University of Barisal

PHISHING WEBSITE DETECTION BY USING MACHINE LEARNING TECHNIQUES

Project Report

Submitted to

Dr. Tania Islam

Assistant Professor

Department of Computer Science and Engineering
University of Barishal

Submitted By

Popy Das

Roll: 09-001-05

Department of Computer Science and Engineering
University of Barishal

Contents

Introduction

Phishing attacks have been among the most prevalent and damaging threats in the digital landscape, exploiting the trust of online users to steal sensitive information, including usernames, passwords, and financial details. These attacks typically employed deceptive tactics such as spoofed emails and counterfeit websites that closely resembled legitimate ones, making detection challenging. As individuals and organizations increasingly relied on online platforms for banking, education, entertainment, and social networking, phishing became a significant security concern, affecting not only individuals but also businesses and governments.

Traditional phishing detection methods, such as blacklists and heuristic-based systems, had limitations in detecting newly emerging phishing sites. In contrast, machine learning (ML) techniques offered a more adaptive and efficient solution for phishing detection. By training models on datasets containing both phishing and legitimate websites, ML algorithms were able to identify patterns and features in URLs and website content indicative of phishing attempts.

The objective of this project was to develop an effective phishing detection system using machine learning techniques. A comprehensive dataset of phishing and legitimate websites was collected, and relevant features were extracted for analysis. Various machine learning models and deep neural networks (DNNs), including SVM, Random Forest, Multilayer Perceptrons, XGBoost, Decision Tree, and Autoencoder, were trained to predict phishing sites. The performance of these models was evaluated and compared to identify the most effective solution. The project aimed to enhance the early detection of phishing websites, thereby reducing the risk of users falling victim to such attacks.

Related Study

Phishing attacks have become an increasing cyber threat, targeting both individuals and organizations to steal sensitive information such as credit card details, login credentials, and personal data. Over the years, the frequency and sophistication of phishing incidents have steadily grown. For instance, in 2008, the Anti-Phishing Working Group recorded 51,401 phishing websites, and by 2016, global losses due to phishing were estimated at $9 billion,

according to RSA Security Inc. These statistics highlight the inadequacy of current anti-phishing solutions, which are unable to effectively combat the rapidly evolving phishing tactics that exploit users' trust in familiar websites and services [1, 2].

The rapid advancement of malicious software presents significant challenges within the cybersecurity landscape, requiring more effective detection mechanisms. Traditional malware detection methods, such as graph-based, rule-based, and entropy-based approaches, have proven impractical for addressing the complexities and dynamics of new malware variants. As cyber threats continue to evolve, these conventional techniques struggle to keep pace, underscoring the need for innovative solutions. Machine learning (ML) techniques have emerged as a promising alternative, offering enhanced capabilities for detecting and mitigating emerging malware threats [6].

In addition to supervised ML approaches, unsupervised techniques have also been explored to strengthen detection efforts. Widely used classification methods include Naïve Bayes (NB), Support Vector Machines (SVM), Random Forests (RF), and AdaBoost, each offering distinct advantages and limitations. Ensemble learning techniques, which combine multiple models to improve performance, have demonstrated superior results in combating malware attacks. This highlights the need for sophisticated detection methods that can adapt to the constantly changing landscape of cyber threats. To enhance defense against phishing attacks, automated and cognitive-based analysis systems are proposed, which benefit from a continuous flow of updated information on malware behaviors and variants, enabling more timely and accurate detection [6].

In phishing detection, machine learning techniques typically rely on feature extraction, which has shown to yield high accuracy rates. Research by Zhu et al. indicates that over 200 distinct features can be extracted from web data, providing a rich foundation for analysis [8]. However, the excessive number of features can complicate classifier design and result in overfitting, where the model becomes too tailored to the training data and fails to generalize to new data. This underscores the importance of optimal feature selection, a significant challenge in traditional machine learning approaches for phishing detection. The current research aims to identify the most relevant features for effective phishing detection.

To address this, the "Decision Tree and Optimal Features based Artificial Neural Network" (DTOF-ANN) methodology has been proposed. This approach uses artificial neural networks (ANNs) to build a robust classifier.

Prior to selecting optimal features, the importance of each feature is evaluated, leading to the formation of an optimal feature vector that enhances the classifier's performance by focusing on the most relevant data points [8].

