Malicious URL Detection Model Using Ensemble Learning

*Note: Sub-titles are not captured in Xplore and should not be used

1st Akhil Chedupaka
Dept. of Computer Science Engineering
B V Raju Institute of Technology
Narsapur, Medak, Telangana, India
Akhilchedupaka11@gmail.com

2nd Subhashini Gali
Dept. of Computer Science Engineering
B V Raju Institute of Technology
Narsapur, Medak, Telangana, India
Subhashinigali22@gmail.com

3rd Mohith Reddy Gogi
Dept. of Computer Science Engineering
B V Raju Institute of Technology
Narsapur, Medak, Telangana, India
mohitgogi9@gmail.com

4th V. Pradeep Kumar
Dept. of Computer Science Engineering)
B V Raju Institute of Technology)
Narsapur, Medak, Telangana, India
(pradeepkumar.v@bvrit.ac.in

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—The digital world, especially the offline world, has evolved a lot in recent years; this is important because most of our business is online. This can lead to malicious users and thousands of dollars in lost revenue each year. Identifying malicious websites is important to prevent the spread of malware and to prevent end users from becoming victims. However, current solutions often rely on processing content on the web, which can be problematic but easy for the search engine itself. Using Uniform Resource Locators (URLs) to identify problems is safer and more efficient than identifying content, but inadequate signatures and poor classification can still lead to poor URL access. This research aims to improve the detection accuracy of malicious URL detection by combining two learning levels and design and build a malicious URL detection model based on the initial network of threats. Extract features based on cyber threat intelligence from web searches to improve authentication accuracy. Extract signature-based cyber threat intelligence from web searches to improve detection accuracy. Global cybersecurity analyst and user reports provide important information about malicious websites. Thus, Cyber Threat Intelligence (CTI) based signatures obtained from Google searches and Whois sites can be used to improve searches. Using the Whois website can increase discovery. Furthermore, this study presents a two-stage collaborative learning model that combines the Random Forest (RF) algorithm for prioritization and finally the Multi-Layer Perceptron (MLP) for seating decision. An MLP classifier learned during the study replaced three random forest distributions of the majority voting process in decision making.**

Keywords: maliciousURLs; cyber threat intelligence; ensemble learning; internet security; cybersecurity

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The number of Internet users has increased in recent years, especially due to the growth of mobile devices, the integration of devices into personal life, smart devices and the Internet of Things (IOT). The COVID-19 pandemic has further increased this growth. Today, internet is a fundamentral part of everyday life globally, with businesses increasingly operating online through cloud storage, financial services, and various digital business solutions. However, this extensive use of the internet also exposes users to numerous online threats, including malware, spam, phishing, financial fraud, data theft, and information leaks. Cybercriminals frequently utilize malicious websites to spread malware and achive their objectives. These sites often contain harmful content, such as malware or phishing attacks, which can infect a visitor's device without any user action like clicking or downloading. Among the most prevalent attacks involving malicious URLs are spam and phishing. Phishing involves cybercriminals attempting to obtain sensitive information, such as login details or account information, from users by masquerading as trustworthy entities through emails or counterfeit websites. For example, Figure 1 illustrates a phishing website designed to look identical to twitter, tricking users into entering there credentials.

consistent with statistics, 18.five million web sites are inflamed with malware. also, in step with Google safe surfing file, the wide variety of phishing web sites has improved through 2,800% seeing that September 2010, accomplishing 2 million in September 2020. when the sufferer is going to check the malicious web sites, the attackers use specific techniques to contaminate the user's seek engine with malicious content material or to trick the victim into interacting with the attacker for fraudulent economic or different purposes. Many internet designers do no longer need to be malicious. Attackers can use malicious web sites for malicious functions. as an example, attackers can inject scripts into malicious websites or conduct phishing assaults to scouse borrow touchy facts from site visitors.

Identify applicable funding agency here. If none, delete this.

