model

being highly accurate. For this problem, the interpretability of TabNet and efficiency make it a good choice.

## 6.3 LSTM (Long Short-Term Memory)

LSTM was known to analyze sequential patterns in the dataset even though it is not a time series. URLs components can be ordered in some patterns, which are tokens, symbols, or subdomains, and LSTM is capable of learning. The LSTM model provides awareness of the dependencies and connections of the features that are generally hard to detect. This makes LSTM particularly useful for capturing the potentially sequential nature of URL structures which enhances the model's capability to distinguish between good and bad URLs.

These deep learning models complement conventional machine learning algorithms due to their ability to learn feature representations from the data. Due to their flexibility and high architecture, they discover underlying relations and profiles, which make them suitable for solving the URL classification issue.

| Machine Learning Models | | | | |
|---|---|---|---|---|
| **Model** | **Accuracy (%)** | **Precision** | **Recall** | **F1-Score** |
| AdaBoostClassifier | 87.80 | 0.89 | 0.84 | 0.86 |
| RandomForestClassifier | 88.87 | 0.90 | 0.85 | 0.87 |
| DecisionTreeClassifier | 88.84 | 0.90 | 0.85 | 0.87 |
| LogisticRegression | 86.13 | 0.87 | 0.82 | 0.84 |
| GradientBoostingClassifier | 88.32 | 0.90 | 0.84 | 0.86 |
| XGBClassifier | 88.93 | 0.90 | 0.85 | 0.87 |
| **Deep Learning Models** | | | | |
| **Model** | **Accuracy (%)** | **Precision** | **Recall** | **F1-Score** |
| Custom DL Model | 88.23 | 0.90 | 0.84 | 0.86 |
| TabNet Classifier | 88.00 | 0.89 | 0.84 | 0.86 |
| LSTM Model | 88.38 | 0.90 | 0.84 | 0.86 |

Table 1: Performance Comparison of Machine Learning and Deep Learning Models

# 7 Results and Discussion

## 7.1 AdaBoostClassifier

The AdaBoost Classifier had an accuracy of 87.80%, 0.89 of precision, and 0.84 of recall. This model performed efficiently to recognize benign URLs as suggested by the precision of 0.89. But the recall for phishing URLs was a bit lower at 0.84 which means the model did well in identifying benign URLs while missing some phishing URLs. This is always a problem in skewed datasets, where there are many more benign URLs than malicious ones. The F1-score of 0.86 shows a good deal of accuracy and recall, meaning that AdaBoost is relevant for detecting phishing URLs but could be improved to capture all phishing cases.

## 7.2 Random Forest Classifier

The accuracy of the Random Forest Classifier was 88.87% which was the highest compared to the other machine learning techniques. According to the evaluation metrics, the proposed approach achieved a precision of 0.90 and recall of 0.85 which is high in both benign URLs and phishing URLs. The achieved value of the F1-score of 0.87 proves that Random Forest performed well in dealing with the class imbalance and in the classification of the URLs into malicious and benign ones. The primary strength of Random Forest is that being an ensemble method it can capture feature interactions very well, which is crucial for URL classification tasks particularly when features are derived from URLs and can be very high dimensional.

## 7.3 Decision Tree Classifier

The decision Tree Classifier had a similar performance to Random Forest with an accuracy of 88.84 %, precision of 0.90, and recall of 0.85. Although the model had an acceptable accuracy and 0.87 of the F1 score, Decision Trees have high risks of overfitting particularly when working with a large dataset such as this one. However, the same model was relatively successful in classifying benign and malicious URLs and may not generalize

as effectively as Random Forest. It is easier to interpret meaning that; it can give a better understanding of the decision process but some weaknesses in terms of stability as compared to the ensemble models.

## 7.4 Logistic Regression

Logistic Regression resulted in an accuracy of 86.13%, the precision being 0.87 and the recall of 0.82. This model was computationally efficient, model had a lower recall score of 0.82 in detecting phishing URLs than other models. This suggests that Logistic Regression might not perform well with the more complicated complex relations which exists in URL data. The F1-score of 0.84 is the balance between precision and recall meaning that Logistic Regression is a simple and easy-to-interpreting model, it might not be capturing all the complexity in the data, and other models could be used.

