Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2, 1374-1390
2025
Publisher: Learning Gate
DOI: 10.55214/25768484.v9i2.4650

A new intelligent system for malicious URLs detection

Hayder Raeed Hekmat AL-Shawk[1*], Ibrahim M. El-Hasnony[2], Hazem M. El-Bakry[3]
[1,2,3]Dept. of Information Systems, Faculty of Computer and Information,Mansoura University,
Egypt; haideralshuk@gmail.com (H.R.H.A.) Ibrahimhesin2005@mans.edu.eg (I.M.E.H.) elbakry@mans.edu.eg (H.M.E.B.)

Abstract: In cybersecurity, recognizing and mitigating malicious URLs represents paramount challenges due to their various cyber threats, including phishing, malware distribution, and fraud. This paper aims to create a URL detection system that employs machine learning and data mining methods. The proposed system comprises several steps: data acquisition, preprocessing, feature selection, URL tokenization, and classification. First, we acquire a recent dataset containing both malicious URLs and normal ones and 87 numerical features. The features are preprocessed by scaling them using a standard scaler to prevent the model from being biased towards certain features. Furthermore, Fick's Law metaheuristic optimization algorithm (FLA) is used for feature selection, utilizing the Light Gradient Boosting Machines (LGBM) accuracy as a fitness function for the algorithm, resulting in a 50% feature reduction. The URLs are tokenized using Bidirectional Encoder Representations from Transformers (BERT) and converted to a feature vector. The combined BERT feature vector and FLA-selected features are input for the Categorical Boosting (CatBoost) classifier, achieving 96.59% accuracy, 96.75% precision, 96.41% recall, and 96.58% F1-score. The system surpasses all other machine learning and deep learning methodologies in its validation. Additionally, the proposed system outperformed the results of previous studies that utilized the same dataset. The proposed system is an effective and efficient approach for detecting malicious URLs, safeguarding digital assets, and ensuring the integrity of online environments.

Keywords: *BERT, CatBoost, Fick's Law Algorithm, LGBM, Malicious URL Detection.*

1. Introduction

Communication technologies have profoundly influenced the advancement and expansion of businesses across several domains, including social networking, e-commerce, and online banking. Unfortunately, the technology is accompanied by sophisticated methods for scamming and attacking consumers. These assaults use rogue websites that deceive users into revealing crucial information, ultimately resulting in identity or financial theft or the installation of malware on the user's machine [1].

Users usually have insufficient understanding of Uniform Resource Locators (URLs), complicating their ability to ascertain the trustworthiness of web pages. Ad redirection, concealed URLs, alternate URLs, or typographical errors enhance consumers' susceptibility to compromised URL assaults. Offenders create counterfeit websites that resemble legitimate websites and distribute them through spam emails, social media, or Short Message Service (SMS) during these attacks [2].

Many attack strategies exist, including SQL injection, phishing, social engineering, denial-of-service, and man-in-the-middle attacks. The constraints of conventional security management solutions are intensifying due to the significant rise in security risks and the fast evolution of IT technology [3]. Malicious URLs are URLs that have been used for cyberattacks. It is estimated that nearly one-third of all websites are potentially malicious [4] and thus frequently used to commit cybercrimes.

Threats employing harmful URLs encompass Spam, Phishing, Drive-by-Download, and Social Engineering [5].

History: Received: 16 December 2024; Revised: 22 January 2025; Accepted: 24 January 2025; Published: 14 February 2025
* Correspondence: haideralshuk@gmail.com

Drive-by-Download describes the indiscriminate acquisition of malware brought on by a URL visit. These attacks are frequently carried out by taking advantage of flaws in software by inserting malicious code using JavaScript or plugins [6]. Social engineering, phishing, and other tactics trick visitors into divulging private information by impersonating trustworthy websites [7]. Spam refers to the utilization of appealing communications for phishing or promotional purposes.

Google's data indicate that 10,000 websites are banned daily due to malicious behavior [8]. The Phishing Activity Trends Report by the Anti-Phishing Working Group (APWG) indicates that the number of phishing websites recorded in the first quarter of 2022 surpasses one million [9]. The APWG research identifies the industrial sectors most frequently targeted by attackers, as seen in Figure 1. Financial services, including banks, are especially susceptible to phishing, accounting for 23.6% of all assaults. Webmail and Software-As-A-Service providers have been targeted by 20.5%, while fewer assaults (3.8%) were aimed at the logistics and shipping industries [9].

This trend persists, as evidenced by the most recent data from the APWG, which indicates that over 1.2 million distinct phishing attacks were reported in the first half of 2023. Additionally, the threat was persistent and expanding, as evidenced by the 47% increase in phishing attempts in the fourth quarter of 2023 compared to the previous quarter [7]. Statistics from the U.S. government Internet Crime Complaint Center (IC3) for 2022 indicated that exploitation, fraud, and internet-based theft are pervasive, resulting in significant financial losses of $10.3 billion that year. In 2022, the IC3 recorded an astonishing 800,944 complaints about email account compromise (EAC) and business email compromise (BEC) [10]. Consequently, efficient methods for identifying fraudulent URLs can significantly mitigate cybersecurity problems.

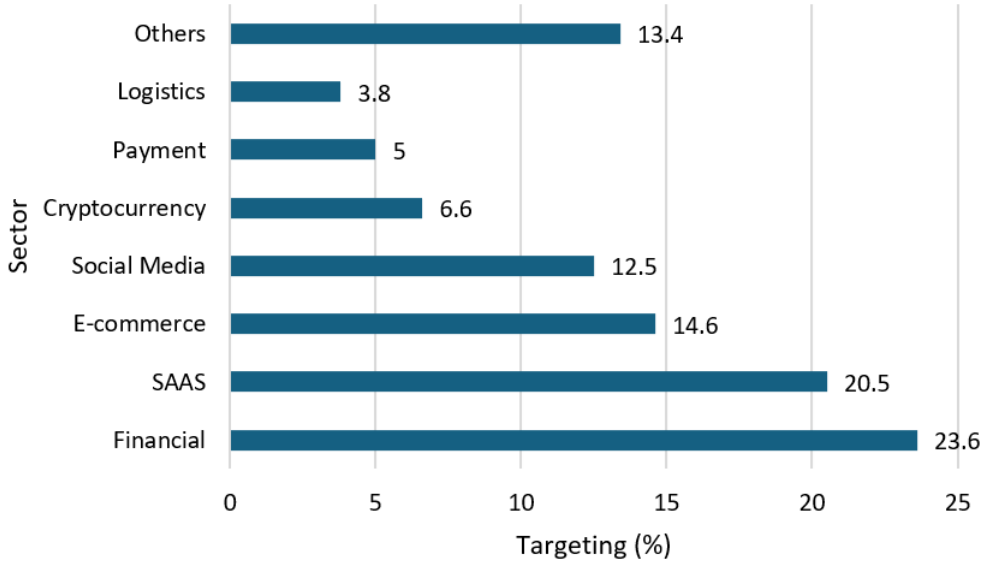

Figure 1.
Targeting percentage of industry sectors by malicious URL attacks.

Maintaining a comprehensive list of harmful URLs is nearly unfeasible, mainly when new URLs are created daily [11]. Attackers use a variety of strategies to get around blacklists, such as fast-flux, which hosts webpages using automatically generated proxies; algorithmic creation of new URLs; and multi-attack execution to change the attack signature and make it invisible to tools that target particular signatures [12].

