Grenze International Journal of Engineering and Technology, January Issue

A Review Study on Advanced Hybrid System for URL Threat Detection

Amol B. Majgave[1] and Nitin L. Gavankar[2]
[1]Research Scholar, Shivaji University, Kolhapur, Maharashtra, 416004, India, Assistant professor, DKTE Society's Textile and Engineering Institute Ichalkaranji, Maharashtra 416115, India
Email: majgaveamolb@gmail.com
[2]Associate Professor, Walchand College of Engineering Sangli, Maharashtra, 416415, India,
Email: nitin.gavankar@walchandsangli.ac.in

***Abstract*— With the exponential growth of web-based services, the security of online resources has become increasingly critical. Malicious URLs pose a significant threat to both individuals and organizations, often leading to data breaches, malware infections, and phishing attacks. Traditional detection methods struggle to keep up with the evolving techniques used by cybercriminals.**

This paper presents a comprehensive review of advanced hybrid systems that combine multiple machine learning models for effective URL threat detection. Specifically, we focus on hybrid systems incorporating Decision Trees (DT), Support Vector Machines (SVM), and Naive Bayes (NB) models, which have shown promising results in handling the complexities of URL classification. Decision Trees offer interpretability and flexibility, while SVM excels in high-dimensional feature spaces, and Naive Bayes provides simplicity with strong probabilistic foundations. By combining these models, hybrid systems can leverage the strengths of each to achieve higher accuracy, faster detection, and improved generalization. The paper reviews various hybrid approaches, evaluates their performance, and identifies challenges and opportunities for future research in URL threat detection. Our study aims to provide a thorough understanding of the synergies between these models and their potential for building more robust, scalable, and real-time security systems for web applications.

***Index Terms*— Phishing, URL, machine learning, cybersecurity, random forest, malicious.**

I. INTRODUCTION

As the Internet develops and grows, many of our activities are now conducted online, including e-commerce, business, social networking, and banking, raising the likelihood of online crime. So, securing the world wide web is becoming increasingly important. Phishing attacks are one of the most common methods used by cybercriminals to deceive users into revealing personal or sensitive data. Phishing URLs often mimic legitimate websites to trick users into entering their private information. Early detection and prevention of phishing URLs are crucial for protecting users and organizations from these attacks.

While several approaches have been proposed for phishing detection, many rely on basic methods that often fail to adapt to the evolving tactics of phishing websites. Recent advancements in machine learning (ML) offer promising techniques to detect phishing URLs with high accuracy.

Grenze ID: 01.GIJET.12.1.8

This research proposes a hybrid machine learning system that combines multiple machine learning models to improve phishing URL detection. The hybrid system integrates decision trees, support vector machines (SVM), and Naïve bayes algorithms to leverage the strengths of each approach, providing more robust and accurate detection.

II. Background and Related Work

Several methods have been proposed for detecting phishing URLs

A. Traditional Phishing Detection Techniques

Blacklist-Based Detection is a traditional technique used to identify and block access to known malicious URLs, IP addresses, or domains by maintaining a database called a blacklist of these harmful entities. When a user attempts to visit a website or access a resource, the system checks whether the URL or IP address is listed in the blacklist. If it is, the system blocks the request or displays a warning to prevent the user from proceeding. This method is straightforward, easy to implement, and has a low false positive rate since it relies on previously confirmed threats. However, it has significant limitations, particularly its inability to detect new or previously unknown phishing sites, known as zero-day threats. To remain effective, the blacklist must be continuously updated with the latest malicious URLs, which can be resource-intensive and reactive rather than proactive. Despite its simplicity, blacklist-based detection remains a foundational layer in many web security systems and is often used in combination with more advanced techniques.

Heuristic-Based Detection is a method of identifying phishing websites or emails by analyzing specific features and patterns that are commonly associated with malicious behavior. Unlike blacklist-based detection, which relies on known entries, heuristic detection uses predefined rules or algorithms to evaluate the content, structure, and behavior of a webpage or email in real-time. For example, it may flag websites that use suspicious URL structures, such as IP address-based domains or typosquatted domains (e.g., g00gle.com), as potentially malicious. It can also detect phishing attempts by looking for characteristics like missing SSL certificates, excessive use of redirection, hidden form fields, or the presence of login forms on non-secure pages. In emails, heuristic analysis may identify threats based on urgent language, mismatched sender information, or embedded scripts and attachments. The main advantage of heuristic-based detection is its ability to identify new or unknown threats that are not yet listed in blacklists. However, it can sometimes produce false positives by flagging legitimate sites or emails that happen to match certain suspicious patterns. Despite this, heuristic detection is a valuable component of multi-layered security systems, offering proactive defense against evolving phishing tactics.

