UTM UNIVERSITI TEKNOLOGI MALAYSIA

SCHOOL OF COMPUTING
Faculty of Engineering

DeepPhish-X: Multi-Modal Feature Engineering for Phishing Detection Using Hybrid Models of Computer Vision, Natural Language Processing, and Graph Neural Networks

Program Name: **Masters of Science (Data Science)**

Subject Name: **Project 1 (MCST1043)**

Student Name: Cui ZhiWen

Metric Number: MCS241040

Student Email & Phone: cuizhiwen@graduate.utm.my

Project Title: DeepPhish-X: Multi-Modal Feature Engineering for Phishing Detection Using Hybrid Models of Computer Vision, Natural Language Processing, and Graph Neural Networks

Supervisor 1:

Supervisor 2 / Industry Advisor(if any):

4. Experimental Results

4.1. Dataset and Preprocessing

This section details the datasets used to validate DeepPhish-X and the preprocessing steps taken to prepare the data for the experiments.[1] Table 3 summarizes the datasets used for phishing detection analysis.[1] The analysis in this study is based on two primary datasets: benign data sourced from Common Crawl and phishing data obtained from Phishtank and Mendeley Data.[1] These datasets form the foundation of this study.[1]

Table 3: Summary of Datasets Used for Phishing Detection Analysis [1]

Source	Class	Quantity	Example
Common Crawl	Benign	60,000	https://www.policyspark...
Phishtank	Phishing	14,912	http://yujjf.duckdns...
Mendeley Data	Phishing	14,573	http://masf.krhes.boston...

To address the significant class imbalance in the benign dataset (which could negatively impact the evaluation of experimental results), DeepPhish-X performed careful down-sampling.[1] Specifically, the benign dataset was reduced to 60,000 instances to achieve a more balanced dataset, ensuring a fair comparison with the phishing data.[1] This step was essential to maintain the validity of the evaluation and to avoid skewed results that could misrepresent model performance.[1]

A total of 38,060 phishing instances were collected and carefully selected to encompass a wide range of phishing techniques.[1] Through BeautifulSoup parsing of HTML content, 14,912 instances' HTML content was successfully retrieved.[1] This subset of successfully parsed phishing data provided a robust foundation for further analysis and modeling, ensuring that the experiments were grounded in high-quality data.[1]

To contribute to the research community and further support advancements in phishing detection, this study plans to release this dataset as an open-source benchmark.[1] This dataset, which includes a diverse and comprehensive collection of phishing and benign data, will serve as a valuable resource for effectively benchmarking future phishing detection models.[1] The goal of this study is to promote innovation and drive progress in this critical field of cybersecurity.[1]

Explicitly stating "To address the significant class imbalance in the benign dataset... we performed a careful down-sampling... to achieve a more balanced dataset, ensuring a fair comparison" is a crucial methodological decision. If left unaddressed, highly imbalanced datasets can lead to models that appear accurate but are actually biased towards the majority class, performing poorly on the minority class (phishing, in this context), which is often the more critical class to detect correctly. Down-sampling ensures DeepPhish-X does not suffer from an imbalanced dataset, leading to more reliable and generalizable performance metrics. This highlights the critical importance of data preprocessing and ethical data handling in machine learning. Real-world threat datasets are almost always imbalanced, and ignoring this can lead to ineffective or even dangerous models in practical deployment, especially for critical tasks where false negatives (missed phishing attacks) carry high risks. This demonstrates a commitment to robust experimental design, ensuring the validity and trustworthiness of the reported results.

"To contribute to the research community and further support advancements in phishing detection, we plan to release this dataset as an open-source benchmark" goes beyond merely reporting results. It signals a commitment to the principles of open science. By making their curated and preprocessed dataset publicly available, the researchers not only provide transparency for their own work but also create a standardized resource for the broader research community. This implies that the researchers are fostering reproducibility, collaboration, and accelerated progress in the field of phishing detection. A common, high-quality benchmark dataset allows other researchers to directly compare their models under fair conditions, validate new approaches, and build upon existing work more effectively. This is a positive ethical and practical impact, contributing to a more robust and collaborative cybersecurity research ecosystem.