Thus, traditional methods of URL analysis that rely on static signatures or blacklists, struggle to keep pace with the evolving tactics of cybercriminals. Employing data mining techniques for malicious

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

URL recognition can address this challenge by enabling more efficient and accurate identification of malicious URLs capable of detecting new and previously unseen threats in real-time. This capability can help organizations and individuals prevent cyber-attacks and safeguard sensitive information.

The paper aims to create a system that employs data mining and machine learning methodologies to detect malicious URLs. The study has the following contributions:

- The study developed an accurate URL detection framework utilizing the recent Web Page Phishing Detection (WPPD) dataset incorporating URL text, syntax, structure, web page content, and features from external services.
- Bidirectional Encoder Representations from Transformers (BERT) is used to tokenize URL text and convert it to a feature vector.
- Fick's Law Algorithm (FLA) is used to select the most essential features from other features in the dataset leading to a 50% dataset reduction.
- The BERT tokens and the FLA selected features are input to the Categorical Boosting (CatBoost) classifier which achieved 97% accuracy.
- The system is contrasted with alternative machine learning techniques, deep learning methodologies, and other studies using the same dataset where the proposed system achieved the best results.

The subsequent sections of the paper are structured as follows: The second section comprehensively examines the related work that utilizes the WPPD dataset. The third section explains the proposed system, describing the WPPD dataset. Then, we explain the proposed methodology's different stages, including data preprocessing, feature section, tokenization, and classification. The fourth section demonstrates how the proposed system is evaluated and validates its efficiency against previous studies and machine learning approaches. The last section concludes the study and provides future recommendations on it.

2. Related Work

The related work that involves developing data mining techniques to identify malicious URLs is covered in this section. We emphasize the related work using the WPPD dataset Abdelhakim and Salima [13]. Rani, et al. [14] used the WPPD dataset to build a malicious URL detection model. They first preprocessed the data to check for missing values, impute them with missing values, and label encode the class label column. The dataset was then divided into two parts: a 30% testing set and a 70% training set. Using the tree method, they used SHapley Additive exPlanations (SHAP) to select the highest-ranked 20 features. The selected features are then input for the Random Forest (RF) classifier for training on the training set. The model was tested on the test set and achieved 90.28% accuracy, 89.96% precision, and 91.52% recall.

Lestari and Ulina [15] developed a system to detect phishing attacks using the WPPD dataset. They first preprocessed that data where they normalized it using standard scalar. They then used Analysis of Variance (ANOVA) for feature selection, which selected 52 features out of a total of 87 features in the dataset. The dataset is divided into 20% for testing and 80% for training. The Deep Neural Networks (DNN) model is then employed for detection. Three hidden layers of 128 neurons, an input layer of 52 features, and an output layer of one neuron determine whether the input URL is phishing. In the input and hidden layers, the activation function was the Rectified Linear Unit (ReLU); while sigmoid is used for the output layer. The optimization of neural networks was done using the Adam optimizer. The model has 95.01% accuracy and 95.05% precision rates.

Chantar, et al. [16] used the WPPD dataset to develop a website phishing detection model. A genetic algorithm selects features while naïve Bayes is used for the detection model. Before feature selection, the model achieved 76% precision, 75% recall, 74% F1 score, and 74.8% accuracy. Then, the genetic algorithm resulted in a 60% feature reduction. After using the genetic algorithm to select the best features, the system achieved 92.7% precision, 92.7% recall, 92.7% F1-score, and 92.6% accuracy.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

Agagu, et al. [17] developed a prediction system for phishing websites using the WPPD dataset. Following preprocessing, which includes label encoding the target, the dataset is divided into a 20% testing set and an 80% training set. They used an ensemble voting system combining Categorical Boosting (CatBoost), eXtreme Gradient Boosting (XGBoost), and Light Gradient Boosting (LGBM) for prediction. The proposed system achieved 93% accuracy, 92.2% recall, 93.4% precision, and 92.8% F1 score.

Manala and Jansen van Vuuren [18] used the WPPD dataset to develop a phishing URL detection model. They used random forest feature importance to select the top 20 essential features. They compared multiple classifiers, including decision tree, bagging AdaBoost, XGBoost, and the random forest classifier, where XGBosst performed best. The proposed model using the random forest for feature selection and XGBoost for detection achieved 92.77% accuracy, 92.19% recall, 93.27% precision, and 92.73% F1-score.

Ouellette, et al. [19] developed a crypto-jacking and phishing detection system from URLs using the WPPD dataset. After preprocessing, the dataset was split into a 30% testing set and a 70% training set. They did a comparative analysis between multiple feature selectors, including Correlation feature selection, Minimum Redundancy Maximum Relevance (mRMR), chi-squared, recursive feature elimination, univariate statistical test, permutation feature importance, sequential feature selection, and sequential backward selection. They concluded that mRMR is the best among them. They also compared different classifiers, including extreme randomized trees (extra tree), Adaptive Boosting (AdaBoost), gradient boosting, decision tree, and random forest algorithms, and random forest gave the best results. The proposed system using mRMR for feature selection and random forest for detection achieved 95.65% recall, 95.01% precision, 95.28% accuracy, and 95.33% F1-score.

To facilitate the continuous updating of models by distributed nodes based on streams of fresh phishing data without data accumulation, Revathi [20] presented a phishing detection model that combines federated learning with continual learning. The proposed model used an attention-based classifier with residual connections for detection tasks. On the WPPD dataset, the system achieved 96% recall, 90% precision, 93% accuracy, and 93% F1 score.

Al-Sabbagh, et al. [21] built a phishing detection algorithm on the WPPD dataset. They first preprocessed the data and cleaned it from noise. They then used the Correlation-based Feature Subset Evaluator (CfsSubsetEval) method to select 19 features. Then they transformed the feature space into a polynomial kernel and used K-means clustering two cluster data into two clusters (phishing or not) based on Euclidean distance. The proposed system achieved 89.2% accuracy, 88.5% precision, 90.1% recall and 89.3% F1-score.

Zonyfar, et al. [22] developed a model to classify legitimate websites from phishing ones using the WPPD dataset. They presented a hybrid model combining convolutional neural networks and long short-term memory (HCNN-LSTM) for this detecting task. The model has an input layer, succeeded by a convolutional layer, a max-pooling layer, and an additional convolutional layer. The output is subsequently sent to an LSTM layer of 50 neurons, from which it is transmitted to a fully connected layer employing the ReLU activation function, followed by a dropout layer, and in the end, the output layer utilizing the sigmoid activation function. Training uses 60% of the dataset, whereas testing uses 40%. The proposed approach attained F1-score, recall, and precision, all at 95% and 95.19% accuracy. Table 1 compares the above-discussed studies regarding the feature selection algorithm, the classification algorithm, and the achieved accuracy.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

Table 1.
Related work using the WPPD dataset comparison.

Study	Feature Selection	Algorithm	Accuracy
Rani, et al. [14]	TreeSHAP	Random Forest	90.28%
Zonyfar, et al. [22]	N/A	HCNN-LSTM	95.19%
Al-Sabbagh, et al. [21]	CfsSubsetEval	K-Means	89.2%
Revathi [20]	N/A	Federated-Continual Learning	93%
Lestari and Ulina [15]	ANOVA	DNN	95.01%
Manala and Jansen van Vuuren [18]	Random Forest	XGBoost	92.77%
Agagu, et al. [17]	N/A	CatBoost, XGBoost, and LightGBM	93%
Chantar, et al. [16]	Genetic Algorithm	Naïve Bayes	92.6%
Ouellette, et al. [19]	mRMR	Random Forest	95.28%

3. Materials and Methods

Combining data miming algorithms with intelligent systems have shown good results in different applications. The goal of this study is to develop an intelligent system that can recognize malicious URLs without depending on a blacklist. To achieve this, we use the data mining framework shown in Figure 2.

