A Semantic-Aware Approach to Phishing URL Detection

Trong Thua Huynh
Information Security Technology Lab, Posts and Telecommunications Institute of Technology, Vietnam
thuaht@ptit.edu.vn

Hoang Thanh Nguyen
Information Security Technology Lab, Posts and Telecommunications Institute of Technology, Vietnam
thanhnh@ptit.edu.vn (corresponding author)

Nhat Huynh Tran
Faculty of Information Technology, Posts and Telecommunications Institute of Technology, Vietnam
n20dcat023@student.ptithcm.edu.vn

Received: 7 July 2025 | Revised: 23 August 2025 | Accepted: 2 September 2025

 DOI: https://doi.org/10.48084/etasr.13187

ABSTRACT

With the rapid growth of online transactions, phishing attacks have become increasingly unpredictable, particularly those involving malicious Uniform Resource Locators (URLs). Currently, few studies have effectively combined manual and semantic features within a unified framework to both fully utilize the structural information and capture the deep context dependence of the character string. To address this gap, this study proposes a deep attention-based approach for phishing URL detection. Our approach first relies on the importance of manual features and the Bidirectional Encoder Representations from Transformers (BERT) model to extract semantic feature vectors. Subsequently, a hybrid deep learning architecture comprising Bidirectional Long Short-Term Memory (BiLSTM) layers, an attention mechanism, and fully connected dense networks is employed to classify URLs as either phishing or legitimate. Experimental results demonstrate that our proposed model achieves a high classification accuracy of 96.77%, while ablation analysis highlights the individual contributions of key components, including BERT embeddings, attention mechanism, LSTM layers, and feature types, to overall model performance. Finally, by training on a Kaggle benchmark dataset and testing on real-world phishing samples, the study confirms the model's strong generalization capability in detecting emerging phishing threats.

Keywords-phishing Uniform Resource Locators (URL); Bidirectional Long Short-Term Memory (BiLSTM); attention mechanism; Bidirectional Encoder Representations from Transformers (BERT); semantic feature

I. INTRODUCTION

Phishing remains a fast-evolving threat that continually outpaces blacklists and rule-based filters. While these defenses are inexpensive and easy to deploy, their effectiveness quickly degrades as adversaries rotate domains, obfuscate strings, and craft zero-day Uniform Resource Locators (URLs) absent from reputation feeds [1-3]. Consequently, research has shifted to learning-based detection that infers discriminative patterns from the URL string and auxiliary metadata; yet robust generalization across datasets and campaigns is still challenging due to distribution shift, brittle features, and evaluation practices that favor single-dataset tuning [2, 3].

Prior studies in phishing URL detection fall into three main strands: (i) Feature-engineering with classical machine learning leveraging lexical and host-level cues (e.g., length/entropy, symbol counts, path/query structure) in combination with models such as Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), or Extreme Gradient Boosting (XGBoost). Representative efforts, including the studies in [4-7], achieve strong in-dataset accuracy through careful feature selection, often guided by importance metrics or wrapper-based methods. These approaches are efficient and interpretable but rely heavily on dataset-specific heuristics, hindering transferability. (ii) Deep sequence and convolutional models over raw URLs, such as Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), or Bidirectional LSTM (BiLSTM) variants, which capture local character n-grams and longer dependencies without manual feature engineering. These models perform competitively, even under class imbalance, yet remain sensitive to tokenization, context length, and the absence of semantic context beyond characters [8, 9]. (iii) Pre-trained language modeling for URLs, such as Bidirectional

Encoder Representations from Transformers (BERT)-style representations with downstream classifiers, which encode richer dependencies and are frequently benchmarked on the widely used Phishing Site URLs dataset, useful for comparability but exposing gaps in cross-dataset robustness [10, 11].

To address these limitations, we propose a semantic-aware hybrid model that unifies (i) BERT embeddings to encode sub-token patterns in paths and parameters, (ii) BiLSTM with attention to focus on informative subsequences, and (iii) a compact, deployment-friendly set of 20 hand-crafted URL features distilled from classical practice. Unlike feature-only approaches that depend on dataset-specific lexical/host cues [4-7], injecting semantic context reduces brittleness to obfuscation. Compared to character-only sequence models [8, 9], coupling BiLSTM and attention with pretrained representations improves sensitivity to meaningful substrings while controlling overfitting. In contrast with general end-to-end benchmarks centered on the Phishing Site URLs corpus [10, 11], we prioritize cross-dataset evaluation and systematic ablation to quantify each block's contribution and validate generalization.