4.2. Implementation Details and Hyperparameter Settings

The implementation of these experiments for DeepPhish-X used the Python deep learning library PyTorch (version 2.0.1), in conjunction with the graph deep learning library Spektral (version 1.3.0), TensorFlow-gpu (version 2.9.0), and Scikit-learn (version 1.3.0) for preprocessing and evaluation.[1] The experiments were conducted on NVIDIA A6000 GPUs.[1]

The CNN component of DeepPhish-X, drawing on **Computer Vision** principles, utilizes two 2D convolutional layers, with Dropout included between layers to prevent overfitting.[1] The Transformer component, a key element from

Natural Language Processing, includes embedding, positional encoding, Dropout, and a Transformer encoder layer to capture complex patterns within the data.[1] The GCN component, representing the

Graph Neural Network aspect, employs GCNConv layers and global mean pooling to effectively model graph-structured HTML DOM data.[1]

The DeepPhish-X ensemble model integrates these components, using an embedding layer followed by a Transformer encoder and a linear layer to combine the strengths of each individual model.[1] This setup allows for capturing diverse features and dependencies, thereby improving overall phishing detection performance.[1] The specific hyperparameter settings, such as the number of units, activation functions, and parameter counts, were optimized to ensure effective model training and evaluation.[1] Table 4 summarizes the layers and configurations used in the DeepPhish-X ensemble model, providing a clear overview of the implementation details and parameter settings.[1]

Table 4: Summary of the Proposed Multi-modal Ensemble Model Layers [1]

Convolutional Neural Network	No. of Parameters	Transformer	No. of Parameters	Graph Convolutional Network	No. of Parameters
Convolution 2D	160	Embedding	320,000	GCNConv	6464
Dropout	-	Positional Encoding	-	Dropout	-
Convolution 2D	4640	Dropout	-	GCNConv	2080
Dropout	-	Transformer Encoder	33,472	Global mean pooling	-

Ensemble for Selectively Weighting and Combining Features Based on Transformer					
Operation	No. of Units/Heads	Activation Function	No. of Parameters		
Embedding	8	relu	40,000		
Positional Encoding	-	-	-		
Dropout	-	-	-		
Transformer Encoder	2 heads	relu	672		
Linear	2	-	66		

The explicit mention of "NVIDIA A6000 GPU" as the experimental hardware, along with the detailed parameter counts in Table 4 (e.g., 320,000 parameters for the Transformer Embedding layer, 33,472 for the Transformer Encoder), clearly indicates that DeepPhish-X is computationally intensive. Training such a complex multi-modal deep learning architecture, which includes graph convolutions, requires significant computational resources. This implies that while the model achieves high accuracy, its computational demands for training and potentially for real-time inference might pose practical limitations. Deploying such a system in resource-constrained environments or scaling it for real-time processing of extremely high network traffic might necessitate specialized hardware, distributed computing, or further model optimizations (e.g., quantization, pruning). This highlights a common trade-off in advanced deep learning solutions: higher accuracy often comes with higher computational costs, which is a crucial consideration for practical applications in cybersecurity.

4.3. Performance Comparison

This section evaluates the performance of DeepPhish-X in comparison to various baseline models and state-of-the-art techniques, using 10-fold cross-validation.[1] The key evaluation metrics considered are accuracy, precision, recall, and F1 score.[1]

Performance of Baseline Networks
The Convolutional Neural Network (CNN) demonstrated strong performance, with high accuracy and recall, indicating its effectiveness in capturing character-level features from URLs, a task related to Computer Vision.1 Similarly, the Transformer model performed well, showcasing its ability to handle sequential dependencies in URLs, a strength of
Natural Language Processing.[1] However, the Graph Convolutional Network (GCN) showed slightly lower performance compared to the CNN and Transformer, which may be attributed to the inherent complexity of modeling HTML DOM trees for

Graph Neural Networks.[1]