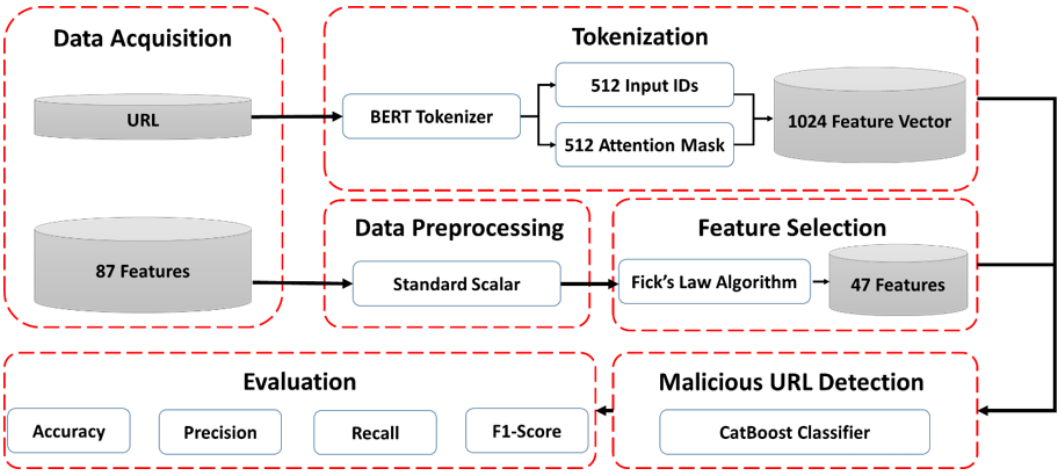

Figure 2.
The proposed system framework.

The framework consists of multiple stages. First, it acquires the WPPD dataset; then, the URLs are tokenized using a BERT tokenizer, which outputs its input ID and attention mask for each token. Each URL has 512 tokens, producing a combined 1024 feature vector. The WPPD dataset also has another 87 URL numerical features scaled using the standard scalar method. The scaled features are input to the FLA which selected 47 features. The selected features and the feature vector produced from the tokenization stage are then input for the classification stage that uses the Cat Boost classifier to be trained to recognize malicious URLs from normal ones. The methodology's effectiveness is validated by evaluating the trained model with unseen URLs to measure various evaluation metrics, including precision, recall, accuracy, and F1-score. Each stage is thoroughly examined in the subsequent subsections.

3.1. Data Acquisition

The WPPD dataset is designed to facilitate the development of machine learning-based phishing detection systems. It includes 11,430 URLs and 87 extracted characteristics. Three distinct categories are the source of features: Fifty-six are derived from the structure and syntax of URLs, twenty-four are

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

obtained from the content of their corresponding sites, and seven are acquired through querying other services. The dataset is equilibrated, comprising precisely 50% phishing and 50% legal URLs, with 5715 instances for each category [23]. The authors gathered authentic web page URLs from Yandex [24] and Alexa [25] whereas phishing URL data were collected from OpenPhish [26] and PhishTank [27]. Figure 3 shows a snapshot from the dataset where "url" is the URL that we store as phasing or not. The "status" column specifies whether the URL is phishing or legitimate. Other columns represent the other 78 features of the dataset.

	url	length_url	length_hostname	ip	nb_dots	...	web_traffic	dns_record	google_index	page_rank	status
0	http://www.crestonwood.com/router.php	37	19	0	3	...	0	1	1	4	legitimate
1	http://shadetreetechnology.com/V4/validation/a...	77	23	1	1	...	0	0	1	2	phishing
2	https://support-appleId.com.secureupdate.duila...	126	50	1	4	...	5828815	0	1	0	phishing
3	http://rgipt.ac.in	18	11	0	2	...	107721	0	0	3	legitimate
4	http://www.iracing.com/tracks/gateway-motorspo...	55	15	0	2	...	8725	0	0	6	legitimate
...	...	...	...	...	...	...	...	...	...	...	...
11425	http://www.fontspace.com/category/blackletter	45	17	0	2	...	3980	0	0	6	legitimate
11426	http://www.budgetbots.com/server.php/Server%20...	84	18	0	5	...	0	0	1	0	phishing
11427	https://www.facebook.com/Interactive-Televisio...	105	16	1	2	...	8	0	1	10	legitimate
11428	http://www.mypublicdomainpictures.com/	38	30	0	2	...	2455493	0	0	4	legitimate
11429	http://174.139.46.123/ap/signin?openid.pape.ma...	477	14	1	24	...	0	1	1	0	phishing

Figure 3.
Snapshot from the WPPD dataset.

The examination of URL text acquires URL-derived attributes. The characteristics of this category are categorized into structural and statistical aspects. Structural-based aspects pertain to fundamental URL components' existence, location, and characteristics (i.e., protocol, top-level domain, and port). The number or distribution of essential URL elements, specific words, or characters within URL text are among the aspects of statistics. These features include the number of dots and subdomains, as well as the length of the word.

Accessing URLs' web pages and looking at their HTML content yields content-based attributes. They are classified into two categories: hyperlinks and abnormal content-based traits. The hyperlink attributes in HTML tags are the number, status, and type of hyperlinks (internal/external). Abnormal content characteristics identify suspicious content or scripts that exhibit questionable behaviors. Examples of dubious content include blank hyperlinks and other domain names within title tags. Examples of questionable behavior include disabling the right-click feature and sending form data to email addresses. Reference third-party services and search engines are utilized to acquire external attributes. Examples include Alexa [25]; WHOIS [28]; Google LLC [29] and OpenPageRank [30].

3.2. Data Preprocessing

The dataset has 74 features of integer values and 13 features of real values. These columns have different mean and standard deviation ranges, which would hinder the learning process, so we need to normalize the data. We used a standard scalar to give the features a unit mean and standard deviation. Standard scaling works for every feature independently, where we calculate the mean as shown in Equation (1).

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (1)$$

where μ is the mean of the feature, N is the total number of rows, and x_i is the feature value. After calculating the mean, we calculate the standard deviation of the feature as in Equation (2).

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad (2)$$

where σ is the standard deviation of the feature, N is the total number of rows, and x_i is the feature value. After calculating the standard deviation, we calculate the feature value after standard scaling as in Equation (3)

$$z_i = \frac{x_i - \mu}{\sigma} \quad (3)$$

where z_i is the transformed x_i after standard scaling. After scaling, the feature values are normalized such that they have uint standard deviation and unit mean. This makes the learning process thereafter more efficient. The standard scalar algorithm can be shown in Algorithm 1.

Algorithm 1: Dataset preprocessing using standard scalar
Input: Dataset Features (F)
Output: Standardized Dataset Features (Z)
Create Z with the same size as F
Foreach feature X in F:
Calculate the mean μ as in Equation (1)
Calculate the standard deviation σ as in Equation (2)
Foreach value x_i in X:
Calculate transformed value z_i as in Equation (3)
Store z_i in the corresponding position in Z
End Foreach
End Foreach
Return Z

3.3. Feature Selection

We implemented the FLA [31] a metaheuristic optimization technique rooted in physics, and employed Fick's rule of diffusion to illustrate the natural migration of molecules from higher concentration to lower concentration regions. The algorithm's fundamental concept is to perceive the problem space as a finite-dimensional space in which the optimal solution is identified by simulating the diffusion of materials. Initially, the randomly generated initial population is divided into two equal groups. The diffusion operator adjusts the population placements during transitions between three stages: 1) exploration, 2) the transition from exploration to exploitation, and 3) exploitation. To determine the updated fitness value, the new location is employed. Once the iteration reaches the termination condition, the global optimal solution is modified per the fitness value. The FLA technique is defined in Algorithm 2.