II. METHODOLOGY

This study provides a novel method to detect phishing URLs using manual URL features and semantic BERT language model attributes. The manual features and BERT embedding are then fed into an architecture with a BiLSTM for sequence relation mining, an attention mechanism for focusing on important semantic parts, a dense network for reducing dimensions and learning the final representation, and a fully connected classification layer.

A. Data Preprocessing

The dataset used in this study is the Phishing Site URLs [11], consisting of 549,346 URL samples. Among these, 392,924 samples correspond to legitimate URLs and 156,422 to phishing URLs.

1) Manual Feature Extracting

In this work, we only examine the formal aspects of the URL string, excluding the web page's content, such as HyperText Markup Language (HTML), images, and Document Object Model (DOM), to simplify implementation.

Our work utilized the 41 features described in [12] to identify the best URL classification features. Then we employed the RF approach to assess the relevance of each piece of input information, which sped up calculations for identifying the best features. After training RF on all 41 features, we utilize (1) to identify the Gini importance for each feature j:

$$Imp(j) = \frac{\sum_{s\,:\,split\ on\ j} \Delta i(s)}{\sum_{s\,\in\,all\ splits} \Delta i(s)} \quad (1)$$

where the impurity reduction at a split is calculated by (2):

$$\Delta i(s) = i(t) - \frac{N_L}{N} i(t_L) - \frac{N_R}{N} i(t_R)$$
$$i(\cdot) = 1 - \sum_k p_k^2 \quad (2)$$

where $i(\cdot)$ is the Gini impurity, p_k is the proportion of the class k samples at the corresponding node, t is the parent node, t_L, and t_R are the left and right child nodes, while N, N_L, and N_R are their respective sample counts. Table I summarizes the 20 most influential features with their corresponding Gini importance scores.

TABLE I. 20 SELECTED MANUAL FEATURES

Feature	Description	Type	Gini score
url_length	URL length	Numeric	0.096
number_of_dots	number of dots in URL	Numeric	0.074
repeated_digits	URL contain repeated digits?	Boolean	0.057
number_of_digits	number of digits	Numeric	0.043
special_chars	number of special characters	Numeric	0.056
hyphens	number of hyphens	Numeric	0.039
underlines	number of underline	Numeric	0.034
slashes	number of slashes	Numeric	0.031
question_marks	number of "?"	Numeric	0.027
equals	number of equals	Numeric	0.032
at_symbols	number of "@"	Numeric	0.032
dollar_signs	number of "$"	Numeric	0.039
exclamations	number of "!"	Numeric	0.027
hashtags	number of "#"	Numeric	0.051
percent_signs	number of "%"	Numeric	0.036
path_length	path length	Numeric	0.058
having_query	Does URL contain a query?	Boolean	0.032
having_fragment	Does URL contain a fragment?	Boolean	0.026
having_anchor	Does URL contain an anchor?	Boolean	0.031
entropy_of_url	Shannon entropy	Continuous	0.087

2) Extracting Semantic Feature Vectors

This study embeds semantic-based features using the BERT paradigm. Its bidirectional encoding capability enables contextual understanding of textual input. BERT also converts URL strings into semantic feature vectors suitable for deep learning models. In this work, the SentenceTransformer package and the lighter BERT model distilbert-base-nli-mean-tokens [13] are used to embed URLs. The library facilitates tokenization, embedding generation, and aggregation into a 768-dimensional vector representation for each URL. The main steps are as follows:

- Tokenization: Punctuation marks segment the URL into tokens, while special tokens such as [CLS] and [SEP] are appended to indicate the start and end of the sequence, respectively.
- Sequence Formatting: Each URL is stored as a token string of up to 512 tokens. If the token string is shorter than 512, [PAD] tokens are added to reach the maximum length. Overlong strings are clipped.
- Attention Masking: Each token is mapped to a corresponding input ID from the BERT vocabulary. An attention mask is applied to distinguish padding tokens (mask = 0) from target tokens (mask = 1).
- Embedding Generation: Following attention masking, the BERT model transforms the tokenized URLs into 768-dimensional semantic feature vectors. Mean pooling is then applied across all token embeddings, excluding [PAD]

tokens. The resulting embedding vectors are stored in a data frame, where each row represents a 768-dimensional vector paired with its corresponding label.