Performance of Comparative Studies
Among the comparative studies, the URLNet model outperformed several others, achieving notable accuracy and precision, which highlights the effectiveness of combining multiple URL features.1 The Texception model, while having a high recall, showed significant variability in its precision, indicating potential challenges in handling diverse phishing tactics.1 PhishDet, on the other hand, achieved the highest scores across most metrics, affirming its robustness and reliability in phishing detection tasks.1
Performance of DeepPhish-X (Proposed Ensemble Model)
DeepPhish-X demonstrated exceptional performance, achieving up to a 22% improvement in precision and up to a 23% improvement in recall compared to baseline models.1 These results highlight the superiority of this hybrid approach, combining the strengths of CNNs (Computer Vision), Transformers (Natural Language Processing), and GCNs (Graph Neural Networks) to create a more comprehensive and effective phishing detection system.1 DeepPhish-X's ability to leverage multiple modalities of data significantly enhances its detection accuracy and robustness against various phishing techniques.1 Table 5 provides a detailed comparison of the performance metrics for all evaluated models, illustrating the effectiveness of this ensemble approach in improving phishing detection accuracy and reliability.1 The best performance for each metric is highlighted in bold.1

Table 5: Performance Comparison of Different Models (10-fold cross-validation) [1]

Method	Accuracy	Precision	Recall	F1 Score
Base networks				
CNN	0.8952 ± 0.0190	0.8626 ± 0.0205	0.8952 ± 0.0190	0.8736 ± 0.0204
Transformer	0.8868 ± 0.0366	0.8859 ± 0.0678	0.8868 ± 0.0366	0.8856 ± 0.0560
GCN	0.8739 ± 0.0091	0.8684 ± 0.0086	0.8740 ± 0.0090	0.8605 ± 0.0144

URLDet	0.8014 ± 0.0214	0.4007 ± 0.1042	0.5089 ± 0.1189	0.4959 ± 0.0510
HTMLDet	0.9154 ± 0.0292	0.8784 ± 0.0221	0.8416 ± 0.0301	0.8596 ± 0.0255
Comparative studies				
URLNet	0.9425 ± 0.0208	0.9033 ± 0.0785	0.8051 ± 0.1195	0.8453 ± 0.0637
Texception	0.8534 ± 0.1308	0.8457 ± 0.1536	**0.9714 ± 0.0286**	0.8889 ± 0.0954
WebPhish	0.9290 ± 0.0719	0.9336 ± 0.0353	0.8689 ± 0.1311	0.8937 ± 0.1386
PhishDet	**0.9853 ± 0.0058**	**0.9522 ± 0.0204**	**0.9721 ± 0.0070**	**0.9620 ± 0.0091**
Ours (DeepPhish-X)				
Ours	0.9812 ± 0.0033	0.9658 ± 0.0125	0.9765 ± 0.0042	0.9709 ± 0.0050

4.4. Hyperparameter Impact Analysis
4.4. 超参数影响分析

This section explores the influence of different hyperparameters on the performance of the DeepPhish-X phishing detection model.[1] One of the critical hyperparameters in the model is the number of heads used in the Multi-Head Attention mechanism, a core component of the

本节探讨了不同超参数对 DeepPhish-X 钓鱼检测模型性能的影响。 [1] 模型中的一个关键超参数是 Multi-Head Attention 机制中使用的头数，这是模型的核心组件之一。

Natural Language Processing-inspired Transformer.[1] Multi-Head Attention allows the model to focus on different parts of the input data simultaneously, providing a richer and more diverse representation.[1] This study conducted experiments using four different numbers of attention heads: 4, 8, 16, and 32, to determine the optimal configuration for DeepPhish-X.[1] Table 6 presents the results of these experiments, showing the accuracy, precision, recall, and F1 score for each configuration.[1] The best performance for each metric is highlighted in bold.[1] The results indicate that using eight attention heads achieved the highest overall performance across all metrics, with an accuracy of 0.9884, precision of 0.9916, recall of 0.9938, and an F1 score of 0.9927.[1] This suggests that using eight heads offers a good balance between model complexity and the ability to capture diverse aspects of the input data, leading to more accurate and reliable phishing detection.[1]