We employed LGBM accuracy as the criterion for fitness function assessment. LGBM [32] is an implementation of gradient-boosted decision trees (GBDT) [33] an ensemble approach that sequentially combines decision trees (as weak learners) through boosting. Decision trees are integrated so that each subsequent learner addresses the preceding tree's residuals, enhancing the model's performance. The final model consolidates the outcomes from each phase, resulting in the attainment of a robust learner. The FLA selected 47 features out of 87, accounting for around 50% feature reduction.

Algorithm 2: Fick's Law Metaheuristic Algorithm for Feature Selection
Input: Population of selected feature solutions (N), LGBM accuracy function (F), number of it
Output: selected features from the best solution particle
Divide the population into two equal groups N_1, and N_2
Foreach group N_i:
Foreach molecule m_i in N_i:

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

Calculate the fitness function (F)
End Foreach
End Foreach
Find the best molecule in each group
Find the global optimum (m_{best})
For t= 1: T
$TF_t = \sinh\left(\frac{t}{T}\right)^{0.5}$
If ($TF_t < 0.9$)
Apply the *Diffusion state* (Exploration Phase):
Calculate the direction of the flow of the molecules in the exploration state
Determine the number of molecules to travel from region to region
Update molecules' position according to the exploration state
Else If ($TF_t \leq 1$)
Apply the Equilibrium State (Transition Phase):
Calculate the direction of the flow of the molecule in the transition state
Update molecule position according to the transition state
Else
Apply the Steady State (Exploitation Phase):
Update molecule position according to the exploitation state
End If
End for
Return molecule with the best fitness value m_{best}

3.4. Tokenization

The tokenization [34] task is performed on the URL text using the BERT tokenizer [35]. Tokenization divides a continuous text stream into smaller, more significant units, referred to as tokens, which may include words, phrases, or symbols. We used "bert-base-uncased," [36] where the tokenizer first converts the text to lowercase and removes redundant spaces. BERT uses WordPiece [37] as a tokenization scheme. The WordPiece algorithm builds the vocabulary during pretraining from a large corpus based on token frequency. It starts with individual characters and combines them into subwords or words based on co-occurrence. The pre-trained vocabulary includes frequent tokens like "www," ".com," and "/login." Tokens not in the vocabulary are split into smaller subwords or individual characters.

Each sequence starts and ends with the addition of special tokens [CLS] and [SEP], respectively. We employ a maximum sequence length of 512, which truncates sequences that exceed 512 tokens, and the padding of shorter sequences with the special token [PAD]. The tokenizer produces two numbers for each token: the input_id and the attention_mask. The input _id is the id number of the token in the BERT tokenizer vocabulary, while attention_mask is a binary mask indicating whether the token is an actual token or a padding one. The tokenization algorithm based on BERT is shown in Algorithm 3. The BERT tokenizer outputs 1024 features for each URL, of which 512 are input_ids and 512 are attention_masks. In our proposed model, we use the output BERT tokenizer as features combined with the dataset features and then fed to the classification step.

Algorithm 3: BERT-based URL Tokenization
Input: pertained "bert-base-uncased" model vocabulary (V), URLs dataset (D), Sequence maximum length (max_len)
Output: Tokens dataset (TD)
Foreach URL u_i in the dataset:
Lowercase the URL u_i

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

Trim the URL u_i
Split URL into words W based on punctuation
Foreach word in w_i in W
If w_i in V:
Add w_i to tokens list K
Else:
Split w_i into subwords S using the WordPiece algorithm
Add subwords S to tokens list K
End If
End Forecah
If (len(K) < max_len-2):
Add [PAD] tokens until reaching max_len-2
Else If (len(K) > max_len-2)
Truncate tokens after max_len-2
Add [CLS] token at the start of K
Add [SEP] token at the end of K
Append K to TK
End Foreach
Foreach token list K_i in TK:
Create an empty feature list F=[]
Foreach token n_i in token list K_i
Find the corresponding input_id id_i of n_i from V
Append id_i to feature list F
If $n_i$is not [PAD]:
attention_mask $mask_i = 1$
Else
attention_mask $mask_i = 0$
End If
Append Attention mask $mask_i$ to feature list F
End Foreach
Append F to TD
End Foreach
Return TD

3.5. Classification

After merging the output of the feature selection and the tokenization step, create a dataset with a feature vector for each URL consisting of 1024 BERT token features and 47 FLA-selected features. Training and testing sets of the dataset were separated to create a machine-learning classifier. We utilize Cat Boost [38] for classification. Cat Boost is a machine learning classifier used primarily for gradient boosting in decision tree models. Ordered Target Statistics (OTS) and Order Boosting (OB) are used by Cat Boost. OTS and OB implemented random permutations of the training data to mitigate the prediction shift caused by a specific type of target leakage present in all gradient-boosting algorithms currently in use. Cat Boost employs decision trees as its primary predictor.

Cat Boost constructs balanced trees as its foundational predictors, referred to as symmetric trees. Cat Boost's symmetric trees guarantee that every leaf node at the same level has the same decision rule, in contrast to conventional gradient boosting techniques that construct trees leaf-wise or depth-wise. This lowers the possibility of overfitting and speeds up execution. Figure 4 illustrates the Cat Boost process using an input dataset with N samples and X attributes, and we create N trees from the N-ordered boosted samples. Each tree in the ensemble corrects the errors of the previous trees using gradient descent on the

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

loss function. After the training procedure ends, the model is tested where each model's prediction is aggregated and averaged to get the final prediction.

4. Experimental Results

The performance of the proposed approach is illustrated in that section, following a discussion of the performance assessment measures. This section also illustrates how the proposed approach outperformed other machine learning and deep learning methods and previous relevant studies that utilized the same dataset.