The challenge of identifying malicious websites dates back to early 2004, prompting numerous proposed solutions for accurate detection. These solutions fall into three investigative categories: URL-based, web content-based, and script-based. URL-based detection, the most extensively studied approach, is followed by content-based detection, while script-based detection has received limited research attention. URL-based detection, the most notably studied technique, is followed by way of content-primarily based detection, at the same time as script-based totally detection has obtained restricted research interest. URL-based totally detection is desired for its proactive and cozy nature, permitting machines to pick out malicious URLs before consumer visits. This method is specifically green for real-time detection and aid-confined packages, inclusive of cellular and internet of factors (IoT) gadgets.

numerous techniques were proposed to discover malicious web sites and malicious content via extracting features from URLs. While most of these methods rely on people using features, some solutions use deep learning to make the process work. These attributes consist of an expansion of domain names, which include web hosting data (e.g., usa call and web hosting provider), private attributes (e.g., .com and .tk), and lexical attributes (e.g., quantity of elements inside the URL). and URL duration). in spite of those efforts, URL-based totally functions are nevertheless at risk of manipulate by using attackers, making them hard to symbolize efficiently. Attackers can use evasion techniques to skip security measures, resulting in signatures being extracted from faulty URLs. therefore, features past the attacker's manipulate have been proven to boom detection accuracy and reduce false positives.

Integration of Cyber Threat Intelligence (CTI) can enhance URL-based capabilities to help increase detection efficiency. This work introduces a malicious URL detection model (CTI-MURLD) as a network threat, using security analysts, user experience, and website reputation as important information. The version has 3 major factors that concentrate on writing specifications, along with URL-based, Whois statistics-primarily based, and CTI functions. facts processing consists of N-gram for video extraction, TF-IDF for illustration, and common facts for function selection. The 1/3 element, classification and decision, uses the RF algorithm with three clustering techniques that combine the output to teach a multilayer perceptronbased classifier. This approach pursuits to efficiently analyze hidden styles from the most impartial selections of random forest people. take a look at results were verified the usage of performance measurements and benchmarks the use of usually common records and display improved overall performance compared to nation requirements. This look at makes the subsequent contributions:

1) A CTI-based model for detecting malicious URLs turned into designed and advanced. Acknowledging the capacity obfuscation of both the URL and internet content, an independent supply of capabilities beyond the attacker's have an impact on changed into critical to reinforce the model's performance. As a result, features extracted from cyber risk intelligence, obtained through Google searches and Whois information, were assimilated as novel information for education the proposed detection version.:

2) 2. The research devised and carried out a model primarily based on ensemble studying, integrating 3 random wooded area-primarily based predictors for pre-detection and characteristic extraction with multilayer perceptron-based classifiers for the ultimate selection. three awesome RF classifiers have been trained the use of numerous function dimensions extracted from URLs, Whois, and Google-based totally CTI. most people vote casting schemes utilized by the trainedRF classifiers had been ultimately substituted with those from the skilled multilayer perceptron-primarily based classifiers to enhance detection accuracy.:

3) Numerous gadget studying algorithms, including deep learning strategies just like the Convolutional Neural network (CNN) version and sequential deep learning version, had been explored. these models had been particularly skilled to distinguish between malicious and benign patterns. The findings indicated that incorporating cyber risk intelligence from Google significantly complements the detection overall performance of malicious websites.: .

The the rest of the manuscript is prepared as follows: phase 2 critiques the associated paintings; section three describes the proposed version;section four explains the experimental layout; section five offers the outcomes with a detailed discussion; phase 6 presents the belief and destiny paintings.

II. Literature Survey

A. Cyber danger Inteligence-based totally Malicious URL Detection version the use of Ensemble

Internet sites are usual of many commercial enterprise sectors but face threats from net tampering and malicious sites. present solutions frequently depend upon context as a feature and therefore run the chance of bewilderment. This report addresses this undertaking through reporting a malicious URL detection model as a community risk. The fact can be brought by using extracting features from internet searches, especially cyber hazard intelligence (CTI) from Google searches and Whois websites. The rank learning model combined with Random woodland (RF) and Multilayer Perceptron (MLP) outperforms the authentic model, increasing the accuracy by means of 7.eight% and reducing the false advantageous charge through 6.7% compared to the primary URL model.

B. Malicious URL Detection based totally on gadget mastering

The escalating chance of network information lack of confidence due to various hacking methods. It emphasizes the importance of detecting malicious URLs and proposes an answer using machine studying, deep getting to know strategies, and massive records technology. However, it does not explicitly offer a literature survey. A literature survey normally includes reviewing existing studies and studies associated with the proposed work, summarizing key findings, and identifying gaps or contributions to the existing information.