## 7.5 Gradient Boosting Classifier

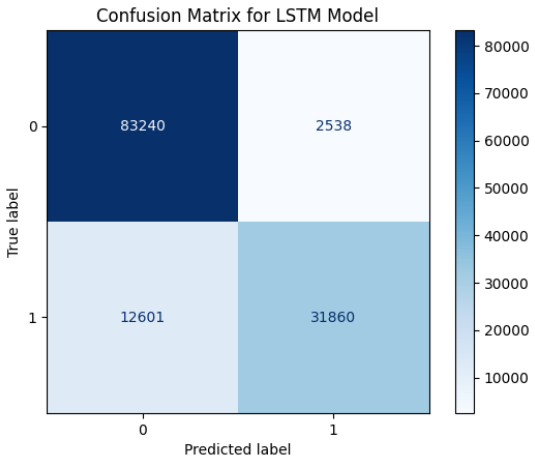
The Gradient Boosting Classifier has performed efficiently with an accuracy of 88.32 % a precision of 0.90 and a recall of 0.84. Like AdaBoost, Gradient Boosting aims is to correct errors of the previous model which makes it easy to recognize difficult cases. But it was slightly lower than Random Forest for precision and recall for detecting phishing URLs. The obtained F1-score of 0.86 means that the proposed model can be considered rather successful in addressing the problem of data imbalance. This makes it better at collecting more complex and precise patterns about the URL features and the sensitivity of the classifier could be improved to detect more phishing URLs.

## 7.6 XGB Classifier

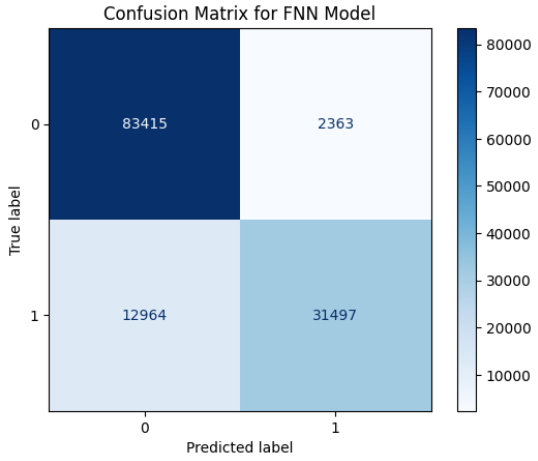
The XGBoosting Classifier has achieved the highest accuracy of 88.93%, precision of 0.90, and recall of 0.85. It showed high accuracy in benign and phishing URLs and its F1-score was 0.87. The optimized version of Gradient Boosting is called XGBoost and this model is known to be very effective for handling large complex datasets in our case XGBoost incorporates methods of regularization to minimize overfitting. It also had high accuracy, which is desirable in phishing detection as misclassification of a legitimate URL as a phishing one can have negative implications. In general, XGBoost is one of the most stable models for this task.

## 7.7 Custom Deep Learning Model

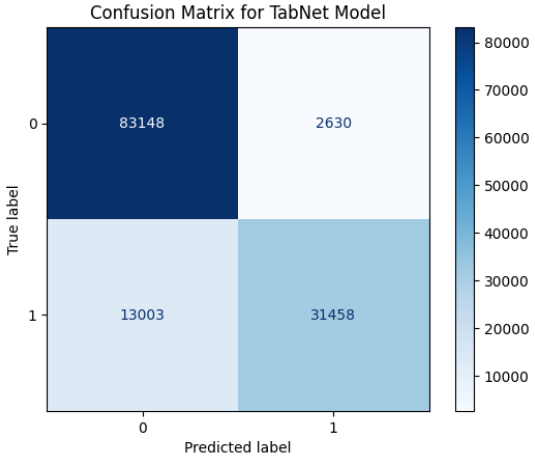
The custom deep learning model had an accuracy of 88.23%, a precision of 0.90, and a recall of 0.84. This model was intended to fit the non-linear relationships in the URL features where features of the URLs might interact with one another. Despite this, the deep learning model was as accurate as the other benchmark machine learning models, suggesting its ability to uncover subtle relationships within the data not discernible to ordinary models. An F1-score of 0.86 suggests that the proposed model was able to achieve a good trade-off between precision and recall, and therefore well suited to the task of phishing URL detection. Custom deep learning models are flexible in architecture allowing the architecture to fit the needs of the dataset.