Visual Similarity Detection is a phishing detection technique that identifies fraudulent websites by comparing their visual appearance to that of legitimate sites they attempt to mimic. This method analyzes elements such as logos, color schemes, layout structures, fonts, and images to detect visual imitations of well-known websites, especially login pages of banks, email providers, and social media platforms. By rendering the suspected phishing page and extracting visual features, the system compares them to a database of legitimate site templates or screenshots using techniques like image hashing, template matching, or structural analysis. If a high degree of visual similarity is found—such as a nearly identical login form or branding—it is flagged as a potential phishing attempt. The primary advantage of visual similarity detection is its effectiveness in uncovering sophisticated phishing sites that may bypass textual or code-based filters. However, this method is computationally intensive and may struggle with minor visual modifications designed to evade detection. Additionally, it requires maintaining an up-to-date database of legitimate site visuals. Despite these challenges, visual similarity detection is a powerful tool, particularly when combined with other detection methods, to enhance the accuracy of phishing identification systems.

Email Header and Content Analysis is a traditional phishing detection technique that involves examining both the metadata (header) and the body (content) of an email to identify signs of malicious intent. The email header contains critical information such as the sender's address, IP origin, return path, and authentication results (e.g., SPF, DKIM, DMARC), which can be analyzed to detect spoofed or forged sender identities. For example, if the email claims to be from a trusted domain but fails domain authentication checks or originates from a suspicious IP address, it may be flagged as phishing. The content analysis focuses on the body of the email, evaluating elements like urgent language (“Your account will be suspended”), misleading hyperlinks (e.g., display text says one domain but the actual link points elsewhere), unexpected attachments, and requests for sensitive information. This method may also use keyword matching, rule-based filters, or pattern recognition to detect phishing characteristics. While email header and content analysis is useful for identifying many common phishing tactics,

it can be less effective against well-crafted or highly obfuscated emails and may generate false positives if legitimate messages share similar traits. Nonetheless, it remains a vital component of email security systems, particularly in conjunction with spam filters and machine learning-based methods.

B. Machine Learning Approaches

Machine Learning Approaches for phishing and malicious URL detection involve training algorithms to automatically learn patterns from data and make predictions or classifications based on learned knowledge. These approaches are more adaptive and scalable compared to traditional rule-based methods. Machine learning models can analyze a wide range of features extracted from URLs, web content, or emails to distinguish between legitimate and malicious behaviour. In supervised learning, the model is trained on labeled data, where each example is marked as either "malicious" or "benign." The goal is to learn a mapping from features (e.g., URL length, domain age, presence of special characters) to the output label. Below are some common machine learning approaches used in cybersecurity:

Decision Trees (DT) are a widely used supervised machine learning algorithm that operates by learning simple decision rules inferred from data features to perform classification or regression tasks. In the context of URL threat detection, a decision tree works by recursively splitting the dataset into subsets based on the value of input features such as URL length, presence of special characters, or use of IP addresses instead of domain names. Each internal node of the tree represents a decision based on a feature, each branch corresponds to an outcome of that decision, and each leaf node represents a final classification label—such as “malicious” or “benign.” The decision tree continues to grow until it reaches a stopping condition, such as a maximum depth or minimum number of samples in a node. One of the major advantages of decision trees is their interpretability; the model's logic is easy to understand and visualize, making it especially useful for security analysts who need to explain or audit decisions.

Additionally, decision trees can handle both numerical and categorical data and do not require extensive data preprocessing. However, a major drawback is their tendency to overfit the training data, especially when the tree becomes too deep. To mitigate this, pruning techniques or ensemble methods like Random Forests are often used. Despite their limitations, decision trees remain a fundamental building block in many hybrid and ensemble systems for cyber threat detection due to their simplicity, speed, and interpretability.

Support Vector Machines (SVM) are powerful supervised machine learning algorithms primarily used for classification tasks, including the detection of malicious URLs. The core idea behind SVM is to find the optimal hyperplane that best separates data points of different classes in a high-dimensional space. In binary classification problems like distinguishing between “malicious” and “benign” URLs, SVM attempts to maximize the margin between the two classes—this margin is defined by the closest data points on each side, known as support vectors. SVM is particularly effective when the data is not linearly separable, as it can use kernel functions (such as polynomial, radial basis function, or sigmoid) to transform the original feature space into a higher-dimensional space where a linear separation becomes possible. This ability makes SVM suitable for complex and non-linear URL patterns often seen in cyber threats. The algorithm is robust to overfitting, especially in high-dimensional spaces, and performs well even with smaller datasets. However, SVMs can be computationally intensive, especially with large datasets, and choosing the appropriate kernel and tuning hyperparameters can be challenging. Despite these challenges, SVM remains a highly accurate and reliable model for threat detection tasks and is often included in hybrid systems to complement simpler, faster models like Naive Bayes or Decision Trees.