C. *Malicious URLs Detection device the using of more desirable Convolution Neural network*

As the World Wide Web has become more prevalent in people's daily lives, cybercrime has increased accordingly, particularly the theft of statistical information and the threat of fraud through Bad URLs. Traditional methods and blacklists are thought to be insufficient in detecting changed and newly created malicious URLs. The answer reportedly reflects the evolution of a convolutional neural network (CNN) capable of recovery. This background information research will include research on cybercrime investigations focusing on strategies for targeting malicious URLs. It evaluates traditional technologies such as blacklists and examines issues surrounding the nature of cyber threats. The survey will then progress through the learning phase of the device, usually a CNN model, to detect negative content on the web. Important factors to keep in mind include the efficiency of the recovery process, accuracy, and speed of detection. The survey will also focus on trends, key trends and the latest trends in combating cybercrime related to malicious URLs.

D. *Malicious URL Detection Using Machine Learning and Deep Learning*

The evolving risks of protecting social statistics as hackers use a variety of techniques demonstrate the urgency of dealing with security vulnerabilities from time to time. Among these threats, the management of Uniform Resource Locators (URLs) plays an important role. Research studies explore the utility of machine learning and understanding in detecting malicious URLs. This project offers a solution based on the needs of URL behavior and properties, combining gadget learning and big data techniques to create discoverability. Experimental results demonstrate the effectiveness of the proposed method in identifying dangerous URLs, revealing its potential to be a satisfactory solution for customers.

III. Existing system

The concept of phishing is a malicious practice where attackers deceive users into sharing personal information by creating fake websites resembling genuine ones. It presents an anti-phishing algorithm called Link Guard, developed by analyzing common characteristics of phishing links. The Link Guard algorithm operates at the end-host level and can detect both known and unknown phishing attacks by focusing on the differences and visual hyperlinks, the use of IP addresses instead of DNS names, malicious encoding tricks, and fake DNS names resembling legitimate ones. Implemented on Windows XP,Link Guard demonstrates effectiveness in detecting phishing attacks with slight false negatives, successfully identifying 195 out of 203 phishing attempts. Some drawbacks or limitations of the LinkGuard anti-phishing algorithm include dependence on characteristics, false positives, a single layer of defense, etc.

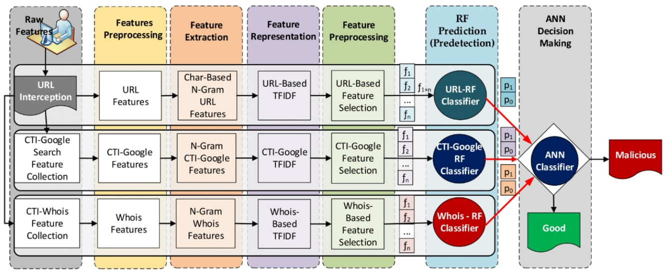

Fig. 1. The proposed CTI-MURLD model.

IV. The Proposed CTI-MURLD Model

The cyber trouble intelligence- grounded vicious URL discovery(CTI- MURLD) model. The CTI- MURLD consists of seven phases data collection, point preprocessing, point birth, point representation, point selection, ensemble literacy-grounded predictors, and decision- timber. In the first six steps, three predictions were generated based on the entire read using the arbitrary tree (RF) algorithm. The bloodsuckers were trained to use Google-based CTI, whois-based functions, and unique URL-based profiles. This RF classifier can do the job. The business representative is correctly recognized, the model is the bad URL (P0 in decision 1), and the business is the detected URL start (p1 in parent 1). Finally, an artificial neural network (ANN) classifier was designed to select the trees. The biased performance of 3 RF classifiers was used to get the ANN classifier for the final selection. As proven in Analysis 1, the URL requested by Junkie was stopped and 3 types of studies were removed. All light points are pre-processed to remove noise. Larger features are also removed using N-gram mode and added in TF-IDF mode. Also, the maximum advisor capacity in each element is listed according to the usage of the received data (represented from f1 to fn). Each item is passed to the expert RF predictor. The negative performance of the predictors (two negative performances for each expert) is fed to the ANN classifier for the final selection of the URL category, whether malicious or safe. The following subsections provide a rough description of each section.

A. *Phase 1: Data Collection Phase*

This includes three types of features: network fault intelligence information accessed through web searches (Google-based CTI), URL content, and statistics about domain owners accessed through Whois searches (Whois-based CTI objects). URL information is collected from HTTP requests of the order subtype. To collect a Google-based CTI, we first extract the domain name from the URL and then look up the IP address associated with that domain name. We then collect statistics for the domain and its IP from Google searches. Whois-based CTIs are extracted from Whois searches, which include information such as website owner, publication date, affiliation, reputation, domain name, and United States domain name.