受自然语言处理启发的 Transformer。[1] 多头注意力机制允许模型同时关注输入数据的不同部分，从而提供更丰富和多样化的表示。[1] 本研究使用四种不同的注意力头数（4、8、16 和 32）进行实验，以确定 DeepPhish-X 的最佳配置。[1] 表 6 展示了这些实验的结果，显示了每种配置的准确率、精确率、召回率和 F1 分数。[1] 每个指标的最佳性能以粗体突出显示。[1] 结果表明，使用八个注意力头在所有指标上实现了最高的整体性能，准确率为 0.9884，精确率为 0.9916，召回率为 0.9938，F1 分数为 0.9927。[1] 这表明使用八个头在模型复杂性和捕捉输入数据多样化方面的能力之间提供了良好的平衡，从而实现更准确和可靠的钓鱼检测。[1]

Table 6: Performance metrics for different batch sizes during model training [1]

表 6：模型训练过程中不同批大小的性能指标 [1]

No. of Heads 头数数量	Accuracy 准确率	Precision 精确率	Recall 召回率	F1 Score F1 分数
4	0.9851	0.9600	**1.0000**	0.9796
8	**0.9884**	**0.9916**	0.9938	**0.9927**
16	0.9371	**1.0000**	0.9239	0.9610
32	0.9732	**1.0000**	0.8250	0.9041

4.5. Ablation Study 4.5. 消融研究

This section presents the results of the ablation study to evaluate the importance of incorporating character-based URL (Computer Vision), word-based URL (Natural Language Processing), and HTML DOM graph (Graph Neural Network) features in the DeepPhish-X phishing detection model.[1] Table 7 summarizes the performance metrics (accuracy, precision, recall, and F1 score) for various configurations of the model, where different combinations of the three feature types are used.[1] It was observed that the highest performance was achieved when all three features were used together, suggesting that incorporating each feature is crucial for effective phishing detection.[1]

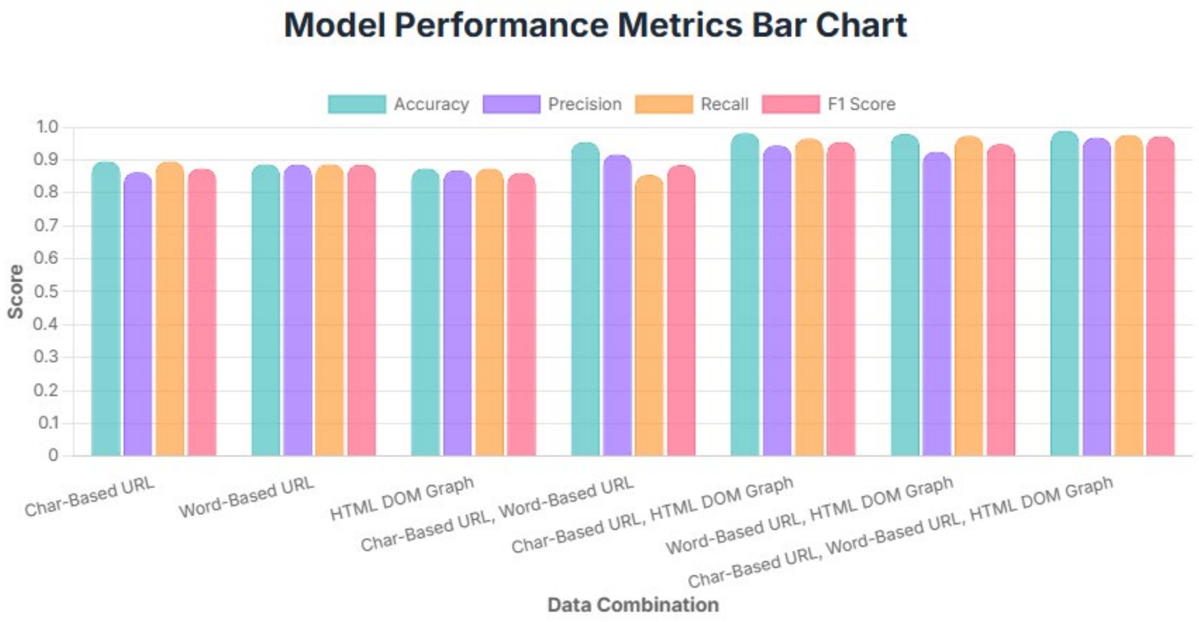
本节展示了消融研究的结果，以评估在 DeepPhish-X 钓鱼检测模型中整合基于字符的 URL