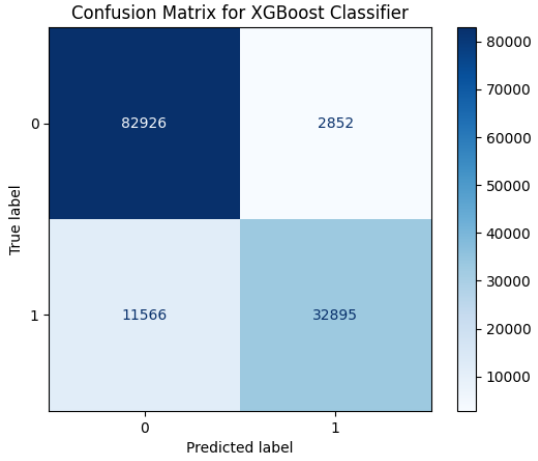

(a) Confusion matrix of LSTM




(b) Confusion matrix of Custom Deep Learning model




(c) Confusion matrix of TabNet model




(d) Confusion matrix of XgBoost

Figure 5: Confusion matrices of various models

## 7.8 TabNet Classifier

The TabNet model accuracy was 88.00%, precision 0.89, and recall 0.84. TabNet model used attention mechanisms. It was comparable to other deep learning models, but the precision for benign URLs was slightly less, 0.89, and the recall for phishing URLs could be enhanced by 0.84. The F1-score of 0.86 shows that TabNet maintains a good balance between accuracy and recall, it can successfully predict malicious URLs without losing sight of which features are driving the prediction.

## 7.9 LSTM Model

The LSTM model acquired an accuracy of 88.38%, slightly higher than other deep learning models. The LSTM model which is developed to capture temporal dependencies made satisfactory results in discovering temporal dependencies within the components of the URLs. This is especially useful in capturing relations in the order of subdomains, tokens, and symbols. In general, LSTM achieved an accuracy of 0.90 and a recall of 0.84 for benign and phishing URLs. Its F1-score of 0.86 shows that LSTM was balanced and was

able to utilize its sequential nature to analyze the structure of URLs and identify specific patterns that might not be noticeable to traditional approaches.

Overall, the models show the highest accuracy in the classification of phishing URLs with the machine learning models of XGB Classifier and Random Forest, the most accurate. Deep learning models, such as the custom deep learning model used, TabNet, and LSTM, were able to learn and recognize the patterns well. The outcomes highlight the significance of feature engineering and the capability of sophisticated models to differentiate between benign and malicious URLs.

## 7.10 Discussion and comparison of models

Both ML and DL models performed well in identifying the phishing URLs and differed slightly in accuracy, precision, recall, and F1 score. Among the machine learning models, the best result is shown by the Random Forest Classifier, which has the highest accuracy of 88.87% and a good precision-recall value of 0.87. Other models such as AdaBoost and Gradient boosting also performed well with AdaBoost having an accuracy of 87.80% and Gradient boost having an accuracy of 88.32 %. Logistic regression and decision tree were not as effective with accuracies of 86.13% and 88.84% respectively. XGBoost is another boosting model that is slightly better with an accuracy of 88.93% and high precision and recall, this is due to its ability to handle non-linearity between the features of the benign and phishing URLs.

