KEC Conference 2025

ISSN: 3021-940X (Print) DOI: https://doi.org/10.3126/injet.v2i2.78614

Surakshit Web: An AI-Powered Phishing URL Detection System

Nirajan Acharya[1, *], Pabitra Rai[2], Rakesh Pandey[3], Sagar Niroula[4], Babu R. Dawadi[5]

[1]*Department of Computer and Electronics Engineering, Kantipur Engineering College, Dhapakhel, Lalitpur, Nepal, nirajan.acharya666@gmail.com*

[4]*Department of Computer and Electronics Engineering, Kantipur Engineering College, Dhapakhel, Lalitpur, Nepal, raipabi101@gmail.com*

[3]*Department of Computer and Electronics Engineering, Kantipur Engineering College, Dhapakhel, Lalitpur, Nepal, rp901522@gmail.com*

[4]*Department of Computer and Electronics Engineering, Kantipur Engineering College, Dhapakhel, Lalitpur, Nepal, sagarniroula112@gmail.com*

[5]*Department of Computer and Electronics Engineering, IOE Pulchowk Campus, Pulchowk, Lalitpur, Nepal, baburd@ioe.edu.np*

Abstract

Phishing attacks have become a significant cybersecurity threat, exploiting users' trust to steal sensitive information through deceptive URLs. Traditional detection methods often fail to keep up with evolving phishing techniques. This research proposes Surakshit Web, an AI-powered phishing URL detection system leveraging deep learning and optimization techniques. The system integrates Convolutional Neural Networks (CNNs) and a Genetic Algorithm (GA)-optimized Multi-Layer Perceptron (MLP) to enhance detection accuracy. A comprehensive dataset of malicious and benign URLs was used, incorporating advanced feature extraction and data pre-processing techniques to improve model performance. The results indicate that the GA-optimized MLP outperforms CNN, achieving a 96.12% accuracy compared to CNN's 93.78%, demonstrating its effectiveness in identifying phishing URLs. This research highlights the potential of evolutionary algorithms in optimizing deep learning models for cybersecurity applications.

Keywords: Phishing attacks, URL detection, AI, MLP, Deep learning, CNN, Genetic Algorithm

1. Introduction

In the modern digital age, the widespread adoption of the Internet has revolutionized communication, commerce, and personal interactions, leading to a significant surge in online transactions and the sharing of sensitive personal information. However, this digital transformation has also opened the door to a new breed of criminal activity known as cybercrime. Among the most prevalent and damaging tactics employed by cybercriminals is phishing, a technique designed to deceive individuals into revealing confidential information by impersonating trusted entities. Phishing attacks encompass various deceptive methods, including vishing, spear phishing, whaling, and email phishing, all aiming to exploit unsuspecting users for malicious gain (Sk. Hasane Ahammad, 2022)**.**

The global COVID-19 pandemic in 2020 resulted in an unprecedented surge in Internet usage, leading to a sharp increase in cybercrimes, particularly phishing. The FBI reported a near doubling of phishing incidents from 114,702 in 2019 to 241,342 in 2020. This surge was attributed to cybercriminals exploiting the pandemic's widespread fear and uncertainty, with phishing attacks accounting for 22% of data breaches in 2020 according to the Verizon Data Breach Investigation Report. The Anti-Phishing Work Group also reported a significant rise in phishing attacks, with many targeting pandemic-related themes, such as fake job offers and fraudulent health organization communications (Arathi Krishna V, 2021).

Deep learning has emerged as a powerful tool for detecting phishing URLs due to its ability to analyze complex patterns and features from large datasets. Several studies have proposed deep learning-based systems

**Corresponding Author*

for phishing URL detection. (Dutta, 2021) employed recurrent neural networks for phishing URL detection, highlighting the superior performance of such advanced techniques compared to previous methods.

With the rise of sophisticated cyber threats, researchers have increasingly turned to deep learning for phishing detection. Various architectures, including deep neural networks (DNNs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs) such as long short-term memory (LSTM), have been widely explored for their ability to analyze complex patterns in phishing attacks. However, challenges remain, particularly in the need for larger public datasets and the development of real time detection models to enhance practical deployment and effectiveness (Cagatay Catal, 2022).