（计算机视觉）、基于词的 URL（自然语言处理）和 HTML DOM 图（图神经网络）特征的重要性。[1] 表 7 总结了模型不同配置的性能指标（准确率、精确率、召回率和 F1 分数），其中使用了三种特征类型的不同组合。[1] 观察到当三种特征全部使用时，性能最高，这表明整合每种特征对于有效的钓鱼检测至关重要。[1]

Table 7: Performance Metrics for Different Feature Combinations in the Ablation Study [1]

表 7：消融研究中不同特征组合的性能指标 [1]

Data 数据	Accuracy 准确率	Precision 精确率	Recall 召回率	F1 Score F1 分数
Char-Based URL	0.8952	0.8626	0.8952	0.8736
Word-Based URL	0.8868	0.8859	0.8868	0.8856
HTML DOM Graph	0.8739	0.8684	0.8740	0.8605
Char-Based URL, Word-Based URL	0.9540	0.9167	0.8544	0.8844
Char-Based URL, HTML DOM Graph	0.9817	0.9436	0.9647	0.9541
Word-Based URL, HTML DOM Graph	0.9789	0.9245	0.9730	0.9481
Char-Based URL, Word-Based URL, HTML DOM Graph	**0.9884**	**0.9677**	**0.9759**	**0.9718**



4.6. t-SNE Visualization of Feature Integration

This section presents t-Distributed Stochastic Neighbor Embedding (t-SNE) visualizations to illustrate the effectiveness of integrating different features for phishing detection within DeepPhish-X.[1] The t-SNE algorithm reduces the dimensionality of the feature space, allowing for a clearer visual comparison of feature distributions.[1] Figure 5 depicts the t-SNE plots for different feature combinations.[1]

Figure 5: t-SNE Visualization of Feature Integration for Graph-Based and Non-Graph-Based Models in DeepPhish-X [1]

(Figure 5, showing t-SNE visualization, should be inserted here)

Figure 5a represents the data distribution using the URLNet method, which relies solely on character-based and word-based URL features.[1] The data points are scattered with some clustering, indicating that while URLNet captures certain phishing characteristics, it lacks robustness due to its exclusive reliance on URL-based features.[1] In Figure 5b, the data distribution is illustrated using the Texception method, which also depends on word-based URL features.[1] This method shows some clustering but falls short of achieving optimal phishing detection accuracy, likely due to its limited feature set.[1] Figure 5c presents the data distribution using DeepPhish-X, which utilizes only character-based and word-based URL features, deliberately excluding the HTML DOM graph structure.[1] However, the absence of HTML structure limits the model's ability to fully separate benign and phishing instances.[1]