B. Proposed Architecture

In the proposed deep learning model, the embedding vectors are first fed into the BiLSTM layer to capture the semantic relations of the data in both directions. The BiLSTM output is then processed through an attention layer, allowing the model to automatically focus on the most informative regions. In addition, a dense network processes the numeric features of the URL. Finally, the results from the attention layer and the extra features from the dense network are combined and passed through a fully connected layer to classify URLs as either phishing or legitimate. Figure 1 illustrates the main components of the proposed architecture.

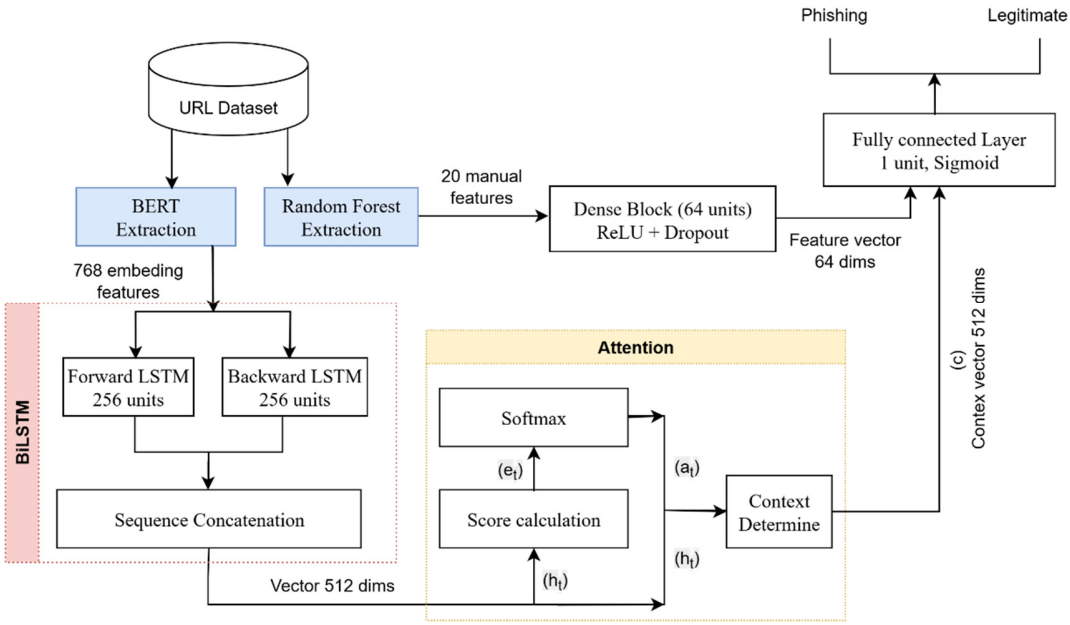

Fig. 1. The proposed architecture.

1) BiLSTM

BiLSTM [14] is a variant of LSTM networks that enables learning from both past and future contexts. It consists of two LSTM layers: one processes the input sequence from left to right, and the other from right to left. By combining the outputs of both directions, the model captures richer contextual information. For an input sequence $X = [x_1, x_2, \dots, x_T]$, the BiLSTM calculates $h_t(f) = LSTM_f(x_1, \dots, x_t)$ in the forward direction and $h_t(b) = LSTM_b(x_T, \dots, x_t)$ in the backward direction. The combined output at each timestep is

$$h_t = [h_t(f); h_t(b)] \in R^{2d} \quad (3)$$

where d is the number of hidden dimensions of a single LSTM; in this study, $d = 256$. This 256-dimensional sequence output is subsequently input to the attention mechanism, enabling the model to emphasize critical positions within the feature sequence rather than relying solely on the final state.

2) Attention Mechanism

The attention mechanism allows the model to automatically assign higher weights to the most relevant parts of the sequence, enhancing the context representation [15]. The attention layer receives the BiLSTM outputs h_t and computes:

- Score calculation:

$$e_t = v^T \tanh(W_h h_t + b_h) \quad (4)$$

where W_h and b_h are the learnable parameters, and v is the learnable weight vector.

