Received: August 31, 2023. Revised: September 30, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

OptiPhishDetect: Optimized Phishing Detection through Learning based GCN with Scoring Model

Subashini K[1]* **Narmatha V[1]**

[1]*Department of Computer and Information Science, Annamalai University, Chidambaram, Tamil Nadu, India*
* Corresponding author's Email: subaphdscholar@gmail.com

Abstract: Phishing detection is a critical component of cyber security, aiming to safeguard users from malicious attempts to deceive and exploit. In this research, proposed an innovative approach that combines an adaptive threshold optimization technique with a learning-based graph convolutional network (GCN) and a scoring model to enhance phishing detection accuracy. The learning-based GCN is designed to analyze the intricate relationships within a graph structure, capturing nuanced dependencies between various entities such as email senders, recipients, and URLs. Leveraging this graph analysis, a scoring model assigns likelihood scores to instances being potential phishing attempts. Adaptive threshold optimization is a technique commonly used in feature selection to determine which features dynamically. In the context of phishing website detection, adaptive threshold optimization is used to select and prioritize the most informative features extracted from the websites. As a result of GCN, the model is able to differentiate between legitimate websites and phishing sites both locally and globally. In an effort to improve online security and safeguard users from phishing attacks, the proposed model is providing 99.3% accuracy in detecting phishing attacks. This enables a flexible and fine-tuned decision-making process, optimizing the trade-off between false positives and false negatives.

Keywords: Learning-based graph convolutional network, Cyber security, Phishing detection, Adaptive threshold optimization, Scoring model.

1. Introduction

In the rapidly evolving digital age, where technology enables seamless communication and transactions, the menace of cyber threats has escalated to unprecedented levels. Among these threats, phishing attacks have emerged as a persistent and pervasive danger, targeting individuals, businesses, and institutions across the globe [1]. Users are duped into divulging passwords, financial details, and personal information through phishing attacks by deceptive tactics. As the sophistication and frequency of these attacks continue to increase, the imperative for robust and adaptive phishing detection mechanisms becomes more pronounced [2].

Phishing attacks encompass a spectrum of techniques, from deceptive emails and counterfeit websites to social engineering ploys. The perpetrators exploit a combination of psychological manipulation and technological subterfuge, often impersonating legitimate entities to instil a false sense of trust and urgency [3]. The consequences of falling victim to a phishing attack can be devastating, resulting in financial losses, identity theft, data breaches, and compromised cyber security infrastructures [4]. This escalating threat landscape underscores the vital role of phishing detection in safeguarding individuals and organizations against malicious intent. Detecting phishing attempts involves scrutinizing many factors, ranging from the authenticity of sender addresses and content analysis to URL verification and user behaviour patterns [5]. Traditional rule-based methods have paved the way for more sophisticated, data-driven techniques, leveraging the power of deep learning, artificial intelligence, and graph analysis to discern subtle patterns indicative of phishing.

The rapid growth of data has fuelled the exploration of innovative techniques to process and extract valuable insights from complex datasets. Graph convolutional networks (GCNs) have emerged as a transformative approach to handling data structured in the form of graphs, paving the way for advancements across various fields, from social network analysis and biology to recommendation systems and cyber security [6]. Unlike traditional neural networks that operate on grid-like data such as images or sequences, GCNs are tailored to tackle data organized in graph structures. Graphs consist of nodes interconnected by edges, representing relationships or interactions between entities. GCNs leverage the inherent connectivity of graph data to capture intricate patterns and dependencies, making them particularly suited for applications where relationships between data points play a crucial role [7].

The architecture of a GCN draws inspiration from convolutional neural networks (CNNs), originally designed for image analysis. However, while CNNs exploit the spatial locality of pixels, GCNs focus on the relational locality of nodes within a graph. This distinction enables GCNs to perform localized information aggregation, where each node learns from its neighbours, effectively encoding local and global contexts [8]. The versatility of GCNs stems from their ability to learn expressive node representations through successive graph convolutional layers. These layers enable the model to capture complex structural information, making GCNs well-suited for node classification, link prediction, and graph classification tasks. The potential of GCNs has been harnessed to solve many problems, from identifying influential users in social networks to predicting molecular properties in chemical compounds.

The proposed OptiPhishDetect model is focused on identifying phishing attacks. In the context of phishing detection, this could mean that the model is trained to analyze and classify various attributes and relationships between elements, such as domain names, HTML contents and URLs, to determine whether they are associated with phishing. If the system uses graph convolutional networks, it may provide valuable insights into the relationships between various elements in the phishing landscape. This could lead to a better understanding of phishing campaigns, their origins, and their evolution.

Following is the rest of the paper: Section 2 covers an extensive review of existing works, section 3 provides the proposed model along with its architecture, section 4 discusses the findings, and finally section 5 concludes the research.

2. Related works

Detecting solutions for phishing websites significantly emphasise features engineering; yet, prior knowledge regarding characteristics is of essential importance to the solutions' overall accuracy [9]. First, the URL is utilized for rapid categorization by retrieving and analyzing attractiveness sequence characteristics. Phishing knowledge is not required for this phase, nor does it require assistance from a third party. Phishing is a deceptive practice that masquerades as legitimate websites to steal sensitive information from unsuspecting users. It is also a cyber-attack that is intended and carried out with the express intention of obtaining this information. This research [10] offered new phishing URL detection models utilising deep learning algorithms using just 10 characteristics of our prior work. These models used only our past work.