To counter this growing threat, this research explores developing an advanced phishing detection system using deep learning techniques. The approach compares two models: CNN and Customized Multi-Layer Perceptron (MLP), to accurately classify URLs. Additionally, genetic algorithm is applied to enhance performance of MLP, aiming to improve detection accuracy.

The remainder of this paper is organized as follows: Section 2 provides a review of related works, discussing existing phishing detection techniques, including traditional machine learning and deep learning approaches. Section 3 describes the proposed methodology, including dataset details, feature extraction, and the architecture of the Multi-Layer Perceptron (MLP) optimized with a Genetic Algorithm (GA), along with a comparative analysis of Convolutional Neural Networks (CNNs). Section 4 presents the experimental setup and evaluation metrics, followed by the results and discussion in Section 5, where model performance and classification effectiveness are analyzed. Finally, Section 6 concludes the study by summarizing key findings and suggesting potential directions for future research.

2. Literature Review

2.1. Related Works

The researchers in in (Sk. Hasane Ahammad, 2022) highlight key feature extraction parameters such as domain name, URL length, and statistical indicators, emphasizing domain-based features like web traffic and age for distinguishing URLs. The study evaluates machine learning models, including Decision Trees, Random Forest, Logistic Regression, and SVM, concluding that Light GBM performs best. Future recommendations include integrating more features, updating URL data, and developing autonomous phishing detection frameworks.

(Sanghyeop Lee, 2021) explores using genetic algorithms to optimize CNN architectures for Alzheimer's diagnosis via PET/CT images. The approach enhances CNN structures and hyperparameters, achieving 81.74% accuracy with a 14-layer model, outperforming traditional methods in efficiency and precision.

In (Md. Nahiduzzaman, 2019), Multi-Layer Perceptron (MLP) is applied to heart disease classification using the Cleveland heart disease dataset. The model effectively captures complex patterns, handling both binary and multi-class classification, improving diagnostic accuracy through optimized training and generalization.

(Youness Mourtaji, 2021) presents a hybrid phishing detection approach combining lexical, content, and visual feature extraction with machine learning models like CART and SVM, alongside CNN and MLP. The study stresses the need for adaptive learning techniques and dynamic rule-based systems to counter evolving phishing tactics.

2.2. Research Gap

Based on the insights derived from our research findings, the development of a Phishing URL Detection System is proposed to accurately identify malicious URLs in real time. This system leverages the strengths of deep learning and optimization techniques to enhance detection performance.

The core architecture of the proposed system is built upon a Customized Multi-Layer Perceptron (MLP) Neural Network, which is further optimized using a Genetic Algorithm (GA). This hybrid approach ensures improved learning capabilities and model efficiency.

While prior studies have explored RNNs, LSTMs, and classical machine learning models such as Light GBM and Decision Trees for phishing detection, this research focuses on deep learning-based approaches optimized for structured feature extraction. CNNs and MLPs were selected due to their effectiveness in recognizing URL patterns without requiring sequential dependencies, ensuring computational efficiency and real time applicability.

2.2.1. Key Components of the System

2.2.1.1. Feature Extraction

This module is the foundation of the detection process, focusing on extracting relevant features from URLs. Features extracted based on the research are:

Table 1. Extracted Features

URL length	Label	Result	Contains IP
Hostname length	Path length	FD length	Count of '@'
Count of '?'	Count of '%'	Count of '='	Count of '-'
Count of '#'	Count of '&'	Count of '+'	Count of '$'
Count of 'www'	Count of digits	Domain name length	Count of directories
Count of letters	HTTP(S) double slash	Hyphen	Shortening service
Count of '.'	Url		

By employing advanced feature extraction techniques, the system ensures a comprehensive understanding of URL characteristics, laying the groundwork for effective classification.

2.2.1.2. CNN

A Convolutional Neural Network (CNN) leverages layers of neurons to process structured data, particularly useful in pattern recognition. It extracts hierarchical features from the input data through convolutional layers, pooling layers, and fully connected layers. This structure enhances the model's ability to capture spatial patterns in URLs (Rikiya Yamashita, 2018).