Naive Bayes (NB) is a simple yet highly effective supervised machine learning algorithm based on Bayes' Theorem, widely used for classification tasks such as spam detection, email filtering, and malicious URL identification. The core principle of Naive Bayes is to calculate the probability that a given input belongs to a particular class (e.g., “malicious” or “benign”) based on the observed features, assuming that all features are conditionally independent of each other given the class label. Although this “naive” assumption rarely holds true in real-world scenarios, the algorithm still performs surprisingly well in many applications due to its efficiency and ability to generalize well with relatively small datasets. In the context of URL threat detection, Naive Bayes can analyze features such as the frequency of suspicious keywords, domain structure, or the presence of unusual characters to calculate the likelihood of a URL being malicious. One of the main advantages of NB is its speed—it requires minimal training time and performs well even on large datasets. It is also easy to implement and interpret, making it suitable for baseline models or integration into hybrid systems. However, its main limitation lies in the strong independence assumption, which can lead to lower accuracy when the features are highly correlated. Despite this, Naive Bayes continues to be a popular choice for initial threat classification tasks due to its simplicity, scalability, and reasonable accuracy.

Random Forest is an ensemble learning technique that combines multiple decision trees to improve classification and regression accuracy by reducing overfitting and increasing robustness. The idea behind Random Forest is to create a "forest" of decision trees, each trained on a random subset of the data, both in terms of data points (through bootstrapping) and features (by selecting a random subset of features at each split). Each individual tree in the forest is trained independently and makes its own prediction, and the final prediction is made by aggregating the predictions of all trees—typically by majority voting for classification tasks or averaging for regression tasks.

This randomization helps Random Forest overcome the problem of overfitting that single decision trees often face, leading to better generalization on unseen data. In the context of malicious URL detection, Random Forest can analyze a wide variety of features, such as URL length, domain name, use of special characters, and the presence of known malicious patterns, and combine the insights from multiple trees to make more accurate and stable predictions. One of the key advantages of Random Forest is its ability to handle large datasets with high dimensionality and its resilience to noise in the data. Moreover, it can provide an estimate of feature importance, helping analysts understand which features contribute most to the model's decisions. However, the model's complexity, due to the large number of trees, can make it computationally expensive and less interpretable compared to a single decision tree. Despite these challenges, Random Forest remains one of the most widely used and effective models in machine learning, particularly for classification tasks like detecting phishing URLs or malicious websites.

III. Hybrid Approaches

A Hybrid Approach in machine learning refers to the combination of two or more different models or algorithms to leverage their individual strengths and improve overall performance. The core idea behind hybrid systems is that combining diverse models can reduce weaknesses associated with any single model, such as overfitting, underfitting, or poor generalization. In the context of URL threat detection or phishing detection, a hybrid approach often integrates models like Decision Trees, Support Vector Machines (SVM), and Naive Bayes, each of which has different characteristics. For instance, Decision Trees offer interpretability and are good at capturing complex relationships, SVMs excel in high-dimensional spaces and are robust to overfitting, while Naive Bayes is fast and performs well with limited data. By combining these models, a hybrid system can take advantage of the accuracy of SVMs, the speed and simplicity of Naive Bayes, and the interpretability of Decision Trees.

In practice, hybrid approaches can be implemented through methods like voting (where each model casts a "vote" on the class label), stacking (where the output of one model is used as an input for another model), or bagging/boosting (ensemble methods that reduce variance and bias). These systems tend to outperform individual models because they are more robust to noise and can handle more complex patterns in the data. Hybrid approaches are particularly effective in security-related tasks, such as detecting phishing attacks, as they offer a higher level of accuracy, faster detection times, and improved adaptability to new threats.

A. Key Features for URL Classification

Lexical features: Lexical features play a crucial role in URL classification by analyzing the structural and textual characteristics of the URL string itself, without accessing the content of the corresponding web page. These features are derived directly from the URL and help identify patterns commonly associated with legitimate or malicious websites.