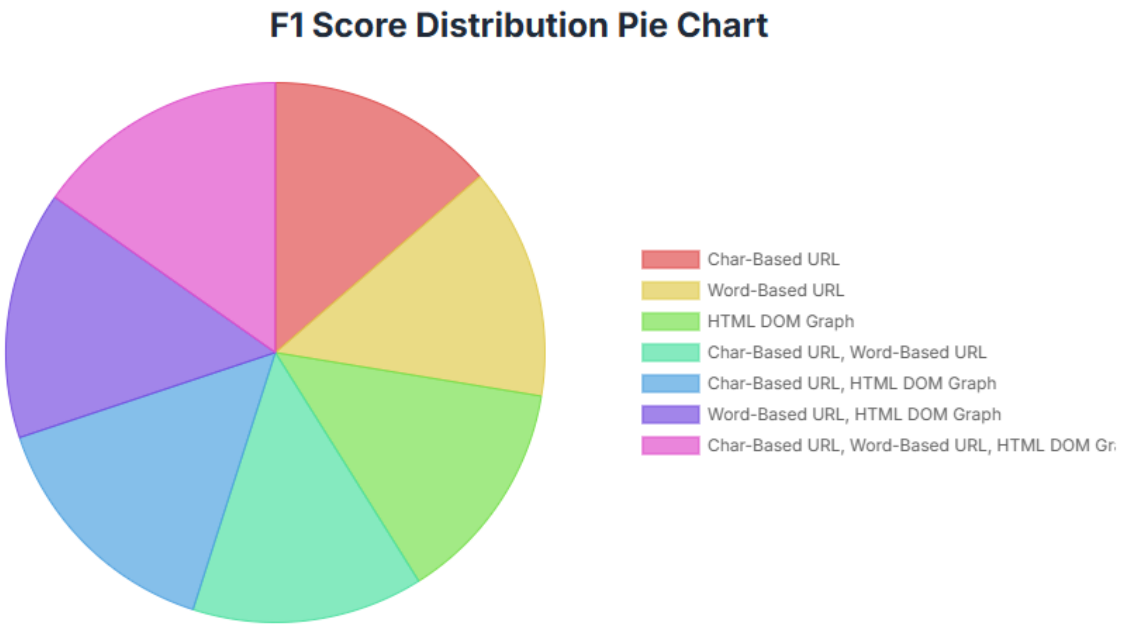


Figure 5d combines character-based URL features (Computer Vision) with the HTML DOM graph structure (Graph Neural Network) as part of DeepPhish-X.[1] The clustering is distinct, demonstrating the effectiveness of integrating these URL features with the HTML structure.[1] However, the clusters are not completely separated, indicating that while the integration improves detection, it is not yet optimal.[1] Similarly, Figure 5e combines word-based URL features (Natural Language Processing) with the HTML DOM graph structure (Graph Neural Network) using DeepPhish-X.[1] The clustering is more defined compared to previous cases, showing improved separation between benign and phishing instances.[1] This result underscores the value of combining word-based URL features with the HTML DOM graph.[1] Finally, Figure 5f employs DeepPhish-X, combining all three features: character-based URL (Computer Vision), word-based URL (Natural Language Processing), and HTML DOM graph (Graph Neural Network).[1] The clear and well-defined separation between clusters highlights the importance of integrating all three feature types for achieving the most accurate phishing detection.[1]

The comparison between these plots emphasizes the critical importance of using all three feature types together.[1] The clear separation seen in Figure 5d-f suggests that the combined feature set provides a more comprehensive representation of the data.[1]

4.7. Confusion Matrix Analysis

This section analyzes the performance of DeepPhish-X through the confusion matrix, as shown in Table 8.[1] The best performance for each metric is highlighted in bold.[1] The confusion matrix provides a detailed breakdown of the model's predictions, highlighting the number of true positives, true negatives, false positives, and false negatives.[1]

Table 8: Confusion Matrix of the DeepPhish-X Ensemble Model [1]

Predicted	Benign	Phishing	Recall
Actual			
Benign	5907	50	0.9916
Phishing	37	1497	0.9759
Precision	0.9938	0.9677	**F1 Score: 0.9718**

The confusion matrix reveals several key insights into DeepPhish-X's performance:

- **True Positives (TP):** The model correctly identified 1497 phishing instances.[1] This high number of true positives indicates the model's effectiveness in detecting phishing attacks.[1]
- **True Negatives (TN):** The model correctly classified 5907 benign instances.[1] The high true negative count demonstrates the model's accuracy in identifying legitimate webpages.[1]
- **False Positives (FP):** There were 50 benign instances incorrectly classified as phishing.[1] Although this number is relatively low, it highlights the importance of further improving the model to minimize false alarms.[1]
- **False Negatives (FN):** The model incorrectly identified 37 phishing instances as benign.[1] This number, while also low, underscores the need for continuous improvement to ensure that phishing attacks are not missed.[1]