2.2.1.3. Customized MLP

The MLP acts as the primary classifier, customized to handle URL-specific features. With additional layers, neurons, and activation functions, this model precisely learns intricate patterns, improving the detection accuracy of phishing URLs (Md. Nahiduzzaman, 2019).

2.2.1.4. Genetic Algorithm for Optimization

The genetic algorithm (GA) optimizes the system by simulating natural selection. It follows population initialization, fitness assessment, selection, crossover, and mutation. This iterative process refines the feature subset and enhances model performance by evolving the best solutions over generations (Serhii Lienkov, 2022).

3. Methodology

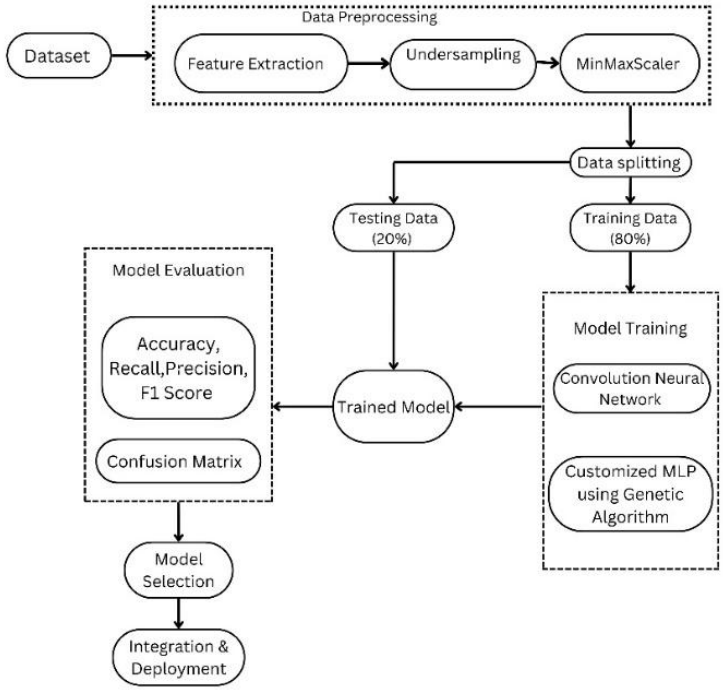

Figure 1. Working Mechanism

Figure 1 depicts a systematic workflow to detect phishing URLs, starting with comprehensive data preprocessing. Key features such as URL length, hostname length, and other engineered metrics have been extracted to enhance the model's predictive capabilities. To address class imbalance, undersampling has been applied, ensuring equal representation of phishing and legitimate URLs. The data has then been scaled using a Minmax Scaler to normalize feature values and improve model training. Finally, the processed data has been split into training (80%) and testing (20%) subsets. The training subset is further divided into 75% to 25% ratio for training and validation data during model training.

3.1. Dataset

The dataset consists of 544,287 entries, combining two publicly available Kaggle sources. The first dataset (https://www.kaggle.com/datasets/siddharthkumar25/malicious-and-benign-urls?resource=download) primarily contains benign URLs, while the second (https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset?resource=download) focuses on malicious URLs. To ensure balance, additional malicious samples were incorporated, resulting in 345,738 benign and 198,549 malicious entries.

3.2. Feature Extraction and Pre-processing

A total of 23 lexical and structural features were extracted to differentiate between benign and phishing URLs. These features include:

a. Structural Features: URL length, hostname length, path length, domain name length, first directory length, directory count.
b. Lexical Features: Presence of special characters (@, ?, %, =, -, ., #, &, +, $), count of numeric digits, letter count, hyphenation, and frequency of subdomains.
c. Binary Features: Presence of an IP address in the URL, detection of URL shortening services, verification of HTTP(S) double slashes.

To address the class imbalance, random undersampling was applied, ensuring an equal distribution of benign and phishing URLs. The dataset was then normalized using Min-Max Scaling, transforming feature values between 0 and 1 for improved model convergence.