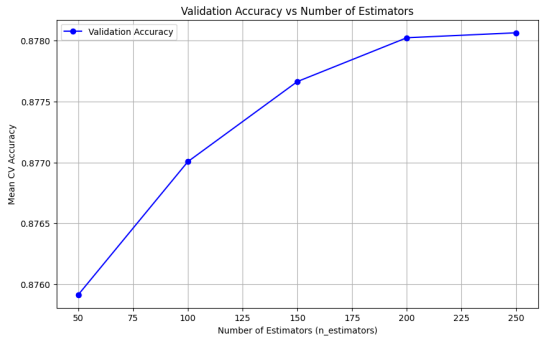
The results of the deep learning models including the custom deep learning model, TabNet, and LSTM were quite similar to the best machine learning models, suggesting the these model's ability to capture intricate URL feature patterns. TabNet obtained the worst result with an accuracy of 88.00%, followed by the custom deep learning model with an accuracy of 88.23% and LSTM with 88.38%. Still, these deep learning models did not achieve better performance than the machine learning models. These models had comparable precision and recall rates for the URL datasets, with minor variations in the efficiency of distinguishing between phishing URLs, with F1-scores of 0.86-0.87 for all models. Such features imply that deep learning models like LSTM and TabNet can help capture complex feature dependencies, but the other standard machine learning models, such as Random Forest and XGBoost, can also perform the same function for this classification process. In conclusion, both ML and DL models are more effective with slight variations so they can be adopted for detecting phishing URLs.

## 7.11 Reason for Deep Learning Models Not Outperforming Machine Learning Models

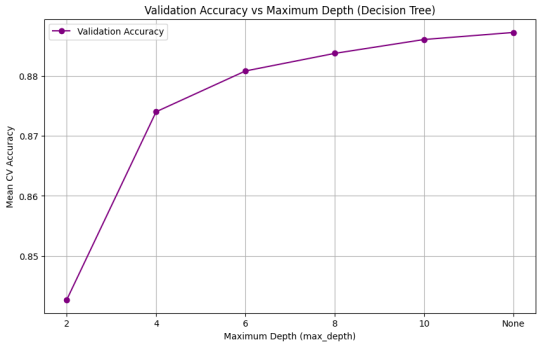
Though LSTM, TabNet, and custom deep learning architectures have shown better performance, they were not as significant as traditional machine learning models including Random Forest and XGBoost. One of the reasons for this is the type of data that is used by the model. These features derived from URLs are elaborate but purely tabular and Random Forest and XGBoost are designed to handle high-dimensional structured data. These models are very efficient in pattern recognition in the tabular data with decision trees fit for the URL classification task.

Deep learning models, which are very effective for sequential and unstructured data (text or image) may not show a huge enhancement when applied to structured data with well-defined features. LSTM and TabNet, as both methods are capable of learning com-

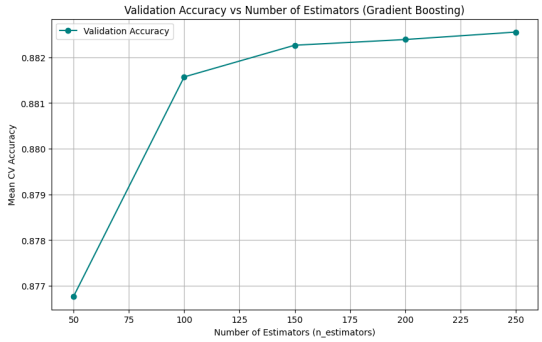
plex relations in the data provided, are not substantially superior to traditional models in this context, mainly because the data does not have inherent sequential or time series characteristics of the URL. Moreover, deep learning models in general perform best when trained on large datasets and our current dataset is large, it may not be large enough to take full advantage of the capabilities of deep learning models.