In conclusion, the confusion matrix analysis shows that DeepPhish-X excels at distinguishing between benign and phishing webpages, with high precision, recall, and F1 scores.[1] These results underscore the model's robustness in enhancing phishing detection capabilities.[1]

4.8. Generalizability Evaluation on Unseen Phishing Data

The primary focus of this study is on accurately identifying phishing websites, with a particular emphasis on minimizing false negatives.[1] To evaluate the generalizability of DeepPhish-X, an additional experiment was conducted using a completely new phishing dataset that was not part of the original training set.[1] For this, 14,573 phishing URLs and their corresponding HTML documents were collected from Mendeley Data.[1] The model, which had been previously trained on the original dataset, was tested on this new phishing data without any further fine-tuning or adjustment of the model's weights.[1]

DeepPhish-X correctly identified 13,919 out of 14,573 phishing instances, resulting in an accuracy of approximately 95.5%.[1] This high accuracy indicates that the model is capable of effectively generalizing to unseen phishing data, maintaining its strong performance even when exposed to phishing tactics that were not included in the training phase.[1] By testing exclusively on phishing data, this experiment aimed to assess the model's robustness in real-world scenarios where detecting phishing attempts is critical.[1] These results suggest that DeepPhish-X is well-suited to generalize across different phishing examples, reinforcing its potential application in diverse phishing detection environments.[1]

Future work could extend this evaluation by incorporating legitimate websites into the test dataset to further validate the model's generalizability across different types of content.[1] Additionally, to further validate the robustness of the model, future research might involve testing on additional unseen phishing datasets from diverse sources to ensure the model's generalizability across different phishing strategies and tactics.[1]

4.9. Robustness against Adversarial Attacks

In addition to evaluating DeepPhish-X's performance under normal conditions, this study conducted experiments to assess its robustness against adversarial attacks.[1] Specifically, the Fast Gradient Sign Method (FGSM) was applied to generate adversarial examples by introducing small perturbations to the input data.[1] These perturbations were designed to test the model's ability to maintain accuracy when faced with adversarially altered inputs.[1]

This study tested DeepPhish-X with various epsilon values (ε), ranging from 0 (no perturbation) to 0.1 (significant perturbation).[1] The results of these tests are summarized in Table 9.[1]

Table 9: Adversarial Attack Performance Results (Macro Average) [1]

Epsilon (ε)	Accuracy	Precision	Recall	F1 Score

0	0.9873	0.9875	0.9894	0.9884
0.02	0.9732	0.9466	0.9754	0.9608
0.04	0.9389	0.8841	0.9451	0.9136
0.06	0.8767	0.7994	0.9004	0.8435
0.08	0.7468	0.6981	0.8101	0.7494

As seen from the results, DeepPhish-X's performance begins to degrade as the epsilon value increases.[1] With no perturbation (ε = 0), the model achieves 98.73% accuracy, with high macro-averaged precision, recall, and F1-score values.[1] However, even a small perturbation (ε = 0.02) reduces the accuracy to approximately 97.32%, with a noticeable decline in performance, particularly in its ability to correctly classify the benign class.[1]

As the perturbation becomes more significant (ε = 0.04 and above), the model's accuracy drops further.[1] At ε = 0.08, the model's accuracy falls to 74.68%, with the confusion matrix indicating that the model is heavily biased towards misclassifying benign samples as phishing.[1] These findings highlight DeepPhish-X's vulnerability to adversarial attacks, particularly when small but targeted perturbations are applied.[1] This underscores the need for incorporating more robust defense mechanisms in the model, such as adversarial training or other forms of regularization, to enhance its resilience against such attacks.[1]

4.10. Discussion: Case Analysis

This section analyzes specific cases to understand DeepPhish-X's performance, particularly focusing on instances where the model correctly and incorrectly classified URLs, as detailed in Table 10.[1] This analysis provides insights into the strengths and limitations of this approach.[1]

Table 10: Detailed Case Analysis of Correct and Incorrect URL Classifications by DeepPhish-X [1]