B. Phase 2: Data Preprocessing

In this phase, the data collected in the previous phase is cleaned and modeled. While URLs are prefixed with the identifier, Google-based CTI profiles are prefixed using a prefix (NLP). This simplifies operations and improves delivery by embedding Google's CTI-based HTML code, tags, and labels on devices and removing all Whois-based CTI data from the site. The stored data is first converted to low-level data and then modeled; this plays two roles in the design process. The first step is to easily convert the test data into training vectors. The second step is to reduce the quality of the vectors by removing unnecessary words and reducing the words in the base text. Normalization first tokenizes the words, then removes the stopwords, lemmatizes them to root words, and finally converts the sentences to their numerical representations. Tokenization turns the report into a list of words that make up the URL profile. Stemming converts words to their base form by removing the ”s” in the plural and the ”ing” in the verb. Lemmatization converts a sentence to its base form by understanding its meaning, including changing ”al” to ”al”.

C. Phase 3: N-gram Feature Extraction

The N-gram system was used to make the features better and produce features that show effects more directly. Numerous inquiries have used N-gram because it helps find bad website links and understand textbooks more. In this study, we used both word N-grams to find features in the URL. For Google-grounded and Whois-grounded cyber trouble intelligence (CTI) data, we used the word N-grams. We broke down the URL data into word vectors, with each word being three, four, or five characters long. To simplify effects, we used the wordbi-gram fashion for the Google- and Whois-grounded CTI data. Each new word added another point. At the end of this process, we had three sets of word commemoratives, which were our point vectors.

D. Phase 4: TF-IDF Feature Representations

To change the words (the tokens) into numbers, we made a list of all the different words and how often they appeared in each group. Then, we used a method called TF-IDF to calculate a numerical value for each word based on this information.

$$tf_i df = tf.logN/df \tag{1}$$

In simple terms, term frequency (tf) measures how often a word appears in a specific sample, while document frequency (df) tells us how many samples contain that word. The more times a word appears in overall (idf), the more important it is to that sample. So, after this step, we end up with three sets of numbers for each sample.

E. Phase 5: Feature Selections

During this stage, we chose the features that best described the URL using a method called information gain. Since the CTI featureGain = 1-E(P)s were gathered from Google, they included a lot of unnecessary information. These extra details made it hard to tell the difference between safe and harmful URLs because there were so many features to consider. This made the learning process difficult and resulted in low accuracy during training. The features that are rare hold more useful information than those that are common. When we choose features based on mutual information, we use entropy to gauge how mixed up they are when splitting the target variable. Entropy measures the level of uncertainty. The higher the entropy, the more unpredictable and informative it is. Mathematically, the entropy is written as:

$$E(p) = -\sum_{i=1}^{n} Pi \log(Pi) \tag{2}$$

In this equation, ’n’ stands for the group we are interested in, and ‘pi’ represents the likelihood of a feature dividing the group into smaller parts. We use this equation to figure out how good the split is in terms of information gain.

Equation:

$$Gain = 1E(p) \tag{3}$$

In this context, ‘n’ represents the main group we are interested in for measuring entropy, and ‘Gain’ tells us how good a split is. When a feature has a high gain, it's considered important for sorting things into groups. The better the gain, the less messy the sorting is. If the sorting is perfectly neat (entropy is zero), it means the split is very clear. At the end of this process, we get a list of features that have high gain, meaning they're the most useful for sorting.

F. Phase 6: RF Ensemble -Based Prediction

At this stage, we use the right RF set for prediction. All experts are trained on members of the industry: URL, Google-CTI and Whois-CTI. We choose a random forest set to generate predictions. We chose RF for two main reasons. First, it is easy to create a truly custom product. 2d holds files of various sizes. Although we have selected a few important factors, we still have 5000 factors of diversity to ensure that we do not miss important statistics and conclude our effect. Random Forest (RF) is a computational tool that works by training several simple decision trees together. Each tree appears as an option to find a good way to provide content. During this process, we create additional bushes that will help the model rise. There are one hundred pruned shrubs in the RF classifier. Each tree is taught to define specific roles and as part of the learning process, three people were injured in the attempt but were isolated. Each of these classifications results in a probability that the model has for something. We average the final result to determine the type of cluster. For each group, each tree in the forest plot is assigned a value between 0 and 1. Numbers above 0 indicate that the sample is less likely to belong to the group, while numbers closer to 1 indicate that it is more likely to join the group. The reach is greater because each tree has a unique profile. Instead of just counting votes, we use these effects as input for all kinds of neural networks