3.3. Model Development

3.3.1. Convolutional Neural Network (CNN)

A 1D CNN architecture was designed to learn feature representations.

Table 2. CNN Architecture

Layer (type)	Output Shape	Param #
conv1d_16 (Conv1D)	(None, 21, 32)	128
max_pooling1d_16 (MaxPooling1D)	(None, 10, 32)	0
dropout_21 (Dropout)	(None, 10, 32)	0
conv1d_17 (Conv1D)	(None, 8, 64)	6,208
max_pooling1d_17 (MaxPooling1D)	(None, 4, 64)	0
dropout_22 (Dropout)	(None, 4, 64)	0
flatten_8 (Flatten)	(None, 256)	0
dense_20 (Dense)	(None, 120)	30,840
dense_21 (Dense)	(None, 80)	9,680
dense_22 (Dense)	(None, 2)	162

The architecture consists of:

i. Two Conv1D layers with 32 and 64 filters, using a kernel size of 3.
ii. MaxPooling layers to reduce dimensionality.
iii. Dropout layers to prevent overfitting.
iv. Flatten layer to convert feature maps into a dense representation.
v. Fully connected (Dense) layers with ReLU activation.
vi. Output layer using Softmax activation for binary classification.

CNN model was trained using the Adam optimizer with a learning rate of 0.001, binary cross entropy loss, and batch size of 32 for 20 epochs.

3.3.2. Multi-Layer Perceptron (MLP) with Genetic Algorithm (GA) Optimization

A customized MLP model was developed, with its hyperparameters optimized using Genetic Algorithm (GA).

Table 3. MLP Model Architecture

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 30)	720
dense_1 (Dense)	(None, 29)	899
dense_2 (Dense)	(None, 1)	30

Table 3 shows MLP (Multilayer Perceptron) model architecture consisting of three fully connected (Dense) layers. The first dense layer has 30 output units with 720 parameters, indicating it receives input features of size 24. The second dense layer has 29 units and 899 parameters, suggesting a connection from the previous 30-unit layer. Finally, the output layer has a single unit with 30 parameters, suitable for binary or regression tasks.

The GA was implemented as follows:

1. Population Initialization: A population of 10 MLP models was generated, each with random hyperparameters.
2. Fitness Evaluation: Each model was trained and evaluated based on validation accuracy.
3. Selection: The top-performing models (50%) were chosen for breeding.
4. Crossover: New offspring models were generated by recombining hyperparameters from selected parents.
5. Mutation: Random alterations were applied to neurons per layer and learning rate to introduce diversity.
6. Evolution Over Generations: This process was repeated for five generations, iteratively refining the hyperparameters.

4. Results and Analysis

4.1. Dataset Visualization

Table 4. Dataset Statistics

#	Column	Non-Null Count	Dtype
1	contains_ip	544287	int64
2	hostname_length	544287	int64
3	path_length	544287	int64
4	fd_length	544287	int64
5	url_length	544287	int64
6	count@	544287	int64
7	count?	544287	int64
8	count%	544287	int64
9	count=	544287	int64
10	count-	544287	int64
11	count.	544287	int64
12	count#	544287	int64
13	count&	544287	int64
14	count+	544287	int64
15	count$	544287	int64
16	count-www	544287	int64
17	count-digits	544287	int64
18	domain_name_length	544287	int64
19	count_dir	544287	int64
20	count-letters	544287	int64
21	http(s)-double_slash	544287	int64
22	hyphen	544287	int64
23	shortening_service	544287	int64

Table 4 shows dataset statistics. It contains 544,287 entries and 23 columns. It includes a mix of URL-related extracted features, with 23 integer (int64) columns. Each column name represents a specific characteristic of the URLs, such as length attributes (hostname length, path length, url length), character counts (count@,

count?, count%), and security-related indicators (contains_ip, shortening_service). The Non-Null Count indicates that there are no missing values in any column.