Classification Result	Case	Ground Truth	URL
Correctly classified	(a)	Benign	https://ninecloud.ae/our-services/interior-exterior-paint-contractors-in-dubai/

			(accessed on 19 August 2024)
	(b)	Phishing	https://pub-88f64e013ca94e82aa5d15393134722c.r2.dev/logs.html (accessed on 19 August 2024)
Misclassified	(c)	Benign	https://rilm.am/wp-content/uploads/2022/07/12e47ac82164e89a8c15f399384e6572.pdf (accessed on 19 August 2024)
	(d)	Phishing	https://amangroup.co/gy/linkedin_/ (accessed on 19 August 2024)

Figure 6: Comparative HTML DOM Graph Visualizations for Benign and Phishing Case Analysis
[1]

(Figure 6, showing DOM structure visualization, should be inserted here)
In Figure 6a, the URL "https://ninecloud.ae/our-services/interior-exterior-paint-contractors-in-dubai/" was correctly classified as benign.[1] The DOM graph visualization (Figure 6a) shows a clear and simple structure, which likely contributed to the model's accurate classification.[1] The URL's direct and legitimate appearance, along with the coherent HTML structure, aligns well with benign patterns the model has learned.[1] In Figure 6b, the URL "

https://pub-88f64e013ca94e82aa5d15393134722c.r2.dev/logs.html" was correctly classified as phishing.[1] As depicted in the DOM graph visualization (Figure 6b), the URL presents a complex and suspicious structure.[1] The presence of random characters and a deceptive path indicates phishing characteristics, which the model successfully identified.[1]

In Figure 6c, the URL "https://rilm.am/wp-content/uploads/2022/07/12e47ac82164e89a8c15f399384e6572.pdf" was incorrectly classified as phishing.[1] The DOM graph visualization (Figure 6c) shows a complex structure, which might have misled the model.[1] Despite being a benign URL, its intricate and lengthy format may resemble phishing patterns, leading to a false positive.[1] This highlights the challenge of distinguishing between complex legitimate URLs and phishing URLs.[1] In Figure 6d, the URL " https://amangroup.co/gy/linkedin_/" was incorrectly classified as benign.[1] The DOM graph visualization (Figure 6d) shows a relatively simple structure.[1] However, this simplicity might have contributed to the model's failure to recognize it as phishing.[1] The deceptive use of familiar keywords like "linkedin" might have made the URL appear legitimate, resulting in a false negative.[1] This analysis underscores the importance of further refining DeepPhish-X to better distinguish between subtle phishing indicators and legitimate but complex URL structures.[1]

A major issue is that the model only used the DOM structure, which, while improving the model's performance overall, presents a challenge when the DOM structure is too simple or resembles that of a legitimate webpage.[1] Therefore, it is necessary to incorporate additional HTML features, such as HTML DOM tag names and hyperlinks, to improve phishing detection.[1] Additionally, the reliance on static features extracted from URLs and HTML DOM structures may be susceptible to obfuscation by constantly evolving phishing tactics, potentially reducing the model's effectiveness over time.[1] Moreover, some benign URLs with complex structures were misclassified as phishing because they resembled phishing patterns in the HTML DOM.[1] This highlights the need for incorporating contextual information, such as user behavior or dynamic content analysis, to enhance detection accuracy.[1] Furthermore, the integration of multi-modal features improved detection rates but also increased the computational complexity of the model, which could be a limitation in real-time applications where processing speed is crucial.[1]

The findings of this study contribute to the broader field of phishing detection by demonstrating the effectiveness of integrating multi-modal features (such as HTML DOM structures and URL characteristics) to improve detection accuracy.[1] These results suggest that combining different data sources can capture more comprehensive patterns associated with phishing attempts.[1] For future research, exploring the integration of user interaction data and behavioral analytics could provide deeper insights into phishing tactics, offering opportunities to develop more adaptive and robust detection systems.[1] Additionally, investigating the application of real-time analysis techniques and leveraging advances in adversarial learning could further enhance DeepPhish-X's resilience against sophisticated phishing attacks.[1]