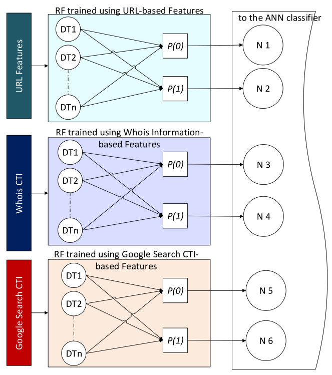
URL Features
RF trained using URL-based Features
DT1
DT2
DTn
P(0)
P(1)
to the ANN classifier
N 1
N 2
Whois CTI
RF trained using Whois Information-based Features
DT1
DT2
DTn
P(0)
P(1)
N 3
N 4
Google Search CTI
RF trained using Google Search CTI-based Features
DT1
DT2
DTn
P(0)
P(1)
N 5
N 6

Fig. 2. The new features extracted from the three ensemble models.

called synthetic neural networks (ANNs). Creating this neural network allows it to make the final choice on the beauty of the version. OK 2 suggests using these sources as new statistics for the next step.

In Figure 2, P(0) shows the typical chance of guessing a safe URL, while P(1) shows the typical chance of guessing a harmful URL. DT stands for a decision tree (a type of weak classifier), and N1-N6 are specific neuron nodes in the following calculations as follows:

Equation:

$$P(0) = \frac{\sum_{i=0}^{n} p(classlabel=0)}{n} \tag{4}$$

Equation:

$$P(1) = \frac{\sum_{i=0}^{n} p(classlabel=1)}{n} \tag{5}$$

In each tree area, "n" indicates how many predictors are there. Usually in Random Forest (RF) methods, these results are mixed by voting and the final choice is made according to the majority opinion. But see, here instead of using the voting method of the RF classifier, we use a special method called multi-layered reliability (MLP) to learn the classification. MLP-based classifiers are trained by combining RF classifiers. It uses this information to teach itself and discover hidden patterns. This model is influenced by the output of the 3 collective models working together. The big picture in this process is defined in the next stage.

G. *Phase 7: ANN Decision Making*

In this step, we also consider the Multilayer Perception (MLP) Artificial Network (ANN). We train a three-layer network with 6 input points, 6 hidden points, and 1 output point. Artificial neural networks are good at predicting even when there is not much data and the problem is very difficult. It learns how different inputs (like x1, x2, x3, etc.) relate to the target we want to find (Y). There are some things that affect how well the neural network works, like how we start the weights and biases, the kind of math we use for activation, how we measure mistakes, the method we use to improve learning, and how many layers and points we have in each layer. The activation math is what decides what comes out of each layer next by adding up the number we put in, multiplied by how important each one is. We use something called a 'loss function' to figure out how many classification mistakes we're making, and then we use an 'optimizer' to try to make fewer mistakes. In this study, we used a specific optimizer called the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm. This algorithm is good at finding the right direction to make things better. It works especially well for certain types of problems where we're trying to minimize errors. When we can't directly figure out how steep the errors are, the BFGS algorithm helps us estimate that. The equation below uses a function called the sigmoid function to activate something.

Equation:

$$segmoid(x) = \frac{1}{1+e^{-x}} \tag{6}$$

In this step, we also consider the Multilayer Perception (MLP) Artificial Network (ANN). We train a three-layer network with 6 input points, 6 hidden points, and 1 output point. Artificial neural networks are good at predicting even when there is not much data and the problem is very difficult.

Briefly, Figure 3 shows how the CTI-MURLD model works. When a URL is retrieved, three types of information are collected: the URL content (such as the domain name and subdomain name), information from Google searches, and Whois information. This content is prepared and developed using some of the techniques described in Sections 3.1 to 3.3. Key points are then selected and fed into our pre-trained RF model for prediction. All models are trained on different elements: CTI-Google, CTI-Whois, or URL features only. Their results are combined into 6 variables and fed into an ANN-based classifier. This classifier provides more accurate detection than just counting votes by understanding the relationship between RF model predictions for class targets. Without this approach, the problem becomes even more complex due to the number of roles involved.
Image : 3

V. Performance Evaluation

This section describes the data used, experimental procedure, and performance evaluation.