- Softmax, which converts the e_t scores into attention weights according to (5):

$$a_t = \frac{\exp(e_t)}{\sum_{k=1}^{T} \exp(e_k)} \quad (5)$$

- Context vector, which calculates the weighted sum of the hidden states as:

$$c = \sum_{t=1}^{T} a_t h_t \quad (6)$$

The resulting context vector c is the vector representing the synthesized context, which is the input for the next dense layer. With this mechanism, the model will automatically determine which position (timestep) is the most important in the BiLSTM output sequence to focus on learning, thereby improving the accuracy and explainability of the classification results.

3) Dense Block

Dense blocks are fully connected layers where each neuron is connected to all neurons in the next layer [16]. In this model, a dense network processes numeric URL features to extract complex patterns before merging them with BERT-derived

semantic representations. The context vector $c \in R^{256}$ is transformed via an affine mapping and nonlinear activation:

$$h = ReLU(W_d c + b_d) \quad (7)$$

where $W_d \in R^{64\times20}$, $b_d \in R^{64}$ are the learned parameters, while the ReLU activation function helps the model learn nonlinear representations. Dropout is employed just after the ReLU layer with a ratio of $p = 0.5$ to reduce overfitting by randomly shutting off 50% of the neurons during training. The output is a vector $h \in R^{64}$ representing the last feature provided to the fully connected layer for classification. Dimensionality reduction from 128 to 64 decreases parameter count, conserving memory and computation while emphasizing the most informative features.

4) Fully Connected Layer

In this model, the input to the fully connected layer is a mix of the attention block's context vector (512 dimensions) and the dense network's output (64 dimensions) that processes the URL features. The input vector will have 576 dimensions after concatenation. A Sigmoid activation is applied to produce the final binary prediction:

$$\hat{y} = \sigma(W_h + b) \quad (8)$$

where $W \in R^{1\times576}$ is the weight matrix, and bias $b \in R$.

Overall, the architecture addresses prior limitations along three axes: (i) a BERT-based semantic representation layer mitigates distribution shift commonly observed in feature-engineered pipelines [4-7]; (ii) a BiLSTM with attention augments URL-sequence-only models by locating and reweighting salient spans [8, 9]; and (iii) a compact set of 20 features preserves inference efficiency while retaining core structural signals, avoiding the overparameterization trend seen in some prior work [4-11].

C. Training and Evaluation

The dataset was partitioned into training (70%), validation (15%), and test (15%) subsets. Validation data is used to tune hyperparameters and monitor performance to prevent overfitting. Table II summarizes the main training parameters. After training, the model is evaluated on the test set to determine the final accuracy. To evaluate the model, the metrics used are as shown in Table III.

The model was trained on a Dell R730 server equipped with NVIDIA Tesla P40 Graphics Processing Unit (GPU) accelerators with 24 GB of Random Access Memory (RAM) each, two Xeon E2680v4 Central Processing Units (CPUs), and 128 GB of DDR4 2,400 MT/s RAM.

TABLE II. MAIN PARAMETERS USED IN TRAINING

Parameter	Value	Description
batch_size	16	Batch size used in training.
learning_rate	2e-5	Learning rate for the AdamW.
epoch	50	Number of iterations with early stop.
optimizer	AdamW	Optimization algorithm with weight decay.
criterion	BCELoss	Loss function used for binary with Sigmoid.
dropout	0.1	The dropout rate helps reduce overfitting.

TABLE III. METRICS FOR MODEL EVALUATION

Metric		Description
$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$	(9)	Ratio of correctly classified URLs.
$\text{Precision (Prec)} = \frac{TP}{TP+FP}$	(10)	Proportion of predicted phishing URLs that are actual phishing.
$\text{Recall (Re)} = \frac{TP}{TP+FN}$	(11)	Ability to correctly identify actual phishing URLs.
$F1-\text{score} = 2 \cdot \frac{Prec \cdot Re}{Prec+Re}$	(12)	Harmonic mean of Precision and Recall.

TP: True Positives, TN: True Negatives, FP: False Positives, FN: False Negatives

III. RESULTS AND DISCUSSION

The results in Table IV indicate that the model demonstrated strong generalization capacity, as the accuracy between the training and validation sets differs by only 0.36%. The low loss values in both sets further confirm that the model effectively learns the relationship between features and labels without overfitting.