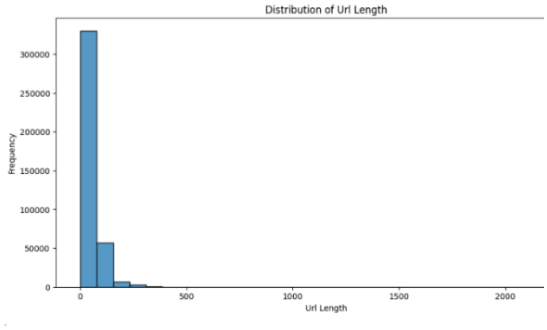

Figure 2. URL Length

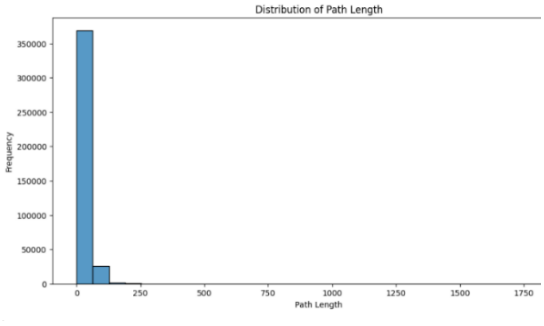

Figure 3. Path Length

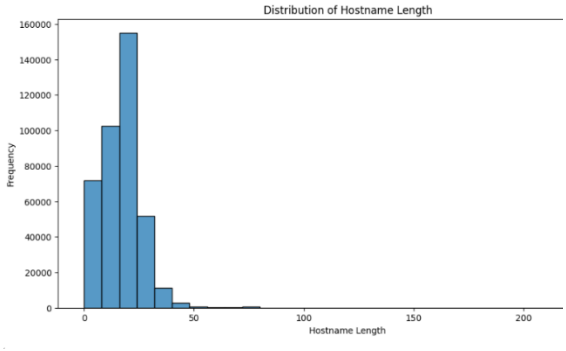

Figure 4. Hostname Length

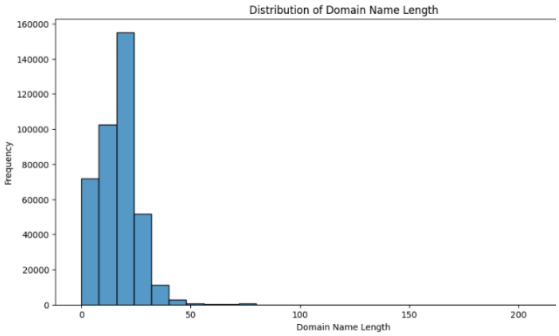

Figure 5. Domain Name Length

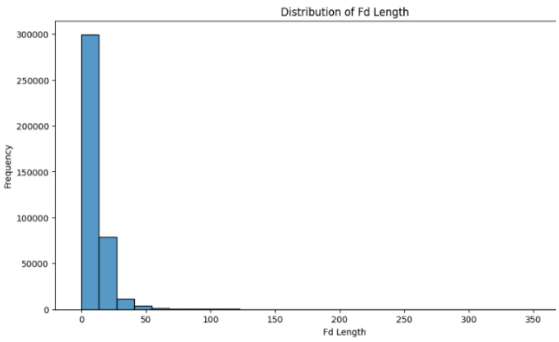

Figure 6. FD Length

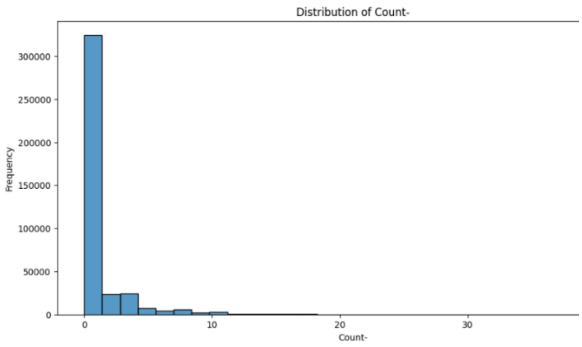

Figure 7. Count Dash

Figure 2 shows that shorter URLs are significantly more common, with most under 500 characters and a peak between 0–100 characters. The frequency declines sharply beyond 500, with almost none exceeding 1000. Figure 3 reveals that path lengths are predominantly short, with most below 250 characters and concentrated between 0–100. The frequency drops as length increases. Figure 4 illustrates that hostname lengths are mostly below 50 characters, rarely exceeding 100. The count rises gradually up to 50 before declining sharply.