A. *Evidence and precedent*

This study uses a database of malicious URLs available at the kaggle.com repository (available at https://www.kaggle .com/sid321axn/malicious - urls-dataset, last accessed May 3, 2024). The data used in this study is collected from well-known sites used in research aimed at detecting malicious URLs. These sources include Phishtank (last visited May 3, 2024) available at https://phishtank.org/ and the URL Dataset (ISCX-URL2016) available at https://www.cic /datasets/url-2016. To achieve this, we selected 20,000 sample URLs for the study, including html, malware links, page defacements,

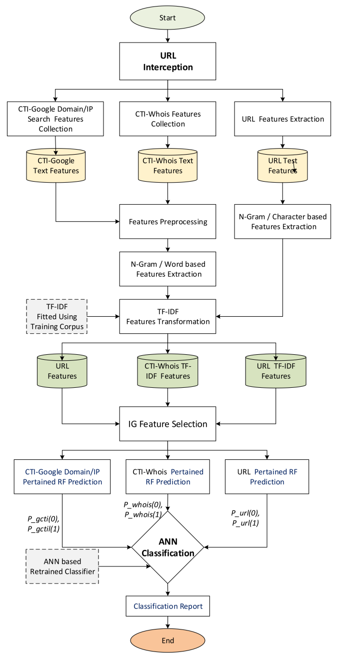
Start
URL Interception
CTI-Google Domain/IP Search Features Collection
CTI-Whois Features Collection
URL Features Extraction
CTI-Google Text Features
CTI-Whois Text Features
URL Test Features
Features Preprocessing
N-Gram / Character based Features Extraction
N-Gram / Word based Features Extraction
TF-IDF Fitted Using Training Corpus
TF-IDF Features Transformation
URL Features
CTI-Whois TF-IDF Features
URL TF-IDF Features
IG Feature Selection
CTI-Google Domain/IP Pertained RF Prediction
CTI-Whois Pertained RF Prediction
URL Pertained RF Prediction
P_gcti(0), P_gctil(1)
P_whois(0), P_whois(1)
P_url(0), P_url(1)
ANN Classification
ANN based Retrained Classifier
Classification Report
End

Fig. 3. Flowchart of the CTI-MURLD model operation

Category	Number of Samples
Total URLs	651,180
Total Benign	428,198
Total Malicious	223,087
Defacement	96,456
Phishing	94,110
Malware Link	32,518

TABLE I
NUMBER AND TYPES OF URLS USED IN THIS STUDY

spam, phishing attempts, etc. available at unb .ca/ (last visited May 3, 2024). Table 1 shows the distribution of different URL models in the literature.

B. Experimental Procedures

The data is divided into two parts: training and testing process, 70% is used for training and 30% for testing. In this framework, Random Forest (RF) and Multilayer Concept/Artificial Neural Network (ML/ANN) classifiers can use training data. The output or predicted values of the RF classifier are used to train the ANN-based classifier.

Each RF classifier has 100 predictors and the ANN prediction model uses the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm as the optimization algorithm. It works as a gradient descent algorithm. A near-zero learning curve is adopted to improve generalization.

Architecturally, a neural network prediction model has three layers: input layer, hidden layer and output layer. Considering the structure of the hidden layer, there are 6 neurons in this layer, while there is a single neuron in the output layer.

C. Performance Evaluation

The text describes the methodology used to evaluate the performance of the proposed malware detection model. Five common performance metrics that are widely used in the current malware field are used: overall accuracy, detection rate, accuracy, f1-score, false positive rate, and not good price. This test uses a tool that checks the impact of URL search, which is commonly used in the tests. The baseline model combines the resources of CTI-MURLD, google-CTI, whois-CTI, and lexical URL-based models to evaluate SDL and CNN-based URL search models. The next stage shows the impact and provides the equation for calculating the general performance indicators used in the tests.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{7}$$

$$FPR = \frac{FP}{TP+FN} \tag{8}$$

$$DR(Recall) = \frac{PT}{TP+FN} \tag{9}$$

$$Precision = \frac{TP}{TP+FN} \tag{10}$$

$$F-measure = \frac{2PRECISIONXRECALL}{PRECISION+RECALL} \tag{11}$$

To provide a comparative analysis, the CTI-MURLD base classifier was trained using machine learning techniques, including deep learning techniques commonly used in malicious web searches. These methods include Naive Bayes (NB), Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), Convolutional Neural Network (CNN), and Sequential Network Model (SDL).