Phishing websites may often reproduce the look and feel of popular websites to trick readers into thinking they are viewing a genuine page. The individual who falls victim to fraud receives a hit to their finances, has their personal information taken, and suffers damage to their image as a result. When it came to the job of classifying phishing URLs, the authors of this research [11] evaluated several machine-learning approaches and found one that produced the best accuracy. This unique technique for modelling and detecting fraudulent URLs is based on deep reinforcement learning, and the ever-changing nature of phishing websites inspired it. The approach is introduced in this work [12]. With the help of the model, the characteristics associated with phishing website detection may be learned as phishing websites' behavior changes.

In today's world, phishing detection using a technique based on machine learning has proven fairly successful. The URL-based phishing detection method was used in this study (13), which demonstrates its usefulness. Using this plug-in, the user is notified if there is a risk of phishing when they view a web page, and if there is a risk, the system recognizes it in real time [14]. A variety of approaches are employed to attain higher levels of precision as part of the real-time prediction service. A novel rule-based hybrid solution was designed [15] in light of the fact that this problem is only getting worse. Six distinct algorithm models are combined to develop a novel hybrid approach to phishing detection and prevention. The investigation takes into account 37 characteristics gleaned from six different approaches.

This study [16] aimed to investigate the relative merits of machine learning and deep learning as approaches to developing a strategy for identifying potentially fraudulent websites by analysing their URLs. State-of-the-art phishing identification methods usually use homepages without login forms as the class for identifying phishing. This study [17] used ROC curve analysis and 10-fold cross-validation to demonstrate that the proposed technique is superior. A comparison with the most recent deep model revealed an increase in sensitivity of 3.98%. Researchers developed and implemented a DL-based website and resource locator-based phishing detection system. Phishing attacks were specifically identified using the approach. Deep learning approaches can also be applied to processing natural language and image classification [18].

This article's authors [19] developed a unique CNN that included self-attention CNN to identify phishing URLs. To be more specific, self-attention CNN initially uses GAN to create phishing URLs to ensure that the datasets have an equal number of authentic and phishing URLs. Linking to the linked website using hyperlinks available in the HTML source code, the author of this work [20] proposes an original method for identifying sites that are engaged in phishing. The suggested approach to identifying malicious websites uses a feature vector containing thirty characteristics.

There has been a great deal of recent research which has indicated that machine learning is becoming more critical in the current anti-phishing environment, and that methods such as deep learning are enabling anti-phishing systems to detect phishing attacks in a far more effective manner. This novel method of identifying phishing websites utilising the Dee learning model and GCN using URL and HTML characteristics is called PhishDet and is proposed in this research [21].

This research [22] has led to the development of a message-passing based graph convolution network which is considered to be an efficient method for detecting phishing nodes. This method was used to identify phishing nodes. In this research [23], the authors suggested a unique GNN-based phishing web page detection approach to successfully use the inherent structure. An equal and opposite expansion has followed the development of internet services in the number of cyberattacks. In this paper [24], the authors suggested a typical phishing e-mail classifier that uses DL techniques and a GCN to analyse the figure text to increase phishing detection accuracy.

This research study [25] presents a strategy for identifying phishing websites based on CNN and a machine learning model. This approach may determine whether or not URLs are legitimate without viewing the website's content or using any third-party services.

Table 1. Comparative analysis of related works

Related works	**Methods**	**Accuracy**	**Findings**
[9]	Multi dimensional feature phishing detection	98.99	Quick classification False Positive Rate 0.59%
[10]	Novel phishing URL detection models (DNN, LSTM, CNN)	99.2	An option to use third-party services, Robustness, Enhances speed
[12]	Deep reinforcement learning	98.2	Adapting To The Dynamic Model
[14]	Deep Learning-based framework, Real-Time prediction	99.18	RNN-GRU model obtained the highest accuracy
[15]	Comparative analysis between different ML and DL models	97.945	Highest accuracy level in deep learning
[17]	Deep convolutional autoencoder	98.7	Sensitivity improved by 3.98%
[18]	LSTM and CNN	93.28	Average detection time of 25s
[19]	CNN-LSTM and GAN	95.6	Improved by 1.4%
[21]	LRCN and GCN	96.42	0.036 False-negative rate

Potential phishing victims are contacted most often via email. Internet users can also be defrauded using phishing websites, as well as social media, ads, text messages, and telephone conversations. As a result, phishing emails cannot be the only type of email that can be detected. Therefore we need to concentrate on domain names, HTML contents and URL for online security.