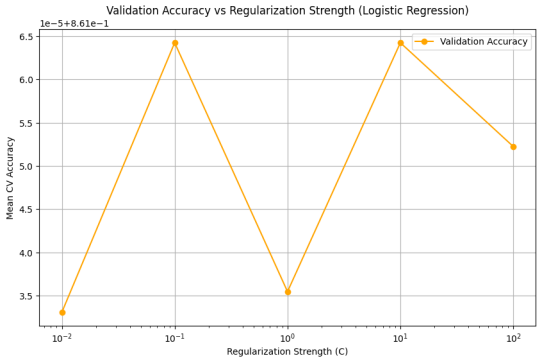

(a) Validation Accuracy on number of estimators on AdaBoost model




(b) Validation Accuracy on maximum depth of tree in decision tree




(c) Validation Accuracy on number of estimators on gradient boosting model



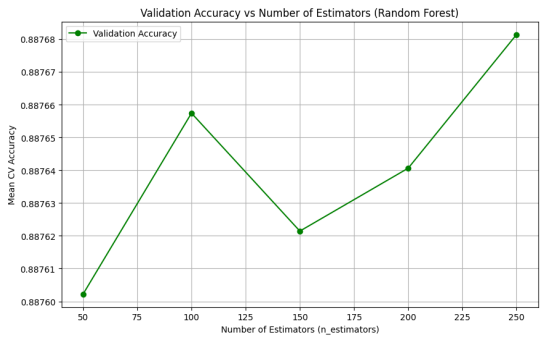

(d) Validation Accuracy over regularisation constant of logistic regression model

Figure 6: Hyper-parameter optimisation of various Machine learning models

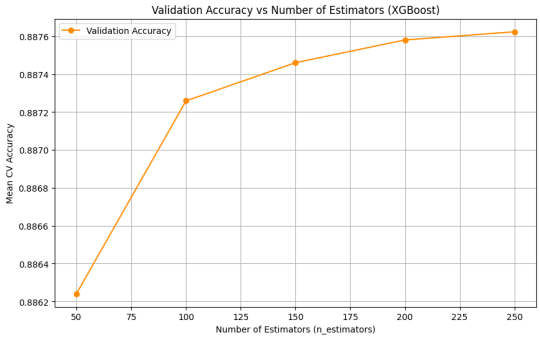
# 8 Hyper-Parameter Tuning

The hyperparameter selection was identified as the important factors that influenced the performance of both ML (Machine Learning) and DL (Deep Learning) approaches to detect phishing URLs. The machine learning models, several parameters were tuned to find a proper balance between model complexity and computational time. The AdaBoostClassifier was tuned by changing the number of estimators feature to 50, 100, 150, 200, and 250. This parameter determines the number of weak learner models iteratively to build a strong model. Similarly, RandomForestClassifier and DecisionTreeClassifier dealt with the maximum depth of decision trees as the parameters equal to 2, 4, 6, 8, 10, and 'none'. These adjustments enabled the identification of the appropriate level of complexity needed to identify patterns in the data without overemphasizing them. In Logistics Regression, the regularization strength with values 0.01, 0.1, 1, 10, and 100. Higher values of , reduced the problem of overfitting while lower values of  let the model

fit the training data more closely. Gradient Boosting and XGBoost classifiers also tried the number of estimators within the same range to ensure that their boosting processes deal with classification errors iteratively.




(a) Validation Accuracy on number of estimators on Random forest model




(b) Validation Accuracy on number of estimators on XGBoost model

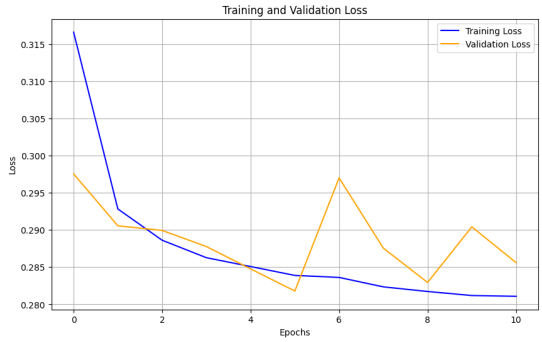
Figure 7: Hyper-parameter tuning of random forest and XgBoost models

To the deep learning models, early stopping was the most effective method to avoid overfitting and train needless epochs. TabNet, which uses attention mechanisms features to focus on, was trained to a maximum of 50 epochs. But it was applied up to 27 epochs because of early stopping which tracks the validation loss and stops the training process if the model does not improve for a set number of epochs. Likewise, the training of the LSTM model was performed for a maximum of 50 epochs and the model was optimized in 45 iterations. This approach helped both models to mitigate overfitting of the data.