Key lexical features include the length of the URL, the number of special characters (such as “-”, “_”, “@”, or “%”), the presence and number of subdomains, the use of suspicious or misleading words (like “login”, “secure”, or “verify”), and the presence of IP addresses instead of domain names. Other important indicators include entropy (which measures randomness), the ratio of digits to characters, and whether the domain is unusually long or contains typosquatting (i.e., slight misspellings of popular domains). These lexical cues are often used in machine learning models for detecting phishing, malware distribution, or spam URLs, as they provide fast and scalable means of filtering without needing to fetch and analyze full web content.

Domain-related features Domain-related features are essential in URL classification as they provide information about the domain's characteristics, which often signal the legitimacy or malicious intent of a website. These features focus on the properties and history of the domain itself rather than the structure of the URL. Key domain-related features include the domain's age (how long it has been registered), registration validity period (short-lived domains are often used for malicious purposes), and the domain's WHOIS information, such as the registrar name and whether the registrant's details are publicly available or anonymized. Additionally, the

presence of the domain in well-known blacklists or whitelists, and its global or regional popularity (as measured by services like Alexa Rank), can be strong indicators of trustworthiness. Another important aspect is DNS-based features, including the number and type of associated IP addresses or name servers, and how frequently the domain's DNS records change, which can suggest suspicious activity. These domain-level insights help enhance the accuracy of URL classification systems by adding contextual credibility signals that are difficult for attackers to fake consistently over time.

Host and Network features: Host and network features are critical components in URL classification as they provide insights into the infrastructure behind the website, often revealing patterns linked to malicious behavior. These features focus on the hosting environment and the network characteristics of the server that delivers the web content. Common host-related features include the IP address associated with the domain, the number of domains hosted on the same IP (shared hosting can be a red flag), and the geographical location of the server. Network-related features often include latency and connection behavior, such as the round-trip time to the server, packet loss, or abnormal routing paths. Additional indicators can be derived from the use of content delivery networks (CDNs), the presence or absence of secure protocols like HTTPS, and reverse DNS lookup results. Some classification systems also evaluate whether the hosting IP or network is listed in known threat intelligence feeds or blacklists. Collectively, these host and network attributes help uncover infrastructural traits often used by attackers, such as fast-flux hosting, use of compromised servers, or clustering of phishing domains on a single IP range.

URL structure and patterns: URL structure and pattern-based features are fundamental in URL classification, as they help detect suspicious or anomalous formatting often used in phishing, malware, or spam campaigns. These features analyze the composition and format of the entire URL to identify irregularities or patterns that deviate from standard web practices.

Key aspects include the depth of the URL path (number of subdirectories), presence of encoded characters (e.g., "%20", "%3D"), excessive use of parameters and query strings, or long, convoluted URLs designed to obscure intent. Repetition of certain elements, use of misleading directory names (like "/secure/login" in non-authentic sites), or inclusion of misleading file extensions (e.g., ".exe" disguised in web content) are also notable indicators.

Patterns such as the placement of suspicious keywords, use of hexadecimal or base64 encoding, or URLs mimicking trusted domains through subdomain tricks (e.g., "paypal.com.secure-login.info") are strong red flags. These structural anomalies are often hard for legitimate sites to justify and thus serve as reliable cues for detecting malicious intent purely from the URL format itself.

B. Hybrid LR, SVM and DT Voting Machine Learning Architecture and Model Training:

A hybrid machine learning architecture that combines Logistic Regression (LR), Support Vector Machine (SVM), and Decision Tree (DT) through a voting mechanism leverages the strengths of each individual algorithm to improve overall classification performance. In this ensemble setup, each model is trained independently on the same dataset and learns to classify instances based on its unique methodology—Logistic Regression provides probabilistic outputs and handles linear relationships well, SVM excels at finding optimal decision boundaries in high-dimensional spaces, and Decision Trees capture complex, non-linear patterns and feature interactions. After training, the predictions from all three models are combined using a voting strategy, which can be either hard voting (based on majority class labels) or soft voting (based on averaged predicted probabilities). This approach enhances robustness and accuracy, as it reduces the likelihood of misclassification caused by the individual weaknesses of any single model. The hybrid voting ensemble is especially effective in domains like URL classification, where different models may respond differently to various feature types (e.g., lexical vs. domain vs. network features), and combining them ensures better generalization across diverse URL patterns.

The decision tree (DT): - The decision tree classifier (DTC) is a non-parametric method used for classification and regression. The decision tree classifier recursively partitions the given dataset of rows by applying the depth-first greedy method [44] or the breadth-first approach [45] until all data parts relate to an appropriate class. A decision tree classifier structure was created for the root, internal, and leaf nodes. Tree construction was used to classify unknown data.