3. Proposed model

In this research, the main objective is to improve the accuracy as well as efficiency of phishing detection by novel deep learning methods such as

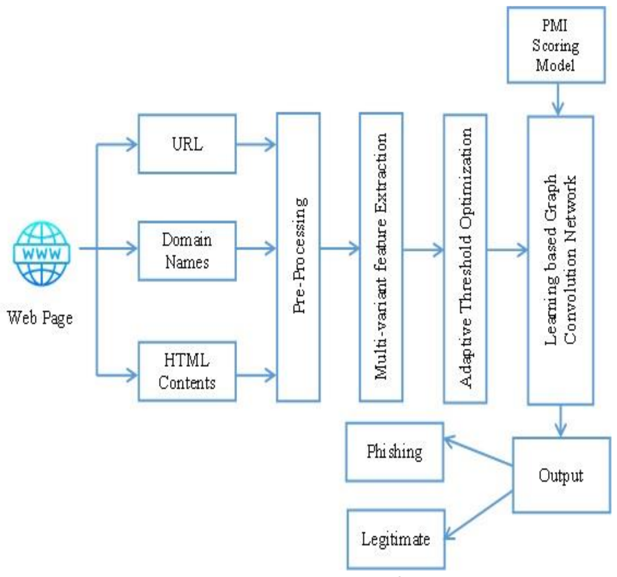
PMI Scoring Model
URL
Domain Names
HTML Contents
Web Page
Pre-Processing
Multi-variant feature Extraction
Adaptive Threshold Optimization
Learning based Graph Convolution Network
Phishing
Output
Legitimate

Figure. 1 An overall architecture of proposed model

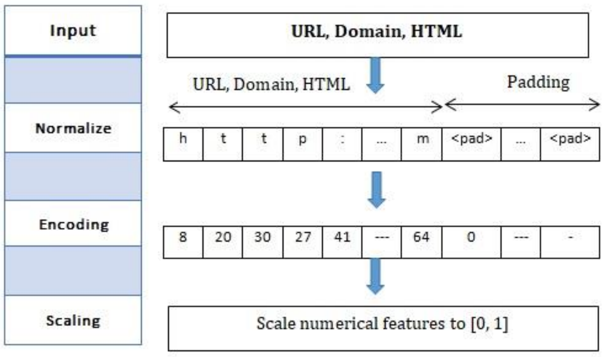

Figure. 2 Pre-processing stages

learning-based GCN and adaptive threshold optimization model. Designing a novel classification model for phishing website detection involves data pre-processing techniques, multi-variant feature extraction, adaptive threshold optimization and learning-based GCN as shown in Fig. 1.

3.1 Pre-processing

The data gathered from phishtank website and pre-process the data, which might include Normalization, encoding and scaling as shown in Fig. 2. These are common preprocessing steps for deep learning tasks in phishing website detection. As a result, machine learning algorithms are able to use the raw data more effectively.

Normalization: Normalization is the process of scaling numerical features to a common range, typically between 0 and 1. This ensures that different features contribute equally during training and prevents one feature from dominating others. In this proposed model, all the numerical features are normalized such as the number of links, the length of the URL, or the number of special characters.

Encoding: A model can understand categorical variables when they are encoded in a numerical format. For phishing website detection, you might have categorical features like the domain's top-level domain (TLD), the presence of certain keywords, or SSL certificate information. Convert categorical variables into binary vectors where each category is represented by a separate binary feature (0 or 1).

Scaling: Normalization involves transforming numerical features to have a mean and standard deviation of 0, but scaling transforms them to have a mean of 0 and standard deviation of 1. This is particularly useful when features have different scales, and you want to standardize them. It can improve the convergence of some algorithms.

Pseudocode for pre-processing

```
dataset = load_dataset('phishing_data.csv')
features = dataset[:, :-1]
labels = dataset[:, -1]
features = clean_and_extract_features(features)
encoded_features=
one_hot_encode_categorical(features)
train_features, val_features, test_features =
split_data(encoded_features, ratios=[0.6, 0.2, 0.2])
train_labels, val_labels, test_labels =
split_data(labels, ratios=[0.6, 0.2, 0.2])
scaled_train_features=
normalize_features(train_features)
scaled_val_features =
normalize_features(val_features)
scaled_test_features =
normalize_features(test_features)
```

In above pseudocode, split the dataset into features and labels, encoding categorical variables and scaling of numerical features using normalization features. The final datasets for training and evaluation have been pre-processed.

3.2 Multi-variant feature extraction

Multi-variant feature extraction is the process of extracting multiple types of features from website for phishing website detection. These features can help to capture the characteristics and behaviors of phishing websites, such as URL structure, domain registration, and HTML content. Multi-variant feature extraction for phishing website detection involves creating a set of informative features from various aspects of a website that may indicate whether it's a phishing site. These features capture different characteristics and behaviors of the website, making it easier for machine learning models to differentiate between legitimate and phishing websites.

URL-based features: URL-based features play a significant role in phishing website detection since URLs often contain valuable information about the nature of a website. Here are some URL-based features that can be extracted and used in the context of phishing website detection: This parameter counts how many characters are in the URL. It is possible to be impersonating a legitimate site or obfuscating the URLs with longer ones. Number of Subdomains is used to Count the number of subdomains in the URL. The Phishing sites might use subdomains to mimic legitimate domains. Tokenize the domain name to analyze its structure (e.g., www.abc.com -> ['www', 'abc', 'com']). Detect irregular or nonsensical token patterns.

These URL-based features provide valuable insights into the characteristics of a website's URL that can help in identifying potential phishing websites. By incorporating these features into the proposed phishing detection model, it is used enhance its ability to differentiate between legitimate and suspicious URLs.