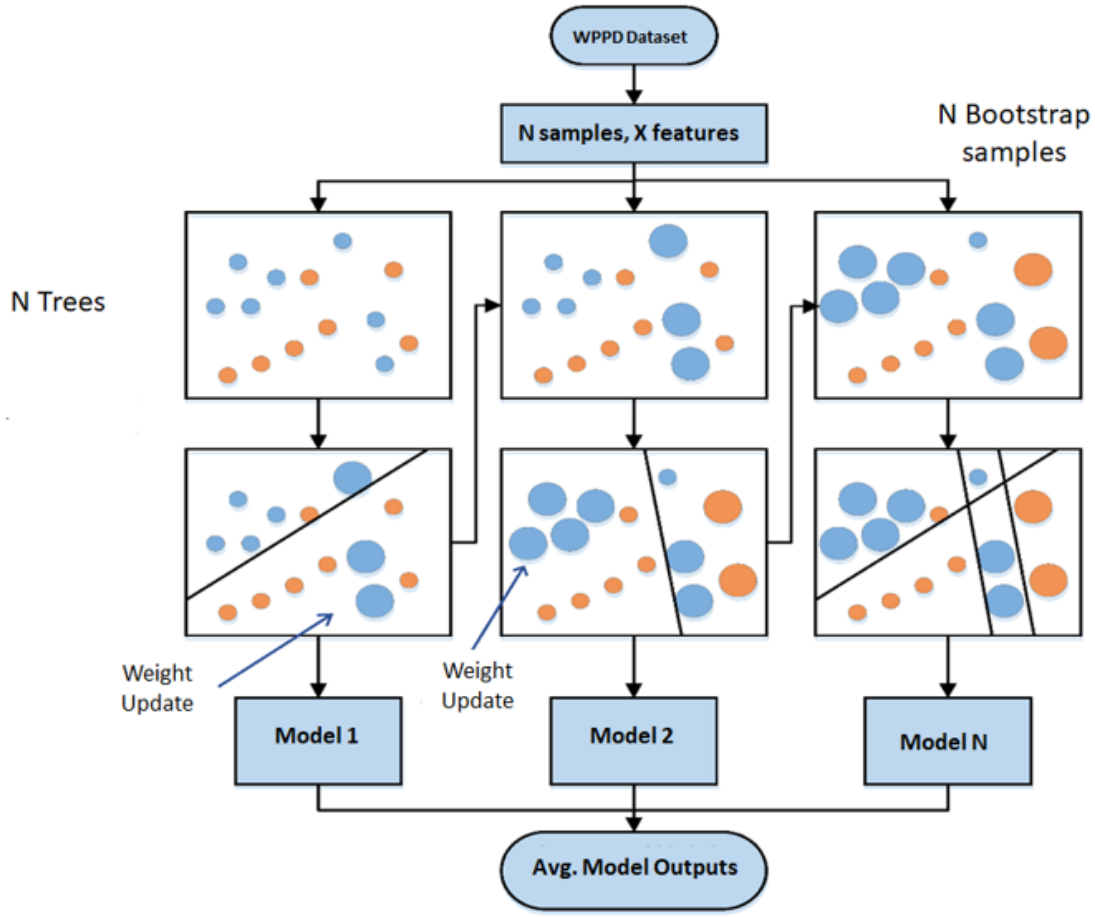

Figure 4.
CatBoost procedure on the WPPD dataset.

4.1. Evaluation Metrics

Four measures are utilized to assess the effectiveness of the suggested system: accuracy, precision, recall, and F1-score. The true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values are computed before calculating these metrics. TP indicates a malicious URL that is correctly predicted as malicious. FP indicates legitimate URLs that are wrongly classified as malicious. TN indicates legitimate URLs correctly classified as legitimate, while FN indicates malicious URLs wrongly classified as legitimate.

The ratio of the true malicious classification to the total number of malicious classifications is known as precision as in Equation (4). A low precision indicates a high number of false positives. According to Equation (5), the recall is the proportion of accurate malicious classifications to all original malicious

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

samples. A high recall indicates a low number of incorrectly identified original samples. The F1-score measures the harmonic mean of recall and precision. This measure, which can be computed using Equation (6), often shows how resilient the classification process is. Equation (7) computes accuracy, the proportion of the model's correctness, by dividing the total number of accurate classifications by the total number of samples.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1_{Score} = \frac{2 * Precison * Recall}{Precision + Recall} \tag{6}$$

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

4.2. Feature Selection Results

Feature selection is performed with the FLA metaheuristic optimization to identify the significant characteristics for malicious URL detection. A population size of 50 molecules and 10 epochs were utilized to establish the termination criterion. The FLA selected 47 features, maximizing the objective LGBM accuracy function to 97.16%. Other metaheuristic algorithms were considered, such as Chernobyl Disaster Optimization (CDO) [39] Energy Valley Optimization (EVO) [40] Coati Optimization Algorithm (CoatiOA) [41] the physical phenomenon of RIME-ice (RIME) [42] and Osprey Optimization Algorithm (OOA) [43]. However, the FLA algorithm outperformed the others in reducing the dataset's dimensionality, as shown in Figure 5.

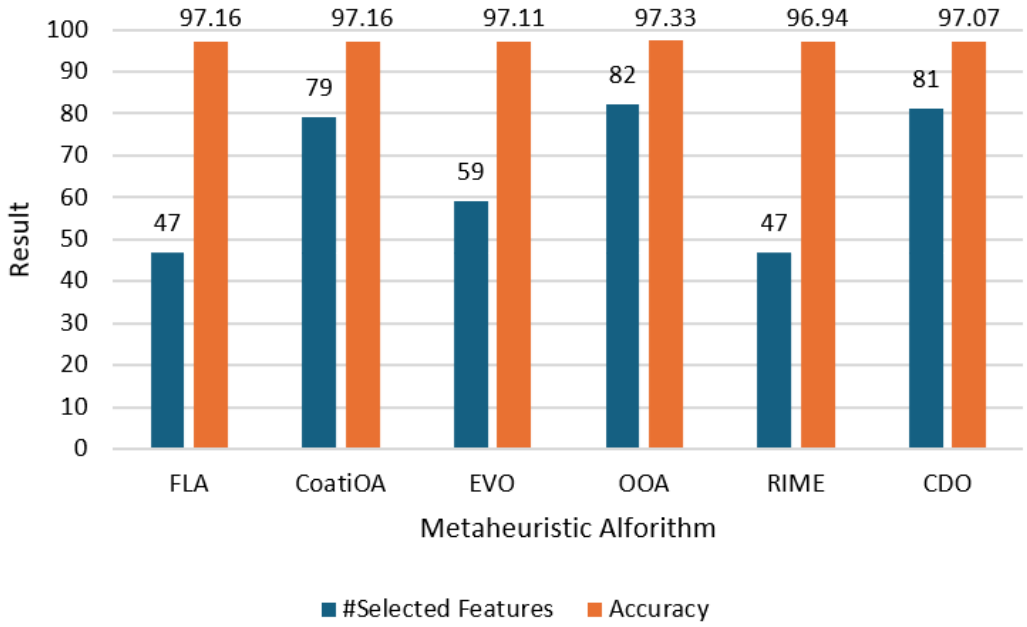

Figure 5.
Comparing FLA to other recent metaheuristic optimization algorithms.

FLA and RIME are the best in decreasing dimensionality by selecting only 47 features, but FLA achieved a higher fitness value than RIME. Also, OOA achieved the best fitness value of 97.33, but it only reduced five features from the original 87 features, accounting for negligible feature reduction. Following OOA, the CoatiOA and the FLA achieved the best fitness value of 97.16, but CoatiOA eliminated only

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

nine features while FLA eliminated 40 features accounting for 50% feature reduction without compromising accuracy, validating choosing it for the feature selection process.

The selected features are 'length_url', 'length_hostname', 'ip', 'nb_dots', 'nb_hyphens', 'nb_or', 'nb_underscore', 'nb_tilde', 'nb_percent', 'nb_slash', 'nb_semicolumn', 'nb_dollar', 'nb_www', 'nb_com', 'nb_dslash', 'http_in_path', 'https_token', 'ratio_digits_url', 'punycode', 'port', 'tld_in_path', 'random_domain', 'shortening_service', 'path_extension', 'nb_redirection', 'length_words_raw', 'char_repeat', 'shortest_word_host', 'longest_word_path', 'avg_words_raw', 'avg_word_host', 'phish_hints', 'statistical_report', 'ratio_extHyperlinks', 'ratio_intRedirection', 'ratio_extRedirection', 'submit_email', 'ratio_intMedia', 'ratio_extMedia', 'iframe', 'popup_window', 'onmouseover', 'domain_in_title', 'domain_registration_length', 'domain_age', 'google_index' and 'page_rank'. The feature distribution of the selected 47 features before normalization is shown in Figure 6. We have 40 discrete integer features from the distribution while having 7 real-valued features. The data is clean and doesn't have null or noisy values.