At each inner node of the tree, the best separation decision is made using impurity measures [46]. The leaves of the tree were created from the class labels in which the data objects were gathered. The DT (decision tree) classification procedure is implemented in two stages: tree building and tree pruning [44]. It is very tasking and computationally fast because the training dataset is frequently traversed. For a single attribute, entropy is mathematically expressed as:

$$E(S) = \sum_{i=1}^{c} -p_i log_2 p_i \quad (1)$$

The Entropy can be numerically expressed for various characteristics as follows:

$$E(T, X) = \sum_{c \epsilon X} p(c) E(c) \quad (2)$$

IG is defined mathematically by:

$$IG(T, X) = E(T) - E(T, X) \quad (3)$$

Support Vector Machine: - A support vector machine (SVM) is a supervised machine learning algorithm defined by a separating hyperplane between different classes. In other words, given the labeled training data (supervised learning), the algorithm outputs an optimal hyperplane that categorizes new test data based on the training data. Support vector machine (SVM) can be used for both classification and regression. However, it is mostly used in classification problems, where it provides the best accuracy between two classes. In this algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features in the dataset), with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyperplane that differentiates the two classes very well, and the classes in this dataset are zero and one that are easily separated by the hyperplane, as this algorithm performs the best for two classes.

$$x.y = x_1 y_1 + x_2 y_2 = \sum_{i=1}^{2} (x_i y_i)$$

NAIVE BAYES: - The naive Bayes classifier is one of the simplest and most effective machine learning classification algorithms, which helps in building a fast machine learning classifier that can make quick predictions from a given dataset. This is a probabilistic classifier, meaning that it is predicted based on the probability of an object. The naive Bayes algorithm is a probabilistic classifier built upon Bayes' theorem as follows:

$$P(A|B) = \frac{(P(B|A) * P(A))}{(P(B))}$$

A is the class and B is the feature vector. The naive Bayes algorithm is a probabilistic classifier built on Bayes' theorem. P (B|A), P(A), and P(B) are the probabilities measured from earlier known instances, such as training data. Classification errors are minimized by selecting a class that maximizes the probability P(A|B) for every occurrence.

IV. Proposed Methodology Architecture

This architecture diagram outlines a hybrid ensemble machine learning framework for phishing URL detection using a combination of Logistic Regression (LR), Support Vector Classifier (SVC), and Decision Tree (DT). Here's a detailed explanation of each step in the pipeline:

A. Preprocessed URL Phishing Dataset

The initial step in the phishing URL detection pipeline involves cleaning and preprocessing the raw dataset to make it suitable for machine learning models. This stage is crucial for improving model accuracy and reducing the impact of noisy or irrelevant information.

Raw URL datasets often contain missing values, duplicate entries, or malformed URLs that can degrade the model's performance. Preprocessing begins by Eliminating duplicate records that can bias the learning process, Removing null, empty, or corrupted entries that may cause errors during training, Standardizing URL formats to ensure consistency and Filtering out non-URL features or unnecessary metadata (e.g., timestamps, comments, or unrelated text) that do not contribute to classification. Since machine learning models cannot work directly on raw text like URLs, each URL must be transformed into a structured numerical format using feature engineering techniques.

B. Canopy Feature Selection

Canopy Feature Selection is a technique that uses a fast and scalable clustering-based approach to identify relevant features for machine learning tasks, particularly in high-dimensional datasets. Originally developed as a pre-processing step for more computationally intensive clustering algorithms, the canopy method groups data into overlapping subsets, or "canopies," using a rough distance metric and two thresholds: a loose threshold (T1) and a tight threshold (T2), where T1 > T2. In the context of feature selection, this approach can be adapted to evaluate and select features based on their contribution to the formation of distinct and meaningful clusters. Features that consistently help differentiate between canopies are considered informative and are retained, while those that do not contribute significantly to clustering are discarded. This method is valued for its efficiency and ability to handle large datasets, as it reduces computational complexity by focusing only on a subset of relevant features. In URL classification and similar applications, Canopy Feature Selection helps improve model performance by reducing noise, enhancing interpretability, and speeding up training without significantly compromising accuracy.

C. Cross Fold Validation

Cross Fold Validation, often referred to as k-Fold Cross Validation, is a robust technique used to evaluate the generalization performance of machine learning models by reducing the risk of overfitting and ensuring that the model performs well on unseen data. In this method, the entire dataset is randomly partitioned into k equal-sized subsets or "folds." The model is trained k times, each time using k-1 folds for training and the remaining one fold for testing. This process ensures that each data point is used exactly once for validation and k-1 times for training. After all k iterations are complete, the evaluation metrics (such as accuracy, precision, or recall) from each fold are averaged to produce a single performance estimate. This technique provides a more reliable measure of model performance compared to a single train-test split, especially when the dataset is small or imbalanced. Cross Fold Validation is widely used in machine learning workflows, including URL classification tasks, to validate model stability and help fine-tune parameters for improved generalization.