Domain-based features: Domain-based features are providing insights into the legitimacy and trustworthiness of a domain. Malicious domains often have shorter domain names than legitimate domains. Legitimate websites typically use certain TLDs, while malicious sites use others. In some cases, newly registered domains may be more likely to be used for phishing. Some registrar and hosting providers are known to host a higher proportion of phishing sites. These domain-based features provide insights into the history, infrastructure, and security of a domain, helping in the identification of potential phishing websites. Incorporating these features into your phishing detection model can enhance its accuracy and ability to differentiate between legitimate and malicious domains.

HTML analysis: HTML analysis is a powerful technique for extracting features from web pages, including phishing websites. It involves parsing the HTML code of a webpage to extract various elements, attributes, and content that can provide insights into the nature of the site. This analysis starts to count the number of HTML forms present on the webpage and detect hidden fields or inputs that might capture sensitive data.

Pseudocode for multi-variant feature extraction

```
dataset = load_dataset('phishing_data.csv')
all_features = [ ]
for entry in dataset:
    url = entry['url']
    domain = extract_domain(url)
    html_content = fetch_html_content(url)
        # Extract URL-based features
    url_features = extract_url_features(url)
        # Extract domain-based features
    domain_features=extract_domain_features(domain)
        # Perform HTML analysis
    html_features =
perform_html_analysis(html_content)
        # Combine features
    combined_features = {url_features,
domain_features, html_features}
    all_features.append(combined_features)
```

This pseudocode demonstrates the process of extracting features from URLs, domains, and HTML content for phishing website detection. By initializing an empty feature matrix that will store the extracted features from multiple sources of data related to websites. An empty feature vector to store the extracted features specific to that website. After extracting features from all relevant sources for a given website, the feature vector with these features may extend.

3.3 Adaptive threshold optimization

Adaptive threshold optimization is a technique used for feature selection in deep learning model which is used to select optimal feature from extracted data and combined with classification model to enhance the prediction of phishing website. It typically refers to the process of dynamically adjusting a threshold value based on certain conditions. The mathematical equation for adaptive threshold optimization is given below:

The phishing and legitimate websites are represented by a set of features (attributes), and each website is characterized by certain characteristics denoted by $X = \{x1, x2, \dots, xn\}$, where n is the number of features and a threshold value, denoted as T

$$T_{new}(X) = T_{old}(X) + \Delta T\,(X) \tag{1}$$

Here, ΔT represents the change in the threshold value based on the feedback from the function $F(x)$ and the current threshold T_old. The actual equation for ΔT would depend on features that extracted from multi–variant algorithm of the proposed model.

For instance, an adaptive strategy that updates the threshold proportionally to the difference between the current feedback value and a desired target value:

$$\Delta T = k \times \left(target - F(x)\right) \tag{2}$$

Here, k is a scaling factor that determines how quickly the threshold should adapt based on the feedback difference. A pseudocode of adaptive threshold optimization that applied to phishing website detection is given below:

```
initial_threshold = 0.5 # Set an initial threshold
previous_T = 0
consecutive_decreases = 0
for epoch in range(num_epochs):
  predictions = predict_with_threshold(model,
dataset, current_threshold)
   T = calculate_T (predictions, true_labels)
   if T < previous_T:
      consecutive_decreases += 1
   else:
      consecutive_decreases = 0
   if consecutive_decreases >=
threshold_adjustment_threshold:
      current_threshold -= threshold_decrement_step
   previous_T = T
# Evaluate the final model with the adapted
threshold
final_predictions = predict_with_threshold(model,
test_data, current_threshold)
```

From the above psuedocode, sets an initial threshold for making binary predictions. In binary classification, this threshold determines whether a predicted probability or score corresponds to the positive class (e.g., phishing website detection) or the negative class (e.g., legitimate website). Here, T is adaptive threshold value. The model makes predictions on the training dataset using the current threshold. After the training loop completes, the final model is evaluated on a test dataset using the adapted threshold.

In this example, the threshold for detecting websites as phishing is adjusted based on the optimized threshold value. This approach helps the model adapt to changes in the data distribution and optimize its performance for detecting phishing websites. Based on evolving patterns and characteristics of data, adaptive threshold optimization can be applied to detect phishing websites and adjust thresholds for classifying websites as legitimate or phishing. The goal is to improve the accuracy of the detection model by dynamically setting the threshold according to the current distribution of data or the specific requirements of the task.

3.4 Learning-based GCN with scoring model

A learning-based graph convolutional network (GCN) is a type of neural network architecture designed to operate on graph-structured data. It performs tasks like node classification, link prediction, and graph classification based on both local and global information within the graph. Unlike traditional neural networks that process regular grid-like data, GCNs can handle irregular and interconnected data, making them well-suited for tasks where entities are intricately linked. The fundamental idea behind GCNs is to aggregate information from a node's neighbours to update its own representation. This process allows nodes to capture context from their immediate surroundings and more global graph-level patterns. The aggregation is achieved through convolutional layers that learn to balance local and global information and the structure of Learning-Base GCN is shown in Fig. 3.

Learning-based graph convolutional networks (GCNs) involve several mathematical equations to define information propagation through the graph, the aggregation of features from neighboring nodes, and the transformation of features within each convolutional layer. The core building block of a GCN is the graph convolution layer, which aggregates features from neighboring nodes. It computes weighted averages of neighboring node features, allowing nodes to incorporate information from their surroundings.