4.3. Classification Results

The experimental results are obtained using a PC with a Tesla T4 GPU and 32 GB RAM. Table 2 shows the hardware and software specifications used to obtain the experimental results. The proposed system is designed and implemented using Python 3.10.13, PyTorch 2.1.2, sklearn 1.3.2 framework, catboost 1.2, and lightgbm 3.3.2 Python libraries. The dataset is split into an 80% training set and a 20% test set as shown in Table 3.

Table 2.
Hardware and software specification.

Hardware	CPU	Intel(R) Xeon(R) CPU @ 2.00GHz
	GPU	15GB Tesla T4×2
	RAM	32 GB
Software	OS	Ubuntu 22.04
	Language	Python 3.10.13
	Frameworks	PyTorch 2.1.2
		scikit-learn 1.3.2
	Libraries	catboost 1.2
		lightgbm 3.3.2

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

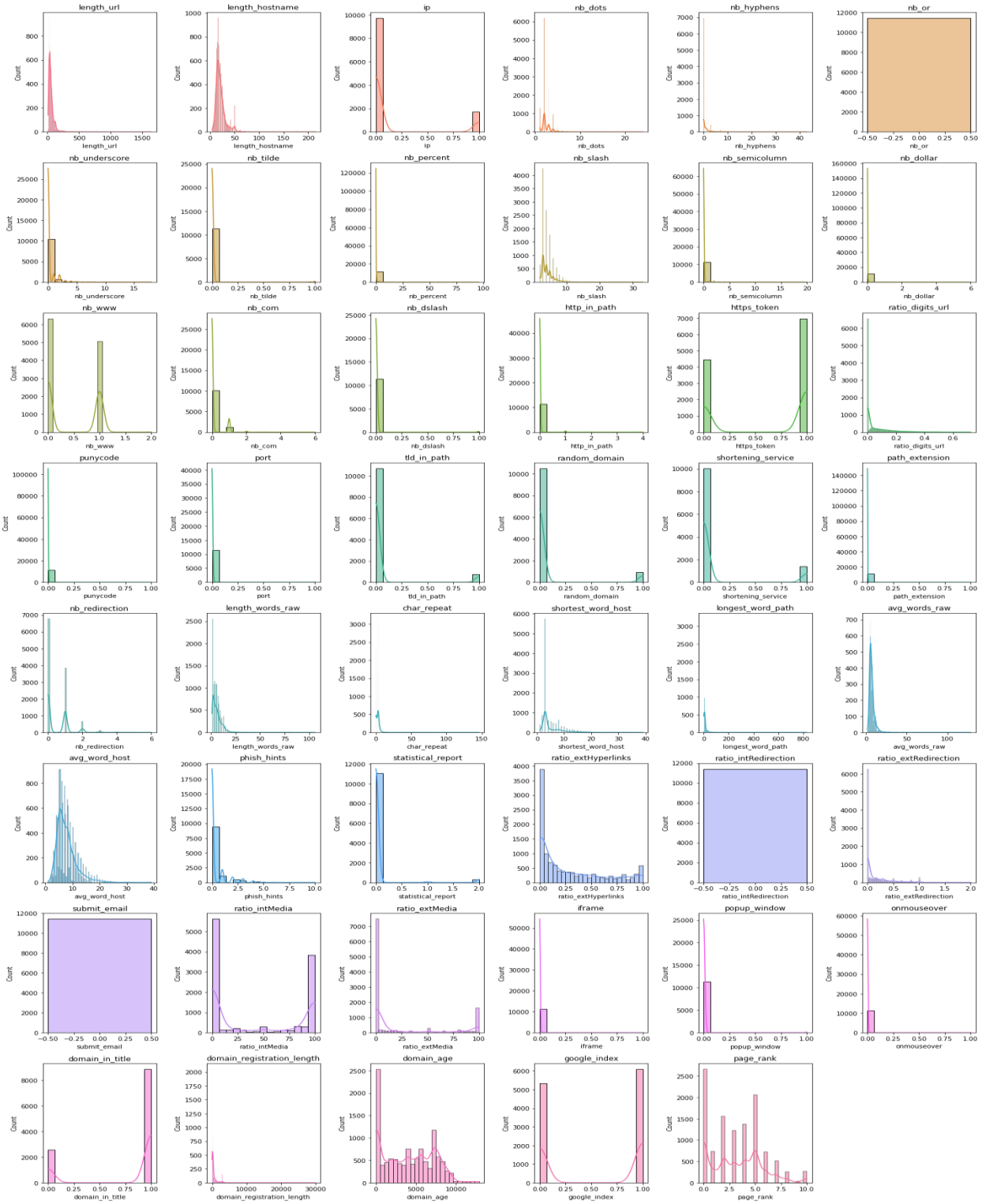

Figure 6.
The 47 FLA selected features distribution.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

Table 3.
Training and testing labels count after data split.

Training labels		Testing labels	
Class	**Count**	**Class**	**Count**
Malicious	4572	Malicious	1143
Legitimate	4572	Legitimate	1143

Experiments were done using the proposed system, which achieved 96.59% accuracy, 96.75% precision, 96.41% recall, and 96.58% F1 score. Experiments were done utilizing other machine learning algorithms, and the proposed system outperformed them, as shown in Table 4. Logistic regression and Adaptive Boosting (AdaBoost) are the best-performing algorithms after the proposed system, while Quadratic Discriminant Analysis is the worst classification algorithm on the WPPD dataset.

Table 4.
Proposed system results against other machine learning algorithms.

Algorithm	Accuracy	Precision	Recall	F1-score
Proposed system	0.9659	0.9675	0.9641	0.9658
AdaBoost [44]	0.937	0.937	0.937	0.937
Support vector machine [45]	0.9256	0.9176	0.9353	0.9263
Logistic regression [46]	0.9374	0.9378	0.937	0.9374
K- nearest neighbor [47]	0.916	0.9288	0.9011	0.9147
Gaussian naive bayes [48]	0.5774	0.9683	0.1601	0.2748
Decision trees [49]	0.9252	0.9248	0.9256	0.9252
Linear discriminant analysis [50]	0.9121	0.9067	0.9186	0.9126
Quadratic discriminant analysis [51]	0.6045	0.965	0.217	0.3543

Also, the system performance is compared with other deep learning architectures, and again, the system outperformed all of them, as shown in Table 5. It shows that when used for classification, BERT is the best-performing deep learning algorithm after the proposed system, reflecting the effect of BERT tokenization on enhancing the proposed system performance. Deep Multilayer Perceptron (MLP) is performing worst among the deep learning methodologies. Furthermore, we compared the proposed system with previous related studies working on the WPPD, as shown in Figure 7, and the proposed system outperformed previous studies.

Table 5.
Proposed system results against other deep learning algorithms.