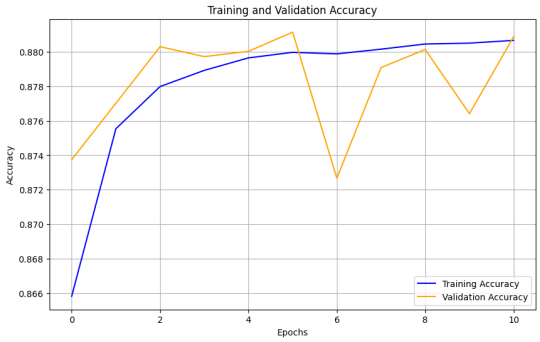
## 8.1 Experiments on secondary dataset

Experiments conducted on a recent dataset, comprising 134,850 legitimate and 100,945 phishing URLs, have revealed near-perfect accuracy across machine learning models like Random Forest, XGBoost, and Decision Tree. This exceptional performance highlights the dataset quality and the effectiveness of the extracted features in capturing critical distinctions between phishing and legitimate URLs. Features such as CharContinuationRate, URLTitleMatchScore, and TLDLegitimateProb, derived from the URL structure and webpage source code, provide highly discriminative information. The models' ability to leverage these rich features contributes to their superior classification accuracy, emphasizing the importance of a well-designed feature set in enabling robust phishing detection.

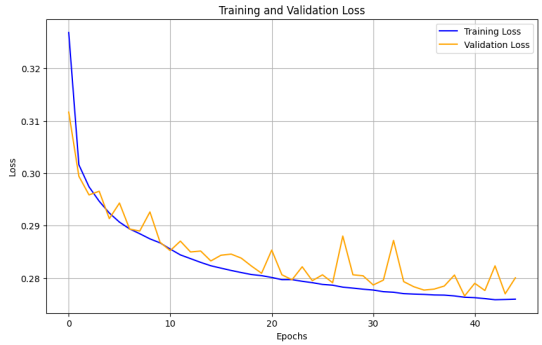
The results outlines the critical role of feature extraction in phishing URLs detection. More features often lead to greater insight into the underlying patterns of the data, as seen with metrics DomainLength, IsHTTPS, and URLSimilarityIndex in this dataset. By capturing nuanced details such as obfuscation, domain-specific characteristics, and webpage elements like hidden fields or external references, the dataset facilitates a detailed understanding of phishing behavior. This illustrates that the richness and relevance of features are pivotal in maximizing the performance of machine learning models, ensuring accurate and reliable detection.



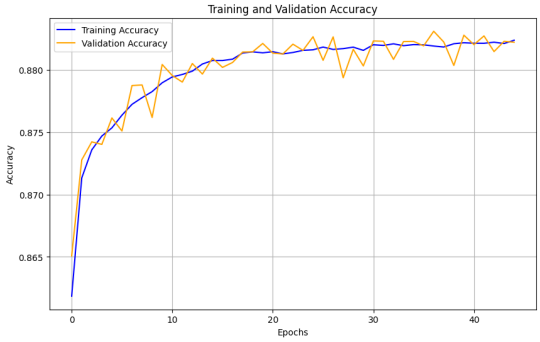

(a) Training and Validation loss plots of custom deep learning model




(b) Training and Validation accuracy plots of custom deep learning model




(c) Training and Validation loss plots of LSTM model




(d) Training and Validation accuracy plots of LSTM model

Figure 8: Various plots of deep learning models

# 9 Conclusion and Future work

## 9.1 Conclusion

This research assessed the effectiveness of machine learning and deep learning models in identifying phishing URLs using a large dataset comprising 651,191 entries including benign, phishing, defacement, and malware URLs. The hyperparameter of the models was then tuned and through systematic feature engineering the models were made as accurate, precise, recall, and F1-score as possible. Among the machine learning techniques, the highest accuracy was achieved by XGBoost and Random Forest algorithms, which provided a means of the intricate patterns within the data set. The LSTM and TabNet models also provided relatively high accuracy. The authors also noted that traditional machine learning models, especially ensemble methods, performed well because of the highly structured data, despite deep learning approaches demonstrating the capacity to learn complex dependencies in the data.