TABLE IV. KEY METRICS AFTER TRAINING

Accuracy (%)		Loss		Training time (m)	Parameters (M)
Train	*Validation*	*Train*	*Validation*		
97.10	96.74	0.0803	0.0888	198	2.5

When evaluated on the test set, the model achieved an accuracy of 96.77% (Table V). Specifically, for the phishing class, the Precision, Recall, and F1-score metrics all exceed 90%, indicating that the proposed method not only achieves high classification accuracy but also minimizes false negatives and false positives. This is further demonstrated in Figure 2, where the proposed model was exceptionally effective at detecting phishing URLs (TP = 15,674). Although some legitimate URLs were misclassified as phishing (FP = 993, or 1.3%) and a small number of phishing URLs were missed (FN = 1,471, or 1.9%), these error rates are relatively low given the dataset size.

TABLE V. MODEL EVALUATION ON TEST SET

Accuracy (%)	Prec (%)	Re (%)	F1-score (%)	Prediction time (ms)
96.77	94.8	90.66	92.68	13

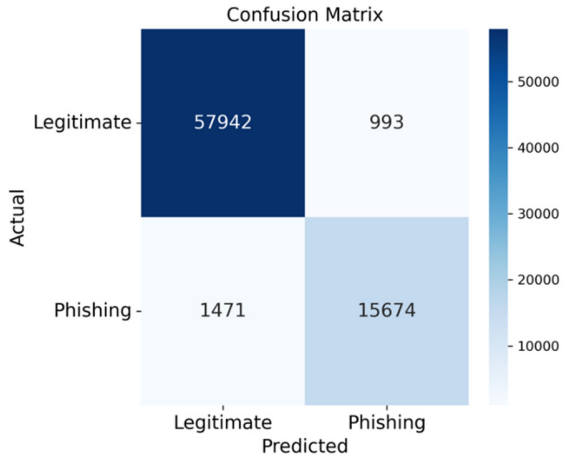

Fig. 2. Confusion matrix of the proposed model.

Furthermore, Figure 3 presents the Receiver Operating Characteristic Area Under the Curve (ROC-AUC) curve with an area of 0.95. The ROC curve shows that as the False Positive Rate (FPR) increases, the true positive rate (Recall) also rises, with the model maintaining high discrimination and minimal false classification errors.

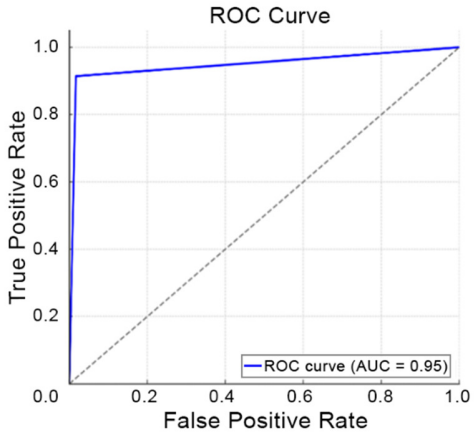

Fig. 3. ROC-AUC of the proposed model.

Compared with previous studies (Table VI), our model outperformed Linear Support Vector Classifier (SVC) (91%) [7], BERT + CNN (96.66%) [9], and RoBERTa + DistilRoBERTa (94.22%) [10], and approached the top performance of the large LSTM model (98.1%) [5]. Notably, the prediction time is significantly improved compared with [9] and [10], while remaining comparable to Linear SVC [7]. This balance between accuracy and speed highlights the practicality of the proposed approach.