VI. RESULTS AND DISCUSSION

Using the dataset, Cyber Threat Intelligence-Based Malicious URL Detection (CTI-MURLD) model is implemented and its performance metrics are evaluated. It is also compared with the most common tools such as URL-based and Whois-based features. Various configurations are analyzed to evaluate the performance of the proposed CTI-MURLD model. The results include statistical accuracy, negative rate, negative rate, high rate, recovery rate and F1 measure as shown in Table 2 and Figures 4-9. This study compares three machine learning algorithms: logistic regression (LR), decision tree (DT), and random forest (RF) on different products and performance indicators. This test focuses on the accuracy, precision, and memorability of the model using three different features: URL features, WHOIS CTI, and CTI MURLD.

In terms of accuracy, Logistic Regression (LR) demonstrates superior performance, particularly with URL featues, achiving an accuracy of 98.3%, and CTI MURLD features, reaching 98%. Decision trees (DT) exhibit notable performance with WHOIS CTI features, obtaining the highest accuracy of 90%.

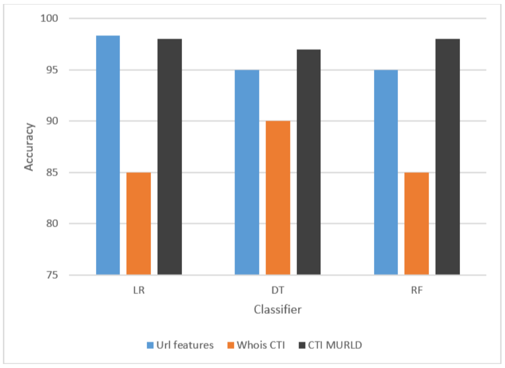

Fig. 4. Comparison in terms of the detection-accuracy performance.

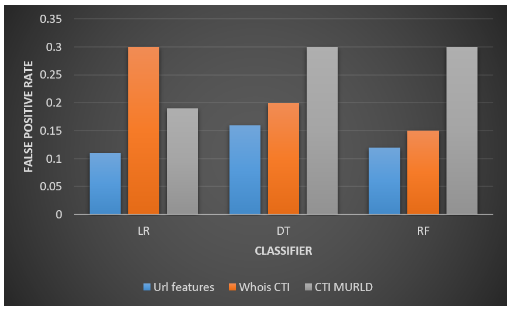

Fig. 5. **Comparison in terms of the FPR**

Random Forest (RF) maintains a consistently high accuracy across all features sets, mirroring LR's performance with 95% for URL features and 98% for CTI MURLD features.

Precision results indicate a broader variability among the models. Decision Trees (DT) achieve the highest precision with URL features (0.16), while Logistic Regression (LR) excels with WHOIS CTI features (0.3). Both Decision Trees(DT) and Random Forest (RF) attain the highest precision (0.3) for CTI MURLD features, suggesting their effectiveness in reducing false positives within this feature set.

When examining recall, which assesses the true positive rate, Decision Trees (DT) outperform with URL features, achieving a perfect recall score of 1. Logistic Regression (LR) and Random Forest(RF) both demonstrate high recall (0.97) for CTI MURLD features, indicating their proficiency in correctly identifying actual positive instances. Random Forest (RF) also leads in recall for WHOIS CTI features with a score of 0.87.

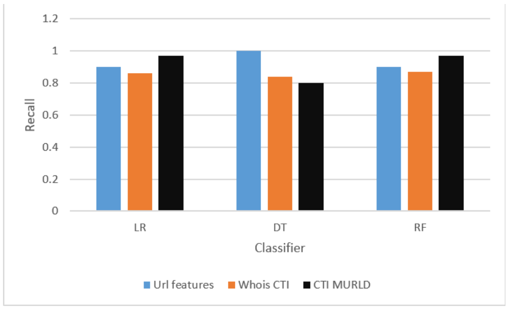

Fig. 6. Comparison in terms of the recall (True Positive Rate)

While Logistic Regression (LR) generally achieves the highest accuracy, Decision Trees (DT) and Random Forest (RF) exhibit significant strengths in precision and recall depending on the feature set. These findings suggest that the optimal choice of model and feature set is contingent upon the specific requirements of the application, such as the need for higher accuracy, precision, or recall. The study underscores the importance of tailoring machine learning models to the nuances of the dataset and the desired outcomes in predictive performance.