Let's consider a graph with nodes, indexed by i, and their features h_i^0at layer 0. The goal is to learn a representation h_i^L and the L-th layer, where L is the total number of layers.

Propagation rule:

The aggregation of neighboring node features and transformation of the aggregated features involve two main steps: aggregation and transformation. The propagation rules include simple mean aggregation, weighted aggregation based on edge weights, and more sophisticated attention mechanisms.

Aggregation step (weighted sum of neighbor features):

$$\alpha_i^l = \sum_{j \epsilon \mathrm{N}(i)} \frac{1}{\sqrt{|\mathrm{N}(i)| \cdot |\mathrm{N}(j)|}} h_j^{l-1} \tag{3}$$

Here, α_i^l represents the aggregated information for node i at layer l, $\mathrm{N}(i)$ is the set of neighboring nodes of i, and h_j^{l-1} is the feature of node j at layer l-1. The term $\frac{1}{\sqrt{|\mathrm{N}(i)| \cdot |\mathrm{N}(j)|}}$ is a normalization factor that scales the aggregated features.

Transformation step (applying a neural network operation):

$$h_i^l = \sigma\left(W^l a_i^l\right) \quad (4)$$

Here, W^l represents the learnable weight matrix at layer l, and σ is an activation function. (e.g., ReLU)

After aggregation, each node's feature is transformed using a shared neural network operation, often involving learnable weights and activation functions. GCNs can consist of multiple stacked graph convolution layers, allowing the network to capture increasingly abstract features from the graph data. To stack multiple layers of GCNs, we iterate the aggregation and transformation steps:

$$h_i^{l+1} = \sigma\left(W^l\left(\sum_{j\epsilon \mathrm{N}(i)} \frac{1}{\sqrt{|\mathrm{N}(i)| \cdot |\mathrm{N}(j)|}} h_j^l\right)\right) \quad (5)$$

Here, l is the current layer index and h_i^{l+1} represents the updated feature for node *I* at layer l+1.

Depending on the specific task, an objective function that measures the model's performance. In a node classification task with cross-entropy loss, the objective function might be:

$$L = -\sum_{i\in Nodes}\sum_{c\in Classes} yi,c \log(\hat{y}i,c) \quad (6)$$

Here, yi,c is the ground truth label for node *i* and class c, and $\hat{y}i,c$ is the predicted probability from the GCN.

Above equations represents the aggregation and transformation in GCNs, allowing nodes to learn informative features from their neighbors and update their representations through multiple layers. The proposed framework of learning from graph-structured data, enabling tasks ranging from node classification to graph-level analysis. Their ability to extract meaningful features while considering local and global context makes them a valuable tool in understanding complex interconnected systems.

Using pointwise mutual information (PMI) as a scoring model to calculate the phishing level of websites involves quantifying the association between certain features and the likelihood of a website being phishing. This approach can help you identify features strongly indicating phishing characteristics and assign a score reflecting the phishing likelihood. Calculate the PMI scores for each feature with respect to the "phishing" class. PMI can be calculated using the formula:

$$PMI\ (x, phishing) = log_2 \frac{P(x.phishing)}{P(x).P(phishing)} \quad (7)$$

Here, x is a feature, and phishing is the phishing class. Phishing class and feature x add up to a combined probability of $P(x.phishing)$. $P(x)$ is he probability of feature x occurring, and $P(phishing)$ is the probability of the phishing class.

For each website, calculate a phishing level score based on the PMI scores of its features. Sum the PMI scores of the features present in the website's content to compute the phishing level score. Higher phishing level scores indicate a higher likelihood of the website being phishing. Determine a threshold value that separates websites into different levels of phishing likelihood (e.g., low, medium, high). Classify websites based on their phishing level scores and the chosen threshold.

Pseudocode for learning-based GCN

A pseudocode outline for implementing a learning-based GCN for the classification of phishing websites is given below. It focuses on node-level classification, treating each webpage as a node in the graph and classifying it as either a phishing or legitimate website.

Input: URL, HTML_contents, Domain_names
Output: Type of website with score

```
Procedure LGCN
N → Number of nodes (Input)
G → Graph
F → Feature of each node (web page)
for N ϵ G do
Ã = A + I
A → Adjacency matrix of G
Ã → Transformation to the adjacency matrix
Di=∑j Ã ij
for F ϵ N do
$W_{out}$ = $i_w$ (hidden unit, Output unit [0,1])
$B_{out}$ = $i_b$ (hidden unit, Output unit [0,1])
end for
end for
for webpage (URL, HTML_contents, Domain_names) do
normalization factors = $\frac{1}{\sqrt{|\mathrm{N}(i)| \cdot |\mathrm{N}(j)|}}$
aggregated_features = $\sum_{j\epsilon \mathrm{N}(i)} \frac{1}{\sqrt{|\mathrm{N}(i)| \cdot |\mathrm{N}(j)|}} h_j^l$
Propogation = σ(Di Ã$h_j^l$ w(l))
$h_j^l$ is the feature matrix at layer 'l'
w(l) is the learnable weight matrix of layer 'l'
σ is activation function
end for
for webpage ϵ range do
$PMI\ (N, phishing) = log_2 \frac{P(x.phishing)}{P(x).P(phishing)}$
Scores= X(l) × PMI
X → Feature matrix
X(l) → Feature matrix of layer l
```

$$l = -\sum \in N \sum_{N \in G} y_i, N log\ (y_i, N)$$
$$O = \text{softmax}\left(D_i \text{ x } \overline{A} \text{ x } h_j^{l+1} \text{x } w^{(l+1)}\right)$$