TABLE VI. COMPARISON OF ACCURACY AND PREDICTION TIME BETWEEN MODELS

Model	Year	Accuracy (%)	Method	Prediction time (ms)
[5]	2024	98.1	LSTM	67134
[7]	2023	91	Linear SVC	12
[9]	2022	96.66	BERT + CNN	64
[10]	2025	94.22	RoBERTa + DistilRoBERTa	41
Our proposal	2025	96.77	BERT + BiLSTM with Attention	13

To further evaluate the contribution of each model component, ablation experiments were conducted (Table VII). As shown in Table VII, using only 20 manual features yielded an accuracy of 94.5% and an F1-score of 90.26%, slightly lower than using the full 41 features, resulting however to lower computation times, striking a balance between precision and speed while making the model compact and easy to deploy. When using only semantic features, the model achieved 95.83% accuracy and an F1-score of 91.49%, confirming the effectiveness of semantic embeddings. Combining manual and semantic features without a sequence model slightly improved performance to an accuracy of 96.12% and an F1-score of 91.94%. Incorporating BiLSTM increased accuracy and F1-score by 0.30 and 0.36, respectively. Finally, adding the Attention mechanism further enhanced performance, resulting in the best-performing model. Overall, semantic embeddings, sequence modeling, attention, and manual features each contribute significantly, and only their combination achieves optimal performance and generalization.

TABLE VII. ABLATION COMPARISON

Dataset	Accuracy (%)	F1-score (%)	Train time (m)
Only 41 manual features.	94.80	90.54	36
Only 20 manual features.	94.50	90.26	23
Only semantic features (BERT, BiLSTM with Attention).	95.83	91.49	175
Manual and semantic features (No BiLSTM, without Attention).	96.12	91.94	147
Manual and semantic features (BiLSTM without Attention).	96.42	92.30	189
Full (Manual and semantic features, BiLSTM with Attention).	96.77	92.68	198

To further assess the generalizability and effectiveness of detecting new phishing campaigns, the model was tested on three external datasets distinct from the training set [11]: the Malicious URLs dataset [17], Phishing URL dataset [18], and Benign and Malicious URLs dataset [19]. Dataset [17] includes 651,191 URLs (428,103 benign, 96,457 malicious, 94,111 phishing, and 32,520 malware). Dataset [18] contains 450,176 URLs (104,438 phishing and 345,738 legitimate), and dataset [19] includes 632,508 URLs equally divided between benign and phishing. From each dataset, 5% of the URLs were randomly sampled, yielding 65,119, 45,017, and 63,250 samples, respectively, for testing. The results are presented in Table VIII.

TABLE VIII. CROSS-DATASET EVALUATION

Dataset	# of URLs tested	Accuracy (%)	Prec (%)	Re (%)	F1-score (%)
Phishing Site URLs [11]	82,401	96.77	94.8	90.66	92.68
Malicious URLs dataset [17]	65,119	95.34	92.10	88.45	90.22
Phishing URL dataset [89]	45,017	96.12	93.75	89.92	91.80
Benign and Malicious URLs [19]	63,250	95.89	93.40	89.55	91.45

The model yielded accuracies of 95.34%, 96.12%, and 95.89% in the external datasets [17], [18], and [19], respectively. The largest drop in accuracy relative to the training data was only 1.43% (from 96.77% to 95.34%), while the others decreased by just 0.65% and 0.88%. These results demonstrate that the proposed model maintains strong recognition performance across diverse URL distributions, confirming its robustness, generalization capability, and suitability for real-world phishing detection applications.

IV. CONCLUSION

This paper presented a semantic-aware hybrid model for phishing Uniform Resource Locators (URLs) detection that

integrates Bidirectional Encoder Representations from Transformers (BERT) embeddings, a Bidirectional Long Short-Term Memory (BiLSTM) with attention, and a compact set of 20 hand-crafted URL features within a unified framework. The main novelty of this work lies in the unified semantic-structural modeling; unlike most prior works, the proposed system jointly integrates pretrained semantic representations with sequence modeling and lightweight manual features in an end-to-end fashion.

On the held-out test set, the model achieved 96.77% accuracy, 94.80% precision, 90.66% recall, and 92.68% F1-score for the phishing class, with a Receiver Operating Characteristic Area Under the Curve (ROC-AUC) of 0.95 and an average inference time of approximately 13 ms per URL, demonstrating both high effectiveness and practical efficiency suitable for real-time filtering applications. The ablation study revealed consistent improvements when successively adding semantic embeddings, BiLSTM layers, and attention mechanisms, with the complete configuration achieving the optimal balance between accuracy and latency. Moreover, the model demonstrated strong generalization across datasets, trained on the Kaggle Phishing Site URLs corpus and evaluated on three external datasets, achieving accuracies of 95.34%, 96.12%, and 95.89%, with a maximum reduction of only 1.43% compared to in-distribution testing, confirming robustness to distributional shifts.