VII. Conclusion

In this study, we developed and implemented a model to detect malicious websites using Google's cyber risk intelligence. The main innovation is the use of cyber threat intelligence as a new function, because it can be powerful and secure to increase the accuracy of malicious website detection. Domain names are extracted from Whois generation, Google site risk, and URL potential. We create and use many advanced templates for the book layout, considering the unique characteristics of attack vectors in malicious websites. Three different random forest classifiers were created, each trained on a specific feature: URL-based functionality. The results of the affected products in each tree are combined and used as input to the multilayer perceptron, which replaces our selection method in the forest area. Many tools to understand the classification have been studied to obtain and validate the proposed model. Also, the cost of counterfeit goods is reduced to 2%. Therefore, future research can find green resource-based solutions to solve this problem.

References

[1] R. Indu, M. Bhavya, V. Pardhasaradhi, Y. S. Ram, and D. Y. Suresh, 'MALICIOUS URL DETECTION', vol. 11, no. 4, 2023.
[2] Hyderabad Institute of Technology and Management. Hyderabad and K. Veena, 'Malicious URL Detection Using Machine Learning', IJSREM, vol. 07, no. 04, Apr. 2023, doi: 10.55041/IJSREM18973.
[3] Prof. Rajeshree Sonawale, Tanmay Khedekar, Amar Kamble, and Juhi Kumavat, 'Malicious URL Detection using Machine Learning', IJARSCT, pp. 282–288, Apr. 2023, doi: 10.48175/IJARSCT-9507.
[4] A. A. Syed, 'Malicious URL Detection Using Machine Learning', 2011. [10] N. Venkatesh, V. Tejaswini, G. Soumya, and T. S. Priya, 'MALICIOUS URL DETECTION USING MACHINE LEARNING', 2023.
[5] O. Parab, N. Patil, V. Patil, and D. B. Sarkar, 'Malicious URL Identification UsingMachine Learning - Brosafe', vol. 8, no. 4, 2023.
[6] M. Alsaedi, F. Ghaleb, F. Saeed, J. Ahmad, and M. Alasli, 'Cyber Threat Intelligence-Based Malicious URL Detection Model Using Ensemble Learning', Sensors, vol. 22, no. 9, p. 3373, Apr. 2022, doi: 10.3390/s22093373
[7] 'MALICIOUS URL DETECTION USING MACHINE LEARNING AND DEEP LEARNING', IRJMETS, Sep. 2022, doi: 10.56726/IRJMETS29486.
[8] 'MALICIOUS URL DETECTION USING MACHINE LEARNING AND DEEP LEARNING', IRJMETS, Sep. 2022, doi: 10.56726/IRJMETS29486.

[9] C. D. Xuan, H. Dinh, and T. Victor, ‘Malicious URL Detection based on Machine Learning’, IJACSA, vol. 11, no. 1, 2020, doi: 10.14569/IJACSA.2020.0110119.

[10] M. Aljabri et al., ‘Detecting Malicious URLs Using Machine Learning Techniques: Review and Research Directions’, IEEE Access, vol. 10, pp. 121395–121417, 2022, doi: 10.1109/ACCESS.2022.3222307.

[11] M. T. Varma, P. Zinjad, S. Vast, I. Vohra, and A. H. Sunsara, ‘Malicious URL Detection using ML’, vol. 07, no. 05, 2020.

[12] B. A. Turiaková, ‘Detecting Malicious URLs’, 2019.

[13] D. Sahoo, C. Liu, and S. C. H. Hoi, ‘Malicious URL Detection using Machine Learning: A Survey’. arXiv, Aug. 21, 2019. Accessed: Feb. 29, 2024. [Online]. Available: http://arxiv.org/abs/1701.07179.

[14] F. Douksieh Abdi and L. Wenjuan, ‘Malicious URL Detection Using Convolutional Neural Network’, IJCSEIT, vol. 7, no. 6, pp. 01–08, Dec. 2017, doi: 10.5121/ijcseit.2017.7601.

[15] A. A. Syed, ‘Malicious URL Detection Using Machine Learning’, 2011.