Figure 5 indicates that most domain names are under 50 characters, with a peak in the shortest range. Domains over 50 characters are uncommon, highlighting a preference for brevity. Figure 6 depicts the distribution of "Fd Length," where shorter values dominate. The frequency drops sharply as length increases, resulting in a right-skewed distribution. Figure 7 shows the distribution of hyphen counts in URLs. Most contain few or no hyphens, with frequency declining sharply for higher counts, indicating that URLs with many hyphens are rare.

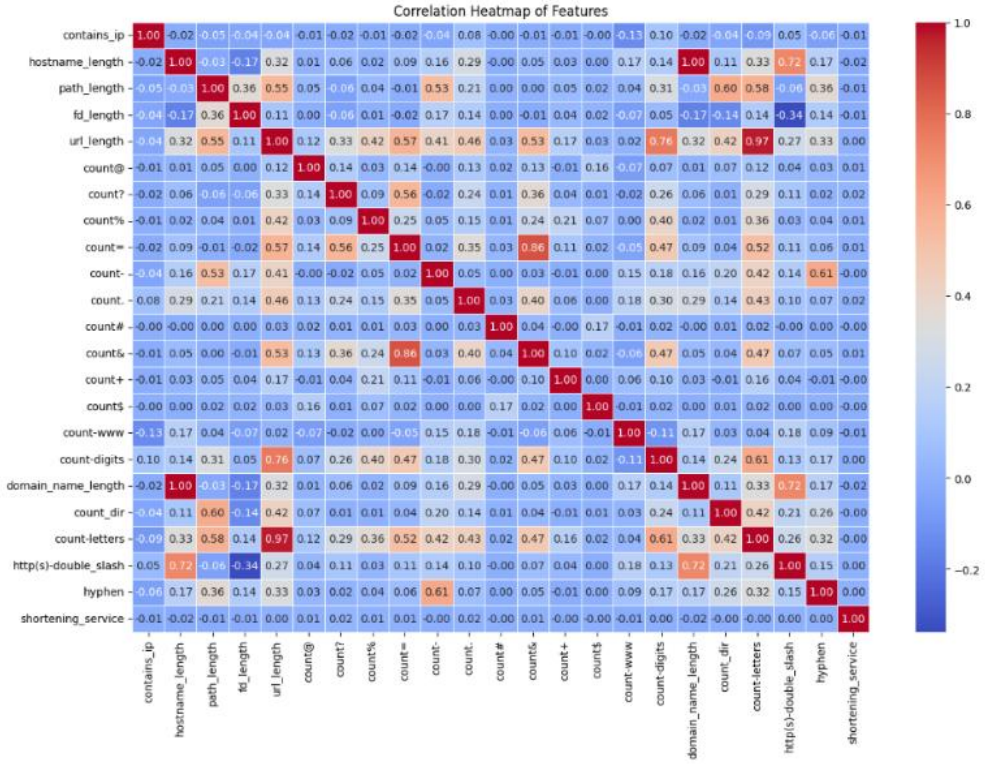

Figure 8. Correlation Heatmap

Figure 8 depicts the correlation between features. There is a strong positive correlation between path_length and url_length. Similarly, count= and count- exhibit a moderate correlation. Additionally, http(s)-double_slash and hostname_length show a notable correlation, implying that longer hostnames frequently appear in URLs with double slashes.

The correlation heatmap also highlights other notable relationships among features. For instance, count-digits and path_length exhibit a moderate correlation, suggesting that URLs with longer paths often contain more numeric characters. Additionally, fd_length and url_length show a significant correlation, which indicates that feature descriptor lengths tend to increase as the overall URL length increases. The relationship between http(s)-double_slash and hostname_length further suggests that URLs with multiple slashes are more likely to have longer hostnames. These correlations provide valuable insights into URL structures and their potential characteristics, aiding in feature selection and model optimization.