In parallel efforts, Waleed Ali proposed a systematic procedure for detecting phishing websites using various supervised machine learning techniques, such as Radial Basis Function Networks (RBFN), Naïve Bayes Classifiers (NB), Back-Propagation Neural Networks (BPNN), Decision Trees, k-Nearest Neighbors (kNN), Random Forests (RF), and Support Vector Machines (SVM). Ali's approach utilizes wrapper feature selection based on these classifiers to optimize detection accuracy. However, research indicates that while neural network models can classify phishing attempts effectively, they are prone to underfitting if poorly structured, leading to inadequate performance. On the other hand, models that are overfitted to the training data tend to perform poorly in real-world applications [3, 4, 7].

The primary objective of this project was to develop an effective phishing detection system using advanced machine learning techniques. A comprehensive dataset of phishing and legitimate websites was collected, and relevant features were extracted for training various machine learning models and deep neural networks (DNNs) aimed at predicting phishing sites. The performance of these models was rigorously evaluated and compared to identify the most effective solution. Ultimately, the goal of this project was to improve early detection capabilities, thereby reducing the risk of users falling victim to phishing attacks.

Phishing website detection is particularly crucial in sectors such as online banking and trading, where users are more vulnerable to these attacks. While many users believe that anti-phishing techniques can protect them, current methods often fail to prevent phishing sites from bypassing defenses. Traditional approaches, such as URL blacklists, only detect known phishing sites, leaving users exposed to newly created threats. Meta-heuristics, which extract features such as URL length, domain age, and website content before applying classification techniques, are another approach for identifying phishing sites. However, these methods are not foolproof and can be circumvented by more sophisticated phishing techniques [5].

Machine learning (ML) has emerged as a more effective approach for phishing detection. ML models analyze a range of features extracted from websites, learning relationships between them to identify patterns that distinguish phishing websites from legitimate ones. These models offer greater flexibility and accuracy, as they can be trained on large datasets and con-

tinuously updated to recognize new phishing tactics. By incorporating ML-based methods, phishing detection systems can adapt to the evolving nature of phishing attacks, providing improved protection for online users.

Research Methodoloy

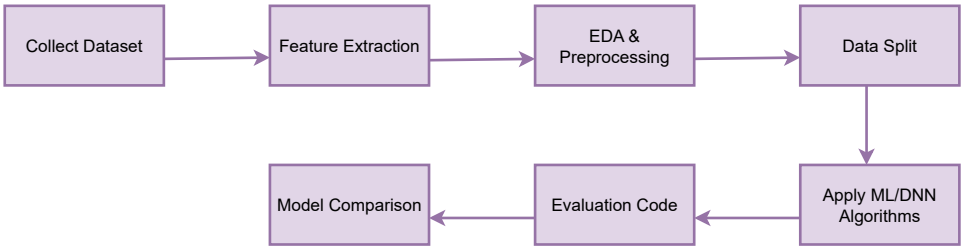

Figure 1: Step by Step Research Methodology

The methodology for detecting phishing websites involves several systematic steps to ensure effective data collection, analysis, and model evaluation. First, a dataset comprising both phishing and legitimate websites is collected from open-source platforms. This diverse dataset is essential for training machine learning models to recognize patterns associated with phishing attacks. Next, code is developed to extract relevant features from the URL database, focusing on critical characteristics such as domain names, URL lengths, and the presence of special characters, which are indicative of phishing behavior. Following feature extraction, exploratory data analysis (EDA) techniques are employed to analyze and preprocess the dataset. This step includes visualizing data distributions, identifying missing values, and normalizing features to enhance the dataset's quality. After preprocessing, the dataset is divided into training and testing sets, typically using an 80/20 split, to evaluate model performance accurately. Selected machine learning algorithms, such as Support Vector Machine (SVM) and Random Forest, along with deep neural network models like Autoencoders, are then applied to the training data. Each model is trained to identify phishing URLs based on the extracted features. To assess the effectiveness of the models, code is written to evaluate their performance using various accuracy metrics, including precision, recall, and F1-score. Finally, the results from the trained models are

compared, highlighting which algorithm performs best in detecting phishing attempts based on accuracy metrics. This comprehensive methodology ensures a robust approach to phishing detection, combining data-driven insights with machine learning techniques. Below is a detailed explanation of the methodology, focusing on feature extraction and model training.