D. Data Split: 70% Training, 30% Testing

In machine learning, splitting the dataset into training and testing sets is a fundamental step to evaluate a model's performance on unseen data. A common practice is to allocate 70% of the data for training and the remaining 30% for testing. The training set is used to train the model—this means the algorithm learns patterns, relationships, and features from the labeled data. Once the model has been trained, it is evaluated on the test set, which contains data it has never seen before. This helps to assess how well the model generalizes to new, real-world data. A 70/30 split strikes a good balance: the training set is large enough to allow the model to learn effectively, while the test set is sufficient to provide a reliable estimate of the model's predictive performance. This approach helps in identifying issues like overfitting or underfitting and ensures the model is both accurate and robust.

E. Grid Search

Grid Search is a powerful hyperparameter tuning technique used in machine learning to systematically find the optimal combination of parameters for a given model. Every machine learning algorithm has a set of parameters (called hyperparameters) that control its behavior, such as the depth of a decision tree, the regularization strength in logistic regression, or the kernel type in an SVM. Grid Search works by exhaustively searching through a specified set (or grid) of possible values for these hyperparameters. For each combination, the model is trained and evaluated—typically using cross-validation—to assess its performance. The combination that yields the best performance metric (such as accuracy, precision, recall, or F1-score) is selected as the optimal setting. Though computationally expensive, Grid Search is simple to implement and very effective, especially when combined with cross-validation to ensure that the selected parameters generalize well to unseen data. It plays a key role in improving model accuracy and robustness by fine-tuning the model settings.

F. Ensemble learning

Ensemble learning is a machine learning technique that combines the predictions of multiple models to improve accuracy, reduce variance, and decrease the likelihood of overfitting. The combination of different models, such as Logistic Regression (LR), Support Vector Classifier (SVC), and Decision Trees (DT), helps leverage the strengths of each model while mitigating their weaknesses.

Logistic Regression (LR) is a linear model that works well when there is a linear relationship between the features and the target variable. However, it may not perform well when the decision boundary is complex or

nonlinear. On the other hand, Support Vector Classifier (SVC) is a powerful model that performs well in high-dimensional spaces and is effective when the data is not linearly separable. It works by finding the optimal hyperplane that maximizes the margin between different classes, making it suitable for classification tasks with complex decision boundaries. Decision Trees (DT) are non-linear models that split the data based on feature values to create a tree-like structure, making them interpretable and effective for capturing non-linear relationships. However, they are prone to overfitting, especially with deep trees.

In an ensemble approach, the predictions of these models (LR, SVC, and DT) are combined, typically using methods such as voting (for classification tasks) or averaging (for regression tasks). By aggregating the outputs, ensemble learning increases robustness and generalizes better to unseen data. For instance, while logistic regression may struggle with complex decision boundaries, the decision tree can capture such patterns, and the SVC can handle high-dimensional spaces. When these models are combined, the ensemble model can outperform individual models, offering greater accuracy and stability.

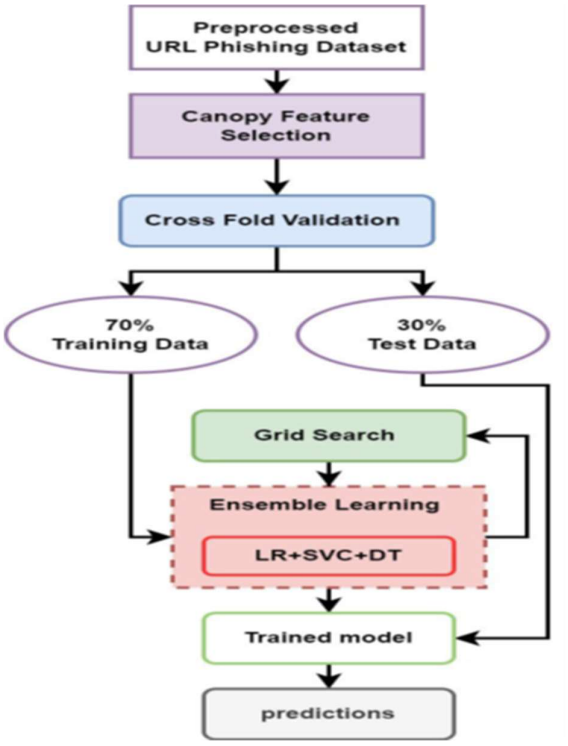

Fig. 1 Proposed approach

V. Evaluation Parameter

Machine learning performance must be evaluated using several evaluation parameters. The machine-learning algorithm provides results in the form of predictions. The evaluation parameters measure the number of true and false predictions made by the model in both legitimate and phishing classes. Parameters such as accuracy, precision, recall, specificity and the F1-score were used. Accuracy measures model performance in terms of the number of accurate predictions made by the model as shown in Equation.

