Minimax Theory and its Applications

Volume 11 (2026), No. 4s, 1080-1098

Detecting LLM-Generated Phishing URLs Using Deep Learning: Evaluating Model Robustness Against AI-Crafted Evasion Techniques

Esther Jyothi V[1*], Sneha Mishra [2], Bhavya Varma K[3], Manisha Semwal[4], Murthy V.S.N[5], Madhu Latha Veerapaneni[6]

[1]Senior Assistant Professor, Department of Computer Applications, Siddhartha Academy of Higher Education(Deemed to be University),Vijayawada.
[2]Assistant professor, Department of Computer Science and Engineering, Dr Akhilesh Das Gupta Institute of Professional Studies, New Delhi.
[3] Assistant professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Bowrampet, Hyderabad 500043, Telangana, India.
[4]Assistant professor, Department of Business Management, Graphic Era (Deemed to be University), Dehradun, Uttarakhand- 248002.
[5]Assistant professor, Department of Computer Science and Engineering, Nadimpalli Satyanarayana Raju Institute of Technology.
[6]Assistant professor, KL Business School, Koneru Lakshmaiah Education Foundation,Vaddeswaram, GunturDistrict, Andhra Pradesh – 522302.
Email: vejyothi@vrsiddhartha.ac.in, snehamishraphd24@gmail.com, bhavyavarmakalidindi@gmail.com, manishasemwal03@gmail.com, vsn.murthy87@gmail.com, madhulv@gmail.com

Abstract
The growing reliance on LLMs for the creation and modification of web content presents new challenges for the detection of phishing URLs, especially when malicious URLs have structural differences. While traditional detection models may perform well on known URL structures, they can be subject to adversarial manipulation and changes in data distributions. This work examines a weakness-driven evaluation of a phishing URL classifier in the scenarios of known structural perturbations, unprecedented evasion patterns, and external data conditions, by studying a robustness-aware character-level detection model on a processed dataset of 496,678 phishing URLs, including 151,097 phishing URLs and 345,581 legitimate URLs, as well as the external ASAPhishingURL_Dataset for cross-source evaluation.For RACA-Former, URLs are input as sequences of characters and a Transformer encoder learns contextual information between characters. A robustness-aware attention mechanism aggregates the character-level representation before the head that classifies a URL as phishing or legitimate. The evaluation includes clean testing, four types of structural perturbations, test on external dataset, held-out evasion, and an ablation comparison based on masked mean pooling. On the clean test set, the RACA-Former achieved an accuracy of 99.16%, precision of 99.45%, and recall of 97.79%. On the known perturbations, accuracy ranged from 97.56% to 99.25% with subdomain injection having the most degradation. In the ASA dataset, accuracy was 62.33% and specificity was 24.67%, despite a high phishing recall. On held-out subdomain injection, accuracy dropped to 33.20% and specificity to 5.62%. The ablation results show that neither RACA-Former's robustness-aware attention led to measurable gains. Overall, these results give a systematic evaluation that goes beyond performance on clean data, exposing persistent deficits in cross-source generalization and unseen evasion detection, raising the need for further research before deploying in realistic phishing situations.

Keywords*:* Character-Level Classification, Evasion Attacks, Phishing URL Detection, Robustness Evaluation, Transformer Encoder

1. Introduction

Phishing is a type of cyber-attack in which the attacker lures the user to a fake website with a clear request for sensitive information, such as account information or financial details [1]. The purpose of phishing URL detection is to identify a pattern of malicious URLs through a detection of the difference between a malicious URL and a legitimate URL [2]. Deep learning representation models have been used to assist in this process: it learn discriminative representations from URL information, which reduces the need for manually engineered lexical and structural features [3]. Character-level modelling, for example, treats a URL as a sequence of characters, allowing a model to capture patterns related to domain names, separators, paths, and other parts of the URL [4]. The relationships between characters in different positions in a URL can also be represented by transformer-based architectures. However, the structure of a URL can be deliberately modified to make a

phishing address appear different from previously observed examples [5]. Subdomain injection, URL padding, path injection and homoglyph substitution are techniques that change the components or sequences of characters in the URL. Such changes can impact the pattern that a detection model has trained and cause misclassification. Thus, the ability of a model to perform well on a set of unaltered URLs is not enough to establish its predictive behaviour in response to structural variation [6]. The detection performance under these special conditions can be investigated by controlled perturbation testing.

The study creates problems with the evaluation as the URL examples found in the LLM can be different in structure and appearance from the ones found in conventional data sets [7]. An algorithm that has been trained on a set of previously gathered URLs is not always going to classify automatically produced LLM-based URLs correctly [8]. Likewise, training and selection on familiar perturbation types do not guarantee performance on evasion types not used during training and model selection. These conditions should be investigated individually to determine if differences in performance exist between standard test data and the ability to generalise to independent test data and to new test changes in URL [9]. The study is performed on the Robustness-Aware Character Attention Transformer Encoder (RACA-Former