4.2. Performance Analysis

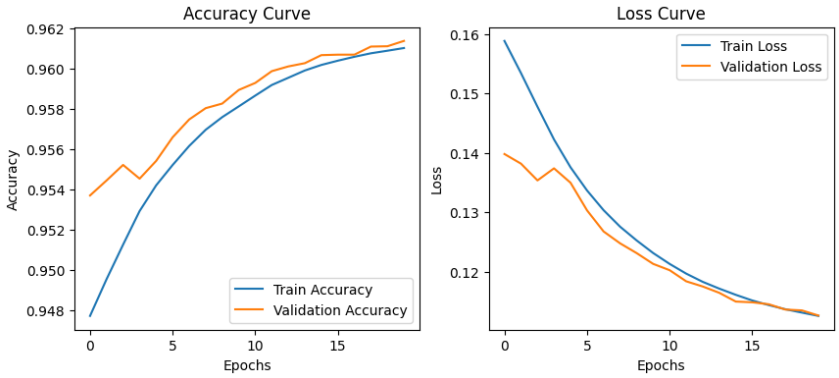

Figure 9. MLP Accuracy and Loss Curves

Figure 9 displays both the accuracy and loss curves for the customized Multi-Layer Perceptron (MLP) assisted with Genetic Algorithm (GA) over 20 epochs, illustrating the model's learning progress and

generalization ability. The training accuracy begins at 0.948 and gradually increases, reaching a final value of approximately 0.961. In contrast, the validation accuracy follows a similar pattern, starting at 0.954 and converging to a final accuracy of 0.961.

The consistent improvement in accuracy, with minimal divergence between training and validation performance, indicates that the model generalizes well and does not suffer from overfitting. This is further supported by the loss curve, where the training loss begins at around 0.16 and steadily decreases to a final value of 0.11. In contrast, the validation loss follows a similar trajectory, starting slightly below 0.14 and converging with the training loss by the final epoch.

The continuous decline in both training and validation loss, coupled with the steady rise in accuracy, suggests that the model has reached an optimal state, balancing bias and variance effectively. The convergence after around 15 epochs indicates that the model has achieved a stable learning state, making it well-suited for deployment in real-world applications.

Table 5. Classification Report

Class	Precision	Recall	F1-score	Support
Benign	0.94	0.99	0.96	39726
Malicious	0.98	0.94	0.96	39694

Table 5 displays a classification report for an MLP model with a Genetic Algorithm (GA). For the Benign class, the model achieved a Precision of 0.94, a Recall of 0.99, and an F1-score of 0.96, with a total of 39,726 samples. For the Malicious class, the model obtained a Precision of 0.98, a Recall of 0.94, and an F1-score of 0.96, with 39,694 samples.

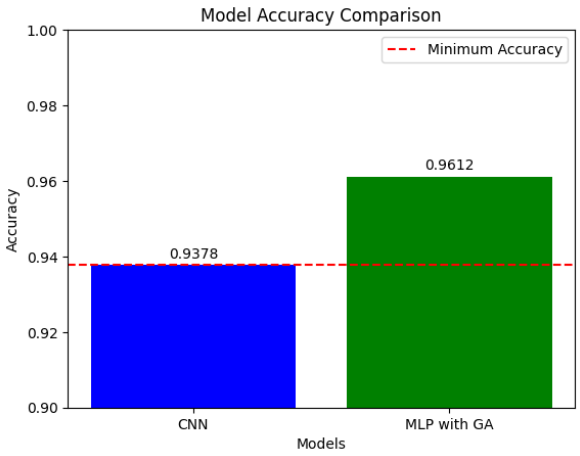

Figure 10. Comparison of Model Accuracies

Figure 10 illustrates a comparative performance analysis between a Convolutional Neural Network (CNN) and a Multi-Layer Perceptron (MLP) optimized with a Genetic Algorithm (GA), highlighting the substantial benefits of evolutionary optimization techniques in deep learning. The green bar, representing MLP with GA, achieves an accuracy of 0.9612, whereas the blue bar, corresponding to CNN, reaches 0.9378. This 2.34% improvement suggests that the integration of GA significantly enhances the learning process by optimizing hyperparameters more effectively. The red dashed horizontal line, labeled as "Minimum Accuracy" reinforcing the superiority of GA-enhanced optimization. This improvement is likely attributed to GA's ability to explore a broader search space, escape local optima, and dynamically adjust parameters for optimal weight initialization and training efficiency. Unlike CNNs, which rely heavily on feature extraction through convolutional operations, the MLP with GA leverages an intelligent and adaptive learning mechanism.