$$Accuracy = \frac{((TP + TN))}{(TP + TN + FP + FN)}$$

Precision is the evaluation parameter that is used for the analysis of the models in which precision identifies the frequency by which a classifier remains correct when we want to predict the positive class. Precision measures the positive rate of the model to the extent to which the model predicts the positive values and indicates the extent to which the model classifies the phishing URLs. The different classifiers performed well in terms of the precision.

$$Precision = \frac{TP}{(TP + FP)}$$

A metric used for the analysis of the classification models that answer out of all the likely positive labels, and how many times did the model accurately identify? To predict phishing and legitimate URLs, the classifier is correctly identified the classifier for both types of URLs.

$$Recall = \frac{TP}{(TP + FN)}$$

The F1 score is the harmonic mean of precision and recall, where the F1 score reaches its best value (perfect precision and recall). The general formula is as follows:

$$F1Score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}$$

VI. Objective of the Study

The objective of studying hybrid machine learning techniques for phishing URL detection is to enhance the accuracy, robustness, and reliability of identifying malicious web addresses by combining the strengths of multiple classifiers. Phishing attacks often employ diverse and evolving tactics to deceive users, making it challenging for any single algorithm to consistently detect them with high precision. By integrating different models—such as Logistic Regression for linear patterns, Support Vector Machines for high-dimensional decision boundaries, and Decision Trees for capturing complex, non-linear interactions—a hybrid approach leverages their complementary capabilities.

The ensemble method, typically through voting mechanisms, allows the system to make more informed and balanced predictions, reducing false positives and negatives. This is particularly valuable in cybersecurity, where misclassifications can lead to significant user harm or loss of sensitive data. Studying hybrid techniques also helps researchers and practitioners understand model synergies, optimize feature handling (e.g., lexical, domain, and host-based features), and design adaptive systems that can better resist the evolving nature of phishing strategies. Ultimately, the goal is to build more intelligent, scalable, and resilient detection frameworks that offer greater protection in real-time web environments.

VII. Conclusion

The study of a hybrid machine learning technique incorporating Decision Tree (DT), Support Vector Machine (SVM), and Naive Bayes for phishing URL detection concludes that combining diverse classifiers significantly enhances the overall detection accuracy and robustness of the system. Each algorithm brings unique strengths: Decision Trees effectively handle non-linear relationships and are easy to interpret, SVM offers strong performance in high-dimensional spaces and excels at defining optimal decision boundaries, while Naive Bayes is fast and performs well with categorical features and probabilistic reasoning. When used in a hybrid ensemble—typically through a voting mechanism—these models complement each other, reducing individual weaknesses such as overfitting or assumptions about data distributions. The ensemble model demonstrates improved generalization and stability across varied phishing patterns, URL structures, and feature sets. This approach ultimately contributes to more reliable and scalable phishing detection systems capable of adapting to new threats. Therefore, hybrid techniques not only increase classification performance but also offer a practical path forward in combating the dynamic and evolving nature of phishing attacks.

References

[1] Kuraku, D.S. and Kalla, D., 2023. Impact of phishing on users with different online browsing hours and spending habits. International Journal of Advanced Research in Computer and Communication Engineering, 12(10).

[2] Nadeem, M., Zahra, S.W., Abbasi, M.N., Arshad, A., Riaz, S. and Ahmed, W., 2023. Phishing attack, its detections and prevention techniques. International Journal of Wireless Security and Networks, 1(2), pp.13-25p.

[3] Ayiku, D., 2023. Comparative Analysis: The increase in phishing activities (Doctoral dissertation, Aalborg University).