Here, $w^{(l+1)}$ is learnable graph of output layer, h_j^{l+1} is final layer

In this pseudocode ,$load\_webpage\_graph$ loads the webpage graph data containing connections between webpages. $load\_webpage\_features$ loads features for each webpage (e.g., URL-based, domain-based, HTML-based features). $initialize\_features$ initializes the feature matrix for webpages. $create\_adjacency\_matrix$ creates an adjacency matrix to represent connections between webpages. For each webpage, information is propagated through its neighbors and transformed using the weight matrix and activation function. The functions for calculating PMI scores, calculating phishing level scores, and classifying websites based on their scores. The softmax function is used for final classification scores to convert them into probabilities. PMI scores indicate the strength of association between those features and phishing behavior. The threshold and classification step allow you to categorize websites into different levels of phishing likelihood based on their scores.

4. Results and discussions

4.1 Dataset

The dataset was collected from "https://www.kaggle.com/code/buneshathankar25/phishing-url-detection". This dataset contains the URLs of more than 11000 websites. Then created 30 samples of websites and assigned a class label indicating whether or not they are phishing websites (1 or -1) based on 30 website parameters. Also, the data set was used to determine the function and non-function requirements for the research as well as to assist in scoping. In total, 4721 high quality URLs were found in the original dataset after the filtering steps had been completed: 2420 of them benign, and 2301 of them phishing. Datasets are crawled and URLs are mapped into graph matrices based on the URLs captured in the crawl. Performance evaluation for a Learning-Based GCN in the classification of phishing website detection involves assessing the model's accuracy, precision, recall, F1-score, and possibly other metrics.

Accuracy: The accuracy of a prediction is determined by the ratio of correct predictions to the total number of predictions.

$$Accuracy = \frac{TP+FN}{TP+TN+FP+FN} \quad (8)$$

Precision (positive predictive value): The precision of a prediction is measured by how many true positive predictions there are compared with how many positive predictions there are in total.

$$Precision = \frac{TP}{TP+FP} \quad (9)$$

Recall (sensitivity, true positive rate): Basically, recall measures how many positive predictions are actually true over the total number of predictions that are actually true.

$$Recall = \frac{TP}{TP+FN} \quad (10)$$

F1-Score: An F1-score provides a balanced measure of model performance by combining the harmonics of precision and recall to give a F1-score.

$$F1 - Score = \frac{2.PRecison.Recall}{Precision+Recall} \quad (11)$$

Specificity (true negative rate): True negative predictions are measured by the ratio of the number of actual negatives to the total number of true negative predictions.

$$Specificity = \frac{TN}{TN+FP} \quad (12)$$

False positive rate: In other words, False Positive Rate is a measure of how many positives are made against the number of actual negatives.

$$False\ Positive\ Rate = \frac{FP}{FP+TN} \quad (13)$$

As shown in Table 2, the proposed model has a number of performance metrics. A classification model's performance can be assessed based on these metrics. Here true negative is stated as TN, true positive as TP, false negative as FN and false positive as FP.

Confusion matrix: According to Table 3, a confusion matrix shows the number of true positives, true negatives, false positives, and false negatives predicted for 4721 samples.

An accuracy curves to monitor how the model's accuracy changes during training and testing phases. As can be seen in Fig. 4 the number of training iterations (epochs) on the x-axis as well as the training accuracy on the y-axis, respectively. The training and testing accuracy might be low at the start, both during the training and during the testing

Table 2. Metrics of proposed model

Metrics	Our Proposed Model
Precision	0.9923
Specificity	0.9938
F-measure	0.9932
Accuracy	0.9931
Recall	0.9924
False Positive Rate	0.0076

Table 3. Confusion matrix for a test set of 4721 samples

	Predicted Positive	Predicted Negative
Actual Positive	2331	36
Actual Negative	29	2325

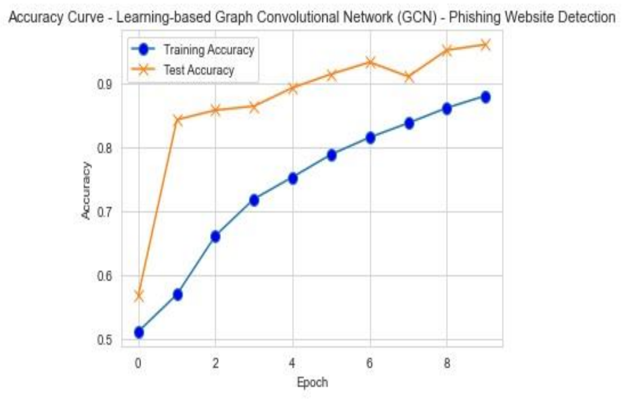

Figure. 4 Accuracy curve during training and testing

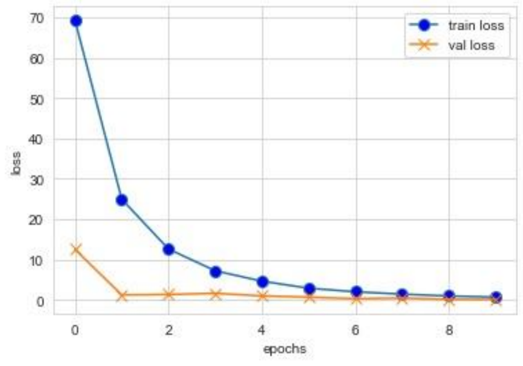

Figure. 5 Loss curve during training and validation

phase. The training accuracy of the model generally increases as the model learns.