5. Conclusion and Future Enhancements

This study evaluates the effectiveness of Convolutional Neural Networks (CNN) and a Genetic Algorithm (GA)-optimized Multi-Layer Perceptron (MLP) for phishing URL detection. By leveraging GA for hyperparameter tuning, particularly optimizing layer sizes and learning rate, the MLP model achieved a superior accuracy of 96.12%, surpassing CNN's 93.78%. The GA-optimized MLP demonstrated a higher recall (94%) for malicious URLs compared to CNN (89%), ensuring a lower rate of undetected threats while maintaining a balanced tradeoff between precision (98%) and recall. The model's robustness is further reflected in its F1-score of 0.96, indicating strong generalization and stability across classes. Given its scalability and adaptability, the GA-enhanced MLP emerges as a more effective solution for phishing detection in security-critical applications.

Future research can explore integrating advanced metaheuristic optimization techniques such as Particle Swarm Optimization (PSO) and automated feature selection methods to enhance interpretability, reduce computational overhead, and further refine model efficiency. While Genetic Algorithm (GA) was employed for hyperparameter optimization in this study, alternative techniques such as Grid Search and Bayesian Optimization could be explored in future work. These methods provide systematic and probabilistic approaches to hyperparameter tuning, potentially improving efficiency and reducing computational overhead. A comparative analysis of these techniques with GA could further refine optimization strategies for phishing detection models.

Acknowledgement

We would like to express our sincere gratitude to our supervisor, Dr. Babu R. Dawadi, for his valuable guidance, encouragement, and continuous support throughout the course of this research. His insights and feedback have been instrumental in shaping the direction and quality of our work.

We also affirm that all authors have contributed equally to the research and preparation of this paper.

References

Arathi Krishna V, A. A. B. J. K. A. O. T. L., 2021. Phishing Detection using Machine Learning based URL Analysis: A Survey. *International Journal of Engineering Research & Technology (IJERT)*, 09(13).

Cagatay Catal, G. G. B. T. S. K. S. S., 2022. Applications of deep learning for phishing detection: a systematic literature review. *Knowledge and information systems*, 64(6), pp. 1457-1500.

Dutta, A. K., 2021. Detecting phishing websites using machine learning technique. *PLOS ONE*, 10, Volume 16, pp. 1-17.

Md. Nahiduzzaman, M. J. N. M. T. A. M. S. U. Z., 2019. *Prediction of Heart Disease Using Multi-Layer Perceptron Neural Network and Support Vector Machine.* s.l., s.n., pp. 1-6.

Rikiya Yamashita, M. N. R. K. G. D. K. T., 2018. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 08, 9(4), pp. 611-629.

Sanghyeop Lee, J. K. H. K.-Y. K. P., 2021. Genetic Algorithm Based Deep Learning Neural Network Structure and Hyperparameter Optimization. *Applied Sciences*, 11(2).

Serhii Lienkov, S. S. O. S. I. T. N. L. T. D., 2022. Deep Learning of Neural Networks Using Genetic Algorithms. *MoMLeT+ DS*, pp. 155-164.

Sk. Hasane Ahammad, S. D. K. G. D. U. S. P. E. V. B. A. V. D. D. K. J. B., 2022. Phishing URL detection using machine learning methods. *Advances in Engineering Software*, 01.Volume 173.

Youness Mourtaji, M. B. D. A. G. A. A. A., 2021. Hybrid Rule-Based Solution for Phishing URL Detection Using Convolutional Neural Network. *Wireless Communications and Mobile Computing*, pp. 824-1104.