Dataset Collection

The first step in the development of the phishing detection system involves the collection of a comprehensive and balanced dataset consisting of both phishing and legitimate websites. To ensure diversity and represent the various tactics used in phishing attacks, the dataset is sourced from reliable open-source platforms that provide labeled URLs. For the legitimate URLs, 5,000 URLs were randomly selected from the University of New Brunswick's publicly available *URL-2016* dataset, which contains a wide range of legitimate websites, including those from news, educational, governmental, and commercial domains. Similarly, phishing URLs were gathered from *PhishTank*, a well-known open-source service that offers up-to-date phishing websites identified by the community. PhishTank provides phishing URLs in formats such as CSV and JSON, and from this collection, 5,000 URLs were randomly selected. By combining 5,000 legitimate URLs with 5,000 phishing URLs, the dataset is both balanced and diverse, which is crucial for training machine learning models capable of distinguishing between phishing and legitimate websites across a variety of attack methods and domains.

Feature Extraction

Feature extraction plays a crucial role in detecting phishing websites, as it helps capture various characteristics that are indicative of malicious behavior. A total of 17 features are extracted from the dataset, which are categorized into three main groups: Address Bar-Based Features, Domain-Based Features, and HTML and JavaScript-Based Features. These features are designed to highlight patterns and anomalies that are commonly associated with phishing websites.

0.0.1 Address Bar-Based Features

These features focus on the structure and content of the URL itself. Phishing websites often manipulate URL elements to appear legitimate, making these features essential for detection.

- **Domain of URL**: Identifying the domain name is fundamental, as phishing sites often attempt to mimic the domain names of legitimate sites. By detecting unusual or suspicious domain names, phishing attempts can be flagged.
- **Redirection ('//') in URL**: Phishing URLs may contain double slashes (//) in the URL to redirect users to a malicious site. This feature checks for unusual URL redirections, which are a common technique used in phishing.
- **IP Address in URL**: Phishing websites sometimes use raw IP addresses in the URL instead of domain names. This feature helps detect these cases by identifying IP addresses embedded within URLs.
- **'http/https' in Domain Name**: The presence of 'http' instead of 'https' in the domain name could indicate a phishing site. Phishing sites are often not secured with HTTPS, which is a hallmark of legitimate websites.
- **@ Symbol in URL**: The '@' symbol in the URL may trick users into thinking a legitimate domain is being used. This feature detects such instances, which are commonly seen in phishing URLs.
- **Use of URL Shortening Services**: Phishing sites may use URL shortening services to disguise their actual destination. This feature detects shortened URLs that are often used to obscure the real website.
- **Length of URL**: Phishing URLs often have an unusually long length or contain excessive characters to hide their true intent. This feature identifies URLs that may be suspicious due to their length.
- **Prefix or Suffix ('-') in Domain**: Phishing sites frequently use hyphens ('-') in domain names, which are relatively rare in legitimate domains. This feature flags domain names containing hyphens as potentially suspicious.

- **Depth of URL**: The depth of the URL refers to the number of directories in the URL path. Phishing sites often use unusually deep URL structures to hide their true location and mislead users.

0.0.2 Domain-Based Features

These features help assess the credibility and trustworthiness of the domain hosting the website. A domain's characteristics are often indicative of phishing activity.

- **DNS Record**: The presence of a DNS record for a website indicates whether the domain is valid and associated with a trustworthy service. Phishing websites may lack proper DNS records.
- **Age of Domain**: Older domains are typically more trustworthy, whereas phishing sites often utilize newly registered domains. This feature detects domains that are registered for a short period, a common practice in phishing campaigns.
- **Website Traffic**: Legitimate websites tend to have higher traffic, while phishing sites often experience minimal or irregular traffic. This feature examines website traffic patterns to assess the likelihood of a site being malicious.
- **End Period of Domain**: The expiration date of a domain is another indicator of its legitimacy. Phishing sites often use domains with short expiration periods, while legitimate websites tend to have longer registration periods.

0.0.3 HTML and JavaScript-Based Features

These features focus on the content and behavior of the website itself, particularly scripts and elements used for malicious purposes.

- **Iframe Redirection**: Phishing sites frequently use iframes to embed content from external sites, making it harder to detect their true intent. This feature checks for the use of iframes that may indicate phishing behavior.

- **Disabling Right Click**: To prevent users from inspecting the source code or interacting with the site, phishing websites often disable the right-click functionality. This feature detects the presence of such behavior on the site.

- **Status Bar Customization**: Phishing websites may manipulate the status bar to display fake security messages, such as a false "secure connection" or other misleading information. This feature identifies instances where the status bar is customized to deceive users.