The training and validation loss curve is another crucial aspect of monitoring the performance of proposed learning models. As shown in figure 5, the y-axis represents the training loss, while the x-axis represents the number of training iterations (epochs). As the model begins with random weights, there is likely to be a high training loss. In order for the model to improve its predictions on the training data, it should decrease its training loss as it learns.

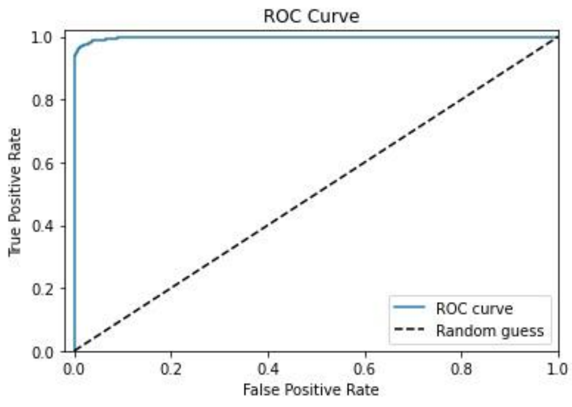

Figure. 6 RoC curve of proposed model

Table 4. A comparison between the proposed model and the existing works

Metrics (%)	[10]	[14]	[9]	Proposed Model
Precision	96.2	97.13	96.25	**99.23**
Recall	97.54	99.12	96.54	**99.24**
F-measure	99.06	97.26	98.24	**99.32**
Accuracy	99.2	99.18	98.99	**99.31**

Fig. 6 illustrates the ROC curve used to evaluate classification models for different threshold settings. The model assigns a probability score to determine whether an instance belongs to a positive category. Adjusting the threshold for classifying instances as positive or negative controls the balance between sensitivity and specificity.

When compared with existing models, Somesha et al., algorithm achieved 99.2% accuracy, 99.06% F-measure, 97.54% recall, and 96.2% precision. Based on Tang et al.'s algorithm, the accuracy was 99.18%, F-measure was 97.26%, recall was 99.12%, and precision was 97.13%. As a result of Yang et al., algorithm applied in dataset, 98.99% accuracy was achieved with 98.24% F-measure, 96.54% recall, and 96.25% precision. Our proposed phishing website detection method outperformed existing methods, obtaining an accuracy of 99.31%, precision of 99.23%, F-measure of 99.32%, and recall of 99.24%. Compared to the baseline model, the results of our proposed model are shown in Table 4.

5. Conclusion

This research presented a novel approach for enhancing phishing detection through the integration of learning-based CGN with a PMI scoring model. By leveraging the power of deep learning and learning-based graph representations, the proposed

Received: August 31, 2023. Revised: September 30, 2023.

method demonstrated significant improvements in accurately identifying phishing attempts with level of severity. The PMI scores indicate the strength of association between those features and phishing behavior. By extracting features from various sources like URLs, domains, and HTML content, the model empowers the classifier with detailed insights to discriminate between phishing and legitimate attributes effectively. Adaptive threshold optimization technique optimizes the trade-off between precision and recall, ultimately leading to a more balanced and effective detection system. GCN can capture local and global patterns within the graph, thereby enhancing the model's ability to differentiate between phishing and legitimate websites. The proposed model contributing significantly to enhancing online security and safeguarding users from the perils of phishing attacks with 99.3% accuracy. In future, developing a real-time phishing detection model that can quickly classify websites as phishing or legitimate in a timely manner. By considering and integrating streaming data analysis and continuous monitoring to identify new phishing campaigns promptly.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, Subashini and Narmatha; methodology, Subashini and Narmatha; software, Subashini; validation, Subashini; formal analysis, Subashini and Narmatha; investigation, Subashini and Narmatha; resources, Subashini; data curation, Subashini; writing-original draft preparation, Subashini; writing-review and editing, Subashini and Narmatha; visualization, Subashini; supervision, Subashini; project administration, Subashini; funding acquisition, Subashini. All authors have read and approved the final manuscript.

References

[1] A. Safi and S. Singh, "A systematic literature review on phishing website detection techniques", *Journal of King Saud University-Computer and Information Sciences*, Vol. 35, No. 15, pp. 590-611, 2023.

[2] R. Zieni, L. Massari, and M. C. Calzarossa, "Phishing or not phishing? A survey on the detection of phishing websites", *IEEE Access*, Vol. 11, pp. 18499-18519, 2023.

[3] A. S. Bozkir, F. C. Dalgic, and M. Aydos, "GramBeddings: a new neural network for URL based identification of phishing web pages through n-gram embeddings", *Computers & Security*, Vol. 124, pp. 102964, 2023.