Algorithm	Accuracy	Precision	Recall	F1-Score
Proposed system	97	97	96	97
Deep MLP [20]	76	78	75	77
Recurrent neural network [20]	88	87	87	87
Convolution neural network [22]	91	91	91	91
Long short term memory [22]	95	95	95	95
BERT [52]	96	96	95	96

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

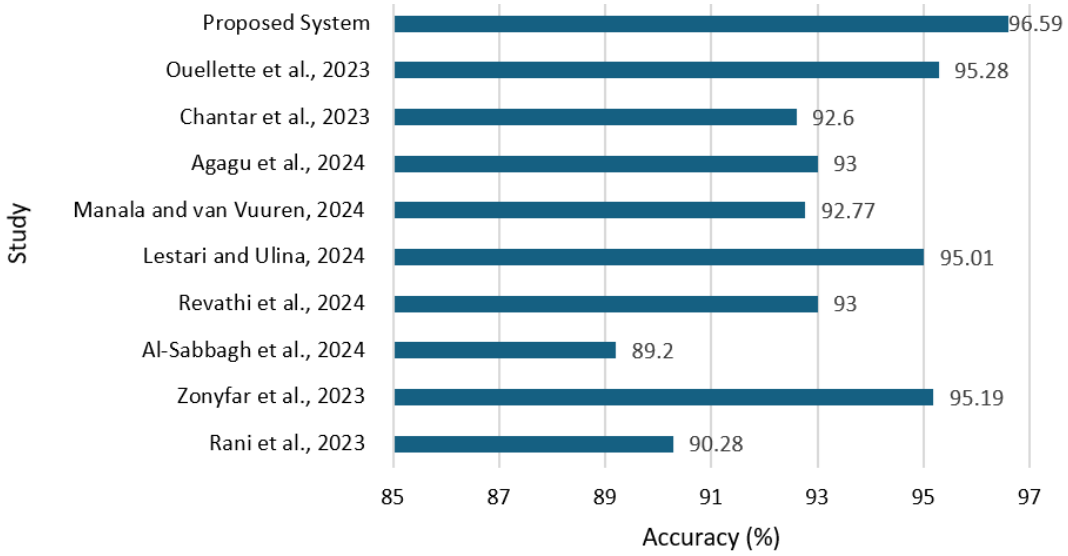

Figure 7.
Comparing the proposed system with previous studies using the same dataset.

5. Conclusion

Identifying malicious URLs poses significant issues owing to diverse cyber threats such as phishing, virus dissemination, and fraud. This study created a URL detection system utilizing data mining and machine learning methodologies. Initially, we acquired a dataset comprising malicious and benign URLs and 87 numerical attributes. The features are preprocessed by scaling using the standard scaler to avert bias in the model towards specific attributes. Additionally, the FLA is employed for feature selection, utilizing the accuracy of LGBM as the objective function, leading to a 50% decrease in features. The URLs are tokenized via BERT tokenizer and transformed into a feature vector. The integrated BERT feature vector and FLA chosen features serve as input for the CatBoost classifier, resulting in 97% accuracy. Alternative deep learning and machine learning methodologies are assessed, and the system outperforms them. The proposed method is also compared to previous studies that employed the same dataset and results surpassing previous findings. Future directions could explore integrating the system with real-time detection frameworks to enhance its applicability in dynamic online environments. Additionally, employing advanced natural language processing techniques, such as generative models, to understand complex URL patterns better may improve detection accuracy. Lastly, extending the system to detect emerging threats like URL obfuscation and adversarial attacks could enhance its robustness.

Transparency:

The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.

Copyright:

References

[1] A. A. Alfalah, "The role of Internet security awareness as a moderating variable on cyber security perception: Learning management system as a case study," *International Journal of Advanced and Applied Sciences*, vol. 10, no. 4, pp. 136-144, 2023. https://doi.org/10.21833/ijaas.2023.04.017

[2] A. K. Jain and B. Gupta, "A survey of phishing attack techniques, defence mechanisms and open research challenges," *Enterprise Information Systems*, vol. 16, no. 4, pp. 527-565, 2022. https://doi.org/10.1080/17517575.2021.1896786

[3] K. L. Snider, R. Shandler, S. Zandani, and D. Canetti, "Cyberattacks, cyber threats, and attitudes toward cybersecurity policies," *Journal of Cybersecurity*, vol. 7, no. 1, p. tyab019, 2021. https://doi.org/10.1093/cybsec/tyab019

[4] B. Liang, J. Huang, F. Liu, D. Wang, D. Dong, and Z. Liang, "Malicious web pages detection based on abnormal visibility recognition," in *2009 International Conference on E-Business and Information System Security*. *https://doi.org/10.1109/EBISS.2009.5138008*, 2009: IEEE, pp. 1-5.

[5] M. Yıldırım, "Using and comparing machine learning techniques for automatic detection of spam website URLs," *NATURENGS*, vol. 3, no. 1, pp. 33-41, 2022. https://doi.org/10.46572/naturengs.1097970

[6] A. Javed, R. Ikwu, P. Burnap, L. Giommoni, and M. L. Williams, "Disrupting drive-by download networks on Twitter," *Social Network Analysis and Mining*, vol. 12, no. 1, p. 117, 2022. https://doi.org/10.1007/s13278-022-00944-2

[7] Z. Alkhalil, C. Hewage, L. Nawaf, and I. Khan, "Phishing attacks: A recent comprehensive study and a new anatomy," *Frontiers in Computer Science*, vol. 3, p. 563060, 2021. https://doi.org/10.3389/fcomp.2021.563060

[8] P. Neelakrishnan, *Data security fundamental requirements. In Autonomous Data Security: Creating a Proactive Enterprise Protection Plan*. Berkeley, CA: Apress, 2024.

[9] APWG, *APWG phishing trends report 2nd Quarter 2022*. Anti-Phishing Work. Gr. https://docs.apwg.org/reports/apwg_trends_report_q2_2022.pdf, 2022.

[10] Federal Bureau of Investigation, "Internet crime report 2022," Fed. Bur. Investig. - Internet Crime Complain. Cent," Retrieved: https://www.ic3.gov/AnnualReport/Reports/2022_IC3Report.pdf. [Accessed 2022.

[11] S. Li, T. Huang, Z. Qin, F. Zhang, and Y. Chang, "Domain Generation Algorithms detection through deep neural network and ensemble," in *Companion Proceedings of The 2019 World Wide Web Conference*. *https://doi.org/10.1145/3308558.3316498*, 2019, pp. 189-196.

[12] D. Sahoo, C. Liu, and S. C. Hoi, "Malicious URL detection using machine learning: A survey. arXiv preprint arXiv:1701.07179," Retrieved: https://arxiv.org/abs/1701.07179. [Accessed 2017.

[13] H. Abdelhakim and Y. Salima, "Web page phishing detection," *Engineering Applications of Artificial Intelligence*, 2021. https://doi.org/10.17632/c2gw7fy2j4.3

[14] L. M. Rani, C. F. M. Foozy, and S. N. B. Mustafa, "Feature selection to enhance phishing website detection based on URL using machine learning techniques," *Journal of Soft Computing and Data Mining*, vol. 4, no. 1, pp. 30-41, 2023. https://doi.org/10.30880/jscdm.2023.04.01.003

[15] W. S. Lestari and M. Ulina, "Optimizing deep neural networks using ANOVA for web phishing detection," *Teknika*, vol. 13, no. 1, pp. 71-76, 2024. https://doi.org/10.34148/teknika.v13i1.758

[16] H. Chantar, S. Ali, and Y. Salem, "Naïve Bayes classifier with genetic algorithm for phishing website detection," in *International Conference for Information and Communication Technologies*, 2023: Springer, pp. 96-108.

[17] M. Agagu, I. A. Ogunbiyi, A. Lasisi, and O. Omorogiuwa, "Detection of phishing websites from URLs using hybrid ensemble-based machine learning technique," in *International Conference on Soft Computing and Data Mining*, 2024: Springer, pp. 11-22.

[18] M. Manala and J. Jansen van Vuuren, "Machine-learning phishing detection model used in the E-banking environment," in *IFIP International Conference on Human Choice and Computers*, 2024: Springer, pp. 69-85.

[19] N. Ouellette, Y. Baseri, and B. Kaur, "I Am bot the 'fish finder': Detecting malware that targets online gaming platform," in *International Congress on Information and Communication Technology*, 2023: Springer, pp. 261-274.