[4] Javaid, H.A., 2024. How Artificial Intelligence is Revolutionizing Fraud Detection in Financial Services. Innovative Engineering Sciences Journal, 4(1).
[5] Taofeek, A.O., 2024. Development of a Novel Approach to Phishing Detection Using Machine Learning. ATBU Journal of Science, Technology and Education, 12(2), pp.336-351.
[6] Devan, M., Krothapalli, B. and Shanmugam, L., 2023. Advanced Machine Learning Algorithms for Real-Time Fraud Detection in Investment Banking: A Comprehensive Framework. Cybersecurity and Network Defense Research, 3(1), pp.57-94.
[7] Ofoegbu, K.D.O., Osundare, O.S., Ike, C.S., Fakeyede, O.G. and Ige, A.B., 2024. Real-Time Cybersecurity threat detection using machine learning and big data analytics: A comprehensive approach.
[8] Singhal, S., 2024. Real Time Detection, And Tracking Using Multiple AI Models And Techniques In Cybersecurity. Transactions on Latest Trends in Health Sector, 16(16).
[9] Chirra, B.R., 2023. Advancing Real-Time Malware Detection with Deep Learning for Proactive Threat Mitigation. International Journal of Advanced Engineering Technologies and Innovations, 1(01), pp.274-396.
[10] Adebowale, M.A., Lwin, K.T. and Hossain, M.A., 2023. Intelligent phishing detection scheme using deep learning algorithms. Journal of Enterprise Information Management, 36(3), pp.747-766.
[11] Andriu, A.V., 2023. Adaptive Phishing Detection: Harnessing the Power of Artificial Intelligence for Enhanced Email Security. Romanian Cyber Secur. J, 5(1), pp.3-9.
[12] Balantrapu, S.S., 2023. Evaluating the Effectiveness of Machine Learning in Phishing Detection. International Scientific Journal for Research, 5(5).
[13] Aldakheel, E.A., Zakariah, M., Gashgari, G.A., Almarshad, F.A. and Alzahrani, A.I., 2023. A Deep learning-based innovative technique for phishing detection in modern security with uniform resource locators. Sensors, 23(9), p.4403.
[14] Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S.B. and Joga, S.R.K., 2023. Phishing detection system through hybrid machine learning based on URL. IEEE Access, 11, pp.36805-36822.
[15] Nagy, N., Aljabri, M., Shaahid, A., Ahmed, A.A., Alnasser, F., Almakramy, L., Alhadab, M. and Alfaddagh, S., 2023. Phishing URLs detection using sequential and parallel ML techniques: comparative analysis. Sensors, 23(7), p.3467.
[16] Alazaidah, R., Al-Shaikh, A., Al-Mousa, M.R., Khafajah, H., Samara, G., Alzyoud, M., Al-Shanableh, N. and Almatarneh, S., 2024. Website phishing detection using machine learning techniques. Journal of Statistics Applications & Probability, 13(1), pp.119-129.
[17] Teja Nallamothu, P. and Shais Khan, M., 2023. Machine learning for SPAM detection. Asian Journal of Advances in Research, 6(1), pp.167-179.
[18] Agarwal, S., 2023. An Intelligent Machine Learning Approach for Fraud Detection in Medical Claim Insurance: A Comprehensive Study. Scholars Journal of Engineering and Technology, 11(9), pp.191-200.
[19] Talaei Khoei, T. and Kaabouch, N., 2023. A comparative analysis of supervised and unsupervised models for detecting attacks on the intrusion detection systems. Information, 14(2), p.103.
[20] Tamal, M.A., Islam, M.K., Bhuiyan, T., Sattar, A. and Prince, N.U., 2024. Unveiling suspicious phishing attacks: enhancing detection with an optimal feature vectorization algorithm and supervised machine learning. Frontiers in Computer Science, 6, p.1428013.
[21] Debener, J., Heinke, V. and Kriebel, J., 2023. Detecting insurance fraud using supervised and unsupervised machine learning. Journal of Risk and Insurance, 90(3), pp.743-768.
[22] Abdul Samad, S.R., Balasubaramanian, S., Al-Kaabi, A.S., Sharma, B., Chowdhury, S., Mehbodniya, A., Webber, J.L. and Bostani, A., 2023. Analysis of the performance impact of fine-tuned machine learning model for phishing URL detection. Electronics, 12(7), p.1642.
[23] Calzarossa, M.C., Giudici, P. and Zieni, R., 2024. Explainable machine learning for phishing feature detection. Quality and Reliability Engineering International, 40(1), pp.362-373.
[24] Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S.B. and Joga, S.R.K., 2023. Phishing detection system through hybrid machine learning based on URL. IEEE Access, 11, pp.36805-36822.
[25] Sahingoz, O.K., Buber, E. and Kugu, E., 2024. Dephides: Deep learning based phishing detection system. IEEE Access.
[26] Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q.E.U., Saleem, K. and Faheem, M.H., 2023. A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN. Electronics, 12(1), p.232.
[27] 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.