[4] U. A. Butt, R. Amin, H. Aldabbas, S. Mohan, B. Alouffi, and A. Ahmadian, "Cloud-based email phishing attack using machine and deep learning algorithm", *Complex & Intelligent Systems*, Vol. 9, No. 3, pp. 3043-3070, 2023.

[5] Y. Wang, W. Zhu, H. Xu, Z. Qin, K. Ren, and W. Ma, "A Large-Scale Pretrained Deep Model for Phishing URL Detection", In: *Proc. of* 2023 *IEEE International Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1-5, 2023.

[6] T. Wen, Y. Xiao, A. Wang, and H. Wang, "A novel hybrid feature fusion model for detecting phishing scam on Ethereum using deep neural network", *Expert Systems with Applications*, Vol. 211, p. 118463, 2023.

[7] Z. Zhang, T. He, K. Chen, B. Zhang, Q. Wang, and L. Yuan, "Phishing Node Detection in Ethereum Transaction Network Using Graph Convolutional Networks", *Applied Sciences*, Vol. 13, No. 11, pp. 6430, 2023.

[8] A. K. Dutta, T. Meyyappan, B. Qureshi, M. Alsanea, A. W. Abulfaraj, M. A. Faraj, and A. R. W. Sait, "Optimal Deep Belief Network Enabled Cybersecurity Phishing Email Classification", *Comput. Syst. Sci. Eng.*, Vol. 44, No. 3, pp. 2701-2713, 2023.

[9] P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning", *IEEE Access*, Vol. *7*, pp. 15196-15209, 2019.

[10] M. Somesha, A. R. Pais, R. S. Rao, and V. S. Rathour, "Efficient deep learning techniques for the detection of phishing websites", *Sādhanā*, Vol. 45, pp. 1-18, 2020.

[11] J. Kumar, A. Santhanavijayan, B. Janet, B. Rajendran, and B. S. Bindhumadhava, "Phishing website classification and detection using machine learning", In: *Proc. of International Conf. on Computer Communication and Informatics (ICCCI)*, pp. 1-6, IEEE, 2020.

[12] M. Chatterjee and A. S. Namin, "Detecting phishing websites through deep reinforcement learning", In: *Proc. of 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 2, pp. 227-232, IEEE, 2019.

[13] M. T. Suleman and S. M. Awan, "Optimization of URL-based phishing websites detection through genetic algorithms", *Automatic Control and Computer Sciences*, Vol. 53, pp. 333-341, 2019.

[14] L. Tang and Q. H. Mahmoud, “A deep learning-based framework for phishing website detection”, *IEEE Access*, Vol. 10, pp. 1509-1521, 2021.

[15] Y. Mourtaji, M. Bouhorma, D. Alghazzawi, G. Aldabbagh, and A. Alghamdi, “Hybrid rule-based solution for phishing URL detection using convolutional neural network”, *Wireless Communications and Mobile Computing*, Vol. 2021, pp. 1-24, 2021.

[16] M. S. Paniagua, E. F. Fernández, E. Alegre, W. A. Nabki, and V. G. Castro, “Phishing URL detection: A real-case scenario through login URLs”, *IEEE Access*, Vol. 10, pp. 42949-42960, 2022.

[17] S. J. Bu and S. B. Cho, “Deep character-level anomaly detection based on a convolutional autoencoder for zero-day phishing URL detection”, *Electronics*, Vol. 10, No. 12, p. 1492, 2021.

[18] M. A. Adebowale, K. T. Lwin, and M. A. Hossain, “Intelligent phishing detection scheme using deep learning algorithms”, *Journal of Enterprise Information Management*, Vol. 36, No. 3, pp. 747-766, 2023.

[19] X. Xiao, W. Xiao, D. Zhang, B. Zhang, G. Hu, Q. Li, and S. Xia, “Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets”, *Computers & Security*, Vol. 108, p. 102372, 2021.

[20] L. Lakshmi, M. P. Reddy, C. Santhaiah, and U. J. Reddy, “Smart phishing detection in web pages using supervised deep learning classification and optimization technique adam”, *Wireless Personal Communications*, Vol. 118, No. 4, pp. 3549-3564, 2021.

[21] S. Ariyadasa, S. Fernando, and S. Fernando, “Combining long-term recurrent convolutional and graph convolutional networks to detect phishing sites using URL and HTML”, *IEEE Access*, Vol. 10, pp. 82355-82375, 2022.

[22] T. Yu, X. Chen, Z. Xu, and J. Xu, “MP-GCN: A Phishing Nodes Detection Approach via Graph Convolution Network for Ethereum”, *Applied Sciences*, Vol. 12, No. 14, pp. 7294, 2022.

[23] L. Ouyang and Y. Zhang, “Phishing web page detection with html-level graph neural network”, In: *Proc. of* 2021 *IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 952-958, IEEE, 2021.

[24] A. Alhogail and A. Alsabih, “Applying machine learning and natural language processing to detect phishing email”, *Computers & Security*, Vol. 110, p. 102414, 2021.

[25] R. Yang, K. Zheng, B. Wu, C. Wu, and X. Wang, “Phishing website detection based on deep convolutional neural network and random forest ensemble learning”, *Sensors*, Vol. 21, No. 24, p. 8281, 2021.