## 9.2 Future Work

Future work could explore by implementing multiple ensemble techniques that use both machine learning and deep learning together because they have their own advantages. Further, the datasets expansion to contain more cases and greater variability may enhance the models' generality in deep learning. Other research directions that could increase

classification ability of the models are associated with the exploration of better feature extraction techniques, for example, the application of regular expression searching for URL textual parts as NLP (natural language processing). Likewise, to overcome practical deployment issues, real-time phishing URL detection systems could be designed and tested. Studying adversarial attacks and the defense capabilities of models against threats of this type is also important to maintain the stability of phishing identification systems in modern Internet space.

# References


Adebowale, M., Lwin, K. and Hossain, M. (2023). Intelligent phishing detection scheme using deep learning algorithms, *Journal of Enterprise Information Management* **36**(3): 747–766.

Balantrapu, S. S. (2023). Evaluating the effectiveness of machine learning in phishing detection, *International Scientific Journal for Research* **5**(5).

Bauskar, S., Madhavaram, C., Galla, E., Sunkara, J. and Gollangi, H. (2024). Ai-driven phishing email detection: Leveraging big data analytics for enhanced cybersecurity, *Library Progress International* **44**(3): 7211–7224.

Divakaran, D. M. and Oest, A. (2022). Phishing detection leveraging machine learning and deep learning: A review, *IEEE Security & Privacy* **20**(5): 86–95.

Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E. and Fujita, H. (2022). Deep learning for phishing detection: Taxonomy, current challenges and future directions, *Ieee Access* **10**: 36429–36463.

El-Metwaly, A., Bedair, M., Abdallah, S., Mahmoud, A., Mohamed, M., Mahmoud, M. and Takieldeen, A. (2024). Detection of phishing urls based on machine learning and cybersecurity, *2024 International Telecommunications Conference (ITC-Egypt)*, IEEE, pp. 394–398.

Ozcan, A., Catal, C., Donmez, E. and Senturk, B. (2023). A hybrid dnn–lstm model for detecting phishing urls, *Neural Computing and Applications* pp. 1–17.

Rani, R., Silpa, N., Satish, G., Amrutha, N. and Reddy, G. (2024). Optimizing phishing detection: Leveraging url features with machine learning, *2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Vol. 1, IEEE, pp. 2094–2099.

Remya, S., Pillai, M. J., Nair, K. K., Subbareddy, S. R. and Cho, Y. Y. (2024). An effective detection approach for phishing url using resmlp, *IEEE Access* .

Sahingoz, O., Buber, E. and Kugu, E. (2024). Dephides: Deep learning based phishing detection system, *IEEE Access* .

Sahoo, D., Liu, C. and Hoi, S. C. (2017). Malicious url detection using machine learning: A survey, *arXiv preprint arXiv:1701.07179* .

Sameen, M., Han, K. and Hwang, S. O. (2020). Phishhaven—an efficient real-time ai phishing urls detection system, *IEEE Access* **8**: 83425–83443.

Tajaddodianfar, F., Stokes, J. and Gururajan, A. (2020). Texception: a character/word-level deep learning model for phishing url detection, *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, pp. 2857–2861.

Tamal, M., Islam, M., Bhuiyan, T., Sattar, A. and Prince, N. (2024). Unveiling suspicious phishing attacks: enhancing detection with an optimal feature vectorization algorithm and supervised machine learning, *Frontiers in Computer Science* **6**: 1428013.

Trinh, N. B., Phan, T. D. and Pham, V.-H. (2022). Leveraging deep learning image classifiers for visual similarity-based phishing website detection, *Proceedings of the 11th International Symposium on Information and Communication Technology*, pp. 134–141.

Ujah-Ogbuagu, B., Akande, O. and Ogbuju, E. (2024). A hybrid deep learning technique for spoofing website url detection in real-time applications, *Journal of Electrical Systems and Information Technology* **11**(1): 7.

Ullah, A., Shah, R., Nawaz, S., Ahmad, N. and Malik, M. (2024). Enhancing phishing detection, leveraging deep learning techniques, *Journal of Computing & Biomedical Informatics* .