- **Website Forwarding**: Some phishing websites use forwarding techniques to redirect users to different sites, often after a delay. This feature detects website forwarding, which is commonly used to confuse users and direct them to harmful destinations.

Together, these 17 features provide a comprehensive profile of each website, helping to distinguish between legitimate and phishing sites. The combination of URL structure, domain characteristics, and website content allows the machine learning models to identify phishing behavior with greater precision, improving the overall detection accuracy.

Exploratory Data Analysis (EDA)

After the feature extraction step, the next critical phase in the data preprocessing pipeline is Exploratory Data Analysis (EDA). EDA plays a vital role in understanding the dataset's underlying structure, identifying patterns, and detecting any potential issues that may affect model performance. The goal of EDA is to ensure that the data is well-prepared for training the machine learning models. The following key steps are involved in the EDA process:

0.0.4 Visualization of Data Distributions

Visualizing the distribution of features is essential for understanding the data's underlying patterns and checking for imbalances or outliers. The distribution of each feature is plotted using histograms, box plots, or kernel density plots, allowing for an assessment of how the data is spread and whether certain values dominate or show unusual patterns. This helps identify:

- **Imbalances in the Dataset:** Phishing datasets may be imbalanced, with more legitimate URLs than phishing ones, which could bias model performance. Visualizing the class distribution helps identify such issues.
- **Feature Skewness:** Features with skewed distributions may need special handling to prevent model overfitting.
- **Outliers:** Identifying extreme values that may disproportionately influence the model's performance.

For example, if the feature "Length of URL" has a few exceptionally long URLs, it can suggest that such cases might be anomalies, and further treatment may be necessary.

0.0.5 Handling Missing Data

Missing data is a common issue in real-world datasets, and how it is handled can significantly impact the performance of machine learning models. In the EDA process, missing values are identified using techniques such as:

- **Null Value Check:** A quick check is performed to identify any missing values in the dataset.
- **Imputation:** If the missing data is small and random, it can be imputed using the mean, median, or mode of the feature, depending on the nature of the data.
- **Removal of Rows/Columns:** If a feature has too many missing values or if the missing data is not random, it may be best to remove the column or row entirely to avoid introducing bias into the model.

Proper handling of missing data ensures that the training process is not impacted by gaps in the dataset and that the model can learn effectively from complete data points.

0.0.6 Normalization

Normalization is a critical step, especially when machine learning algorithms like Support Vector Machine (SVM), k-Nearest Neighbors (kNN), or neural networks are used. These algorithms are sensitive to the scale of the input

features, and features with different scales can bias the model or degrade its performance. The goal of normalization is to transform the features so that they all have a comparable scale. Common techniques for normalization include:

- **Min-Max Scaling:** This technique transforms each feature to a specific range, usually between 0 and 1, by subtracting the minimum value of the feature and dividing by the range of values (max-min).
- **Z-Score Standardization:** This technique transforms the feature to have a mean of 0 and a standard deviation of 1 by subtracting the mean of the feature and dividing by the standard deviation.

Normalization ensures that no single feature disproportionately influences the model due to its larger numerical scale, allowing the machine learning algorithm to process all features uniformly.

By performing thorough EDA, any underlying patterns, biases, or issues with the data are identified and addressed. This ensures that the dataset is properly prepared and ready for model training, ultimately contributing to the development of a more accurate and reliable phishing detection system.

Data Splitting

The data splitting process is an essential step in preparing the dataset for training machine learning models, ensuring that the model can generalize well to unseen data. In this project, the dataset is split into training and testing sets using an 80/20 ratio. This means that 80% of the data is used to train the model, while the remaining 20% is reserved for testing its performance. This split ensures that the model has enough data to learn the patterns while also being evaluated on data it has not encountered during training, thus preventing overfitting. To maintain the original distribution of phishing and legitimate websites, stratified sampling is applied, ensuring that both categories are proportionally represented in both the training and testing sets. This is particularly important in cases where the dataset is imbalanced, as it prevents the model from being biased toward the majority class. Additionally, before splitting, the data is shuffled to randomize the order of URLs, preventing the model from learning unintended sequential patterns. Proper data splitting is crucial for evaluating the model's generalization ability and ensuring that it performs well on real-world, unseen phishing threats.