Limitations include relatively high training time and model size; future research will investigate model compression and knowledge distillation techniques, continual learning to adapt to evolving phishing tactics, deeper analysis of challenging cases such as homograph and obfuscation attacks, and large-scale evaluation on continuously updated URL streams.

ACKNOWLEDGMENT

This research is supported by the Posts and Telecommunications Institute of Technology (PTIT).

REFERENCES

[1] S. Kavya and D. Sumathi, "Staying ahead of phishers: a review of recent advances and emerging methodologies in phishing detection," *Artificial Intelligence Review*, vol. 58, no. 2, Dec. 2024, Art. no. 50, https://doi.org/10.1007/s10462-024-11055-z.

[2] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection." arXiv, 2018, https://doi.org/10.48550/ARXIV.1802.03162.

[3] A. Abuadbba *et al.*, "Towards Web Phishing Detection Limitations and Mitigation." arXiv, 2022, https://doi.org/10.48550/ARXIV.2204.00985.

[4] S. A. Murad, N. Rahimi, and A. J. Md Muzahid, "PhishGuard: Machine Learning-Powered Phishing URL Detection," in *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*, Las Vegas, NV, USA, Jul. 2023, pp. 2279–2284, https://doi.org/10.1109/CSCE60160.2023.00371.

[5] R. Ferdaws, "Machine Learning and Deep Learning for Phishing Site URL Classification," Master of Science, California State University San Marcos, San Marcos, CA, USA, 2024.

[6] A. A. Albishri and M. M. Dessouky, "A Comparative Analysis of Machine Learning Techniques for URL Phishing Detection," *Engineering, Technology & Applied Science Research*, vol. 14, no. 6, pp. 18495–18501, Dec. 2024, https://doi.org/10.48084/etasr.8920.

[7] D. Kalla and S. Kuraku, "Phishing Website URL's Detection Using NLP and Machine Learning Techniques," *Journal on Artificial Intelligence*, vol. 5, pp. 145–162, 2023, https://doi.org/10.32604/jai.2023.043366.

[8] K. S. Jishnu and B. Arthi, "Phishing URL Detection Using BiLSTM With Attention Mechanism," in *Machine Intelligence Applications in Cyber-Risk Management*, M. A. Almaiah and Y. Maleh, Eds. IGI Global, 2024, pp. 159–184.

[9] M. Elsadig *et al.*, "Intelligent Deep Machine Learning Cyber Phishing URL Detection Based on BERT Features Extraction," *Electronics*, vol. 11, no. 22, Nov. 2022, Art. no. 3647, https://doi.org/10.3390/electronics11223647.

[10] P. H. Hussan and S. M. Mangj, "BERTPHIURL: A Teacher-Student Learning Approach Using DistilRoBERTa and RoBERTa for Detecting Phishing Cyber URLs," *Journal of Future Artificial Intelligence and Technologies*, vol. 1, no. 4, pp. 417–428, Feb. 2025, https://doi.org/10.62411/faith.3048-3719-71.

[11] *Phishing Site URLs*. (2020), T. Tiwari. [Online]. Available: https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls.

[12] M. A. Tamal, M. K. Islam, T. Bhuiyan, and A. Sattar, "Dataset of suspicious phishing URL detection," *Frontiers in Computer Science*, vol. 6, Mar. 2024, Art. no. 1308634, https://doi.org/10.3389/fcomp.2024.1308634.

[13] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks." arXiv, 2019, https://doi.org/10.48550/ARXIV.1908.10084.

[14] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997, https://doi.org/10.1109/78.650093.

[15] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate." arXiv, 2014, https://doi.org/10.48550/ARXIV.1409.0473.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Cambridge, Massachusetts: The MIT Press, 2016.

[17] *Malicious URLs dataset*. (2021), M. Siddhartha. [Online]. Available: https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset.

[18] J. K. S. Kaitholikkal, "Phishing URL dataset." Mendeley Data, Apr. 2024, https://doi.org/10.17632/VFSZBJ9B36.1.

[19] *Benign and Malicious URLs*. (2022), S. Malibari. [Online]. Available: https://www.kaggle.com/datasets/samahsadiq/benign-and-malicious-urls.