[20] M. Revathi, "Exploring the efficacy of federated-continual learning nodes with attention-based classifier for robust web phishing detection: An empirical investigation," *arXiv preprint arXiv:2405.03537*, 2024. https://doi.org/10.1109/APCI61480.2024.10617245

[21] A. Al-Sabbagh, K. Hamze, S. Khan, and M. Elkhodr, "An enhanced k-means clustering algorithm for phishing attack detections," *Electronics*, vol. 13, no. 18, p. 3677, 2024. https://doi.org/10.3390/electronics13183677

[22] C. Zonyfar, J.-B. Lee, and J.-D. Kim, "HCNN-LSTM: Hybrid convolutional neural network with long short-term memory integrated for legitimate web prediction," *Journal of Web Engineering*, vol. 22, no. 5, pp. 757-782, 2023. https://doi.org/10.13052/jwe1540-9589.2251

[23] A. Hannousse and S. Yahiouche, "Towards benchmark datasets for machine learning based website phishing detection: An experimental study," *Engineering Applications of Artificial Intelligence*, vol. 104, p. 104347, 2021. https://doi.org/10.1016/j.engappai.2021.104347

[24] L. Yandex, "Yandex," Retrieved: https://yandex.com/. [Accessed December 14, 2024], 2024.

[25] A. Alexa, "Alexa website," Retrieved: https://www.alexa.com/. [Accessed Dec. 14, 2024], 2024.

[26] OpenPhish, "OpenPhish FAQ," Retrieved: https://openphish.com/faq.html. [Accessed December 14, 2024], 2024.

[27] PhishTank, "PhishTank," Retrieved: https://www.phishtank.com. [Accessed December 14, 2024], 2021.

[28] WHOIS, "WHOIS service," Retrieved: https://www.domain.com/whois/whois. [Accessed Dec. 14, 2024], 2024.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650

[29] Google LLC, "Google search engine," Retrieved: https://www.google.com/. [Accessed Dec. 14, 2024], 2024.
[30] OpenPageRank, "Openpagerank website," Retrieved: https://www.domcop.com/openpagerank/. [Accessed Dec. 14, 2024], 2024.
[31] F. A. Hashim, R. R. Mostafa, A. G. Hussien, S. Mirjalili, and K. M. Sallam, "Fick's Law Algorithm: A physical law-based algorithm for numerical optimization," *Knowledge-Based Systems*, vol. 260, p. 110146, 2023. https://doi.org/10.1016/j.knosys.2022.110146
[32] G. Ke *et al.*, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, vol. 30, 2017. https://arxiv.org/abs/1712.09913
[33] D. Zhang and Y. Gong, "The comparison of LightGBM and XGBoost coupling factor analysis and prediagnosis of acute liver failure," *IEEE Access*, vol. 8, pp. 220990-221003, 2020. https://doi.org/10.1109/ACCESS.2020.3042848
[34] H. H. Mahmoud, A. M. Hafez, and E. Alabdulkreem, "Language-independent text tokenization using unsupervised deep learning," *Intelligent Automation & Soft Computing*, vol. 35, no. 1, p. 321, 2023. https://doi.org/10.32604/iasc.2023.026235
[35] N. Venkatesan and N. Arulanand, "Implications of tokenizers in BERT model for low-resource Indian language," *Journal of Soft Computing Paradigm*, vol. 4, no. 4, pp. 264-271, 2023. https://doi.org/10.36548/jscp.2022.4.005
[36] M. Geetha and D. K. Renuka, "Improving the performance of aspect based sentiment analysis using fine-tuned Bert Base Uncased model," *International Journal of Intelligent Networks*, vol. 2, pp. 64-69, 2021. https://doi.org/10.1016/j.ijin.2021.06.005
[37] X. Song, A. Salcianu, Y. Song, D. Dopson, and D. Zhou, "Fast wordpiece tokenization," *arXiv preprint arXiv:2012.15524*, 2020. https://doi.org/10.18653/v1/2021.emnlp-main.160
[38] J. T. Hancock and T. M. Khoshgoftaar, "CatBoost for big data: An interdisciplinary review," *Journal of Big Data*, vol. 7, no. 1, p. 94, 2020. https://doi.org/10.1186/s40537-020-00369-8
[39] H. A. Shehadeh, "Chernobyl disaster optimizer (CDO): A novel meta-heuristic method for global optimization," *Neural Computing and Applications*, vol. 35, no. 15, pp. 10733-10749, 2023. https://doi.org/10.1007/s00521-023-08261-1
[40] M. Azizi, U. Aickelin, H. A. Khorshidi, and M. Baghalzadeh Shishehgarkhaneh, "Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization," *Scientific Reports*, vol. 13, no. 1, p. 226, 2023. https://doi.org/10.1038/s41598-022-27344-y
[41] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 259, p. 110011, 2023. https://doi.org/10.1016/j.knosys.2022.110011
[42] H. Su *et al.*, "RIME: A physics-based optimization," *Neurocomputing*, vol. 532, pp. 183-214, 2023. https://doi.org/10.1016/j.neucom.2023.02.010
[43] M. Dehghani and P. Trojovský, "Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems," *Frontiers in Mechanical Engineering*, vol. 8, p. 1126450, 2023. https://doi.org/10.3389/fmech.2022.1126450
[44] Y. Ding, H. Zhu, R. Chen, and R. Li, "An efficient AdaBoost algorithm with the multiple thresholds classification," *Applied Sciences*, vol. 12, no. 12, p. 5872, 2022. https://doi.org/10.3390/app12125872
[45] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121-167, 1998. https://doi.org/10.1023/A:1009715923555
[46] S. Sperandei, "Understanding logistic regression analysis," *Biochemia Medica*, vol. 24, no. 1, pp. 12-18, 2014. https://doi.org/10.11613/BM.2014.003
[47] P. Cunningham and S. J. Delany, "K-Nearest neighbour classifiers: (with Python examples)," *arXiv preprint arXiv:2004.04523*, 2020. https://doi.org/10.1145/3459665
[48] N. G. Ramadhan and F. D. Adhinata, "Sentiment analysis on vaccine COVID-19 using word count and Gaussian Naïve Bayes," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 26, no. 3, pp. 1765-1772, 2022. https://doi.org/10.11591/ijeecs.v26.i3.pp1765-1772
[49] H. Blockeel, L. Devos, B. Frénay, G. Nanfack, and S. Nijssen, "Decision trees: From efficient prediction to responsible AI," *Frontiers in Artificial Intelligence*, vol. 6, p. 1124553, 2023. https://doi.org/10.3389/frai.2023.1124553
[50] M. Melinda, O. Maulisa, N. H. Nabila, Y. Yunidar, and I. K. A. Enriko, "Classification of EEG signal using independent component analysis and discrete wavelet transform based on linear discriminant analysis," *JOIV: International Journal on Informatics Visualization*, vol. 7, no. 3, pp. 830-838, 2023. https://doi.org/10.30630/joiv.7.3.1219
[51] C. Minoccheri, O. Alge, J. Gryak, K. Najarian, and H. Derksen, "Quadratic multilinear discriminant analysis for tensorial data classification," *Algorithms*, vol. 16, no. 2, p. 104, 2023. https://doi.org/10.3390/a16020104
[52] M. Bilal and A. A. Almazroi, "Effectiveness of fine-tuned BERT model in classification of helpful and unhelpful online customer reviews," *Electronic Commerce Research*, vol. 23, no. 4, pp. 2737-2757, 2023. https://doi.org/10.1007/s10660-022-09560-w

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 1374-1390, 2025
DOI: 10.55214/25768484.v9i2.4650