Model Selection and Training

In this study, multiple machine learning (ML) models and deep learning techniques are utilized to detect phishing websites based on the extracted features. Each model is carefully selected for its unique ability to handle different types of data patterns and complexities, allowing for a comprehensive approach to phishing detection. The models used include:

0.0.7 Support Vector Machine (SVM)

SVM is a powerful supervised learning algorithm that aims to find the optimal hyperplane that best separates the data points of different classes—in this case, phishing and legitimate websites. It works effectively in high-dimensional feature spaces, which is crucial given the large number of features extracted from the URLs. SVM is known for its robustness and ability to handle complex, non-linear relationships in data through the use of kernel functions.

0.0.8 Random Forest (RF)

Random Forest is an ensemble learning method that combines the predictions of multiple decision trees to produce a more accurate and stable result. Each decision tree is trained on a random subset of the data, and the final prediction is based on the majority vote from all the trees. This model is particularly effective in reducing overfitting and provides feature importance metrics, which help in understanding which features most contribute to the model's decision-making process.

0.0.9 Multilayer Perceptrons (MLP)

MLP is a type of artificial neural network that consists of multiple layers of neurons, each layer fully connected to the next. It is capable of capturing complex, non-linear relationships between the input features and the target variable. MLP is particularly useful for recognizing intricate patterns in the data, and its depth enables it to model complex interactions that simpler models may miss.

0.0.10 XGBoost (Extreme Gradient Boosting)

XGBoost is an advanced gradient boosting technique that iteratively trains decision trees and corrects the errors made by previous trees. It is known for its high performance and efficiency, especially in handling imbalanced datasets. XGBoost uses regularization techniques to prevent overfitting, making it an ideal model for classification tasks with a large feature set, like phishing detection. Its boosting nature ensures that weak models are corrected and improved upon in each iteration.

0.0.11 Decision Tree (DT)

A Decision Tree is a simple yet effective model that makes decisions based on feature thresholds. It splits the dataset into subsets using the most informative features at each node, creating a tree-like structure where each branch represents a decision rule. The simplicity of Decision Trees makes them easy to interpret and understand, though they can sometimes be prone to overfitting if not carefully tuned.

0.0.12 Autoencoder (Deep Neural Network)

Autoencoders are a type of deep neural network used for anomaly detection. In this context, autoencoders are trained to reconstruct input data (website features) and are used to flag outliers—websites that deviate significantly from the norm, which in this case, are phishing sites. The autoencoder learns to compress the data into a lower-dimensional representation and then reconstructs it to match the original. Phishing websites, which have characteristics that differ from legitimate websites, are identified by high reconstruction errors, signaling that the model cannot accurately recreate these outlier websites.

Each of these models is trained on the *training set*, with the goal of learning the patterns and characteristics that distinguish phishing websites from legitimate ones. After training, the models are evaluated using various performance metrics, such as *accuracy*, *precision*, *recall*, *F1-score*, and *ROC-AUC* to assess their effectiveness in detecting phishing sites. These metrics allow for a detailed comparison of each model's strengths and weaknesses, ensuring that the most reliable model is chosen for phishing detection. By employing a range of models, this approach seeks to maximize detection accuracy and minimize false positives and false negatives.

Model Evaluation

Model performance is primarily evaluated using **accuracy**, which is the most fundamental metric for assessing classification models. Accuracy measures the proportion of correctly classified websites—both phishing and legitimate—out of the total number of predictions made by the model. It is calculated as the ratio of the number of correct predictions (true positives and true negatives) to the total number of instances in the dataset. Accuracy is particularly important in this study as it provides a clear indication of how well the model is performing in distinguishing phishing websites from legitimate ones. High accuracy implies that the model is consistently making correct predictions, whereas low accuracy suggests that the model may be struggling with either false positives (classifying legitimate sites as phishing) or false negatives (failing to detect phishing sites). Therefore, the accuracy metric serves as a key measure in determining the overall effectiveness of the different machine learning models trained to detect phishing websites.

Final Outcome

The table below summarizes the performance of different machine learning models, including their training and test accuracies. It shows how each algorithm performed on the dataset, which provides insights into its effectiveness and generalizability to unseen data.

ML Model	**Train Accuracy**	**Test Accuracy**
Multilayer Perceptrons	0.862	0.854
XGBoost	0.870	0.850
Random Forest	0.821	0.807
Decision Tree	0.815	0.797
SVM	0.804	0.795
Autoencoder	0.499	0.503

Table 1: Training and Test Accuracy of Various Machine Learning Models

Here's the description of the table converted into a paragraph format:

The model analysis reveals that **Multilayer Perceptrons (MLP)** exhibit strong performance with a training accuracy of 86.2% and a test accuracy of 85.4%. This indicates that MLPs are capable of learning complex

patterns from the data and generalizing well to unseen data, with only a minor difference between the training and test accuracies (0.8%), suggesting that the model is well-regularized and not prone to overfitting.

XGBoost performs slightly better in terms of training accuracy (87.0%) but achieves a similar test accuracy of 85.0%. This shows that while XGBoost's gradient boosting technique provides a slight advantage during training, it generalizes almost as effectively as MLP. The small drop between the training and test accuracies indicates that XGBoost is also well-regularized and performs well in distinguishing phishing from legitimate sites.

The **Random Forest** model, with a training accuracy of 82.1% and a test accuracy of 80.7%, demonstrates slightly lower performance compared to MLP and XGBoost. Despite this, Random Forest, being an ensemble model, still provides robust performance, and the minimal gap between training and test accuracies suggests it is not overfitting, though it might be slightly underfitting.

Decision Tree, with a training accuracy of 81.5% and a test accuracy of 79.7%, performs similarly to Random Forest but with slightly lower accuracy across both training and testing datasets. The Decision Tree model tends to either overfit or underfit the data more than ensemble models like Random Forest or XGBoost, which can lead to less stable predictions, particularly on the test set. However, it is an interpretable model that offers insights into the decision-making process.

SVM (Support Vector Machine) achieves a training accuracy of 80.4% and a test accuracy of 79.5%. Although SVM performs reasonably well, it is outperformed by ensemble models and neural networks in this case. The small difference between training and test accuracy shows that the SVM is well-regularized and effective for high-dimensional data, but it does not outperform the other models for phishing detection.

Lastly, the **Autoencoder** model, which is designed for anomaly detection, shows a significantly lower performance, with a training accuracy of only 49.9% and a test accuracy of 50.3%. This is expected, as Autoencoders are unsupervised models meant to detect outliers by reconstructing input data. In this context, the Autoencoder struggles to capture the complex patterns associated with phishing websites, resulting in very low accuracy. This suggests that Autoencoders are not suitable for this classification task without significant modifications.

Overall, the comparative analysis shows that **XGBoost** and **MLP** are the most effective models, with both achieving the highest test accuracies,

making them the preferred choices for phishing detection tasks.

Conclusion

In this study, various machine learning and deep learning models were evaluated for phishing website detection, including SVM, Random Forest, MLP, XGBoost, Decision Tree, and Autoencoders. The goal was to identify a model that could accurately distinguish between phishing and legitimate websites. Among the models tested, **MLP** and **XGBoost** performed the best, with test accuracies of 85.4% and 85.0%, respectively, demonstrating strong generalization and high precision. In contrast, the **Autoencoder** model showed significantly lower performance, indicating it is not well-suited for phishing detection without modifications. These results highlight the importance of selecting advanced algorithms like **MLP** and **XGBoost** for phishing detection, and suggest that further research could focus on enhancing feature engineering and exploring hybrid models to improve accuracy.

References

[1] H Bleau. Global fraud and cybercrime forecast, 2016, 2017.

[2] Kang Leng Chiew, Choon Lin Tan, KokSheik Wong, Kelvin SC Yong, and Wei King Tiong. A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. *Information Sciences*, 484:153–166, 2019.

[3] Stefan Duffner and Christophe Garcia. An online backpropagation algorithm with validation error-based adaptive learning rate. In *International Conference on Artificial Neural Networks*, pages 249–258. Springer, 2007.

[4] Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. Predicting phishing websites based on self-structuring neural network. *Neural Computing and Applications*, 25:443–458, 2014.

[5] Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, and Banu Diri. Machine learning based phishing detection from urls. *Expert Systems with Applications*, 117:345–357, 2019.

[6] Jagsir Singh and Jaswinder Singh. A survey on machine learning-based malware detection in executable files. *Journal of Systems Architecture*, 112:101861, 2021.

[7] Fadi Thabtah, Rami M Mohammad, and Lee McCluskey. A dynamic self-structuring neural network model to combat phishing. In *2016 international joint conference on neural networks (ijcnn)*, pages 4221–4226. IEEE, 2016.

[8] Erzhou Zhu, Yinyin Ju, Zhile Chen, Feng Liu, and Xianyong Fang. Dtof-ann: an artificial neural network phishing detection model based on decision tree and optimal features. *Applied Soft Computing*, 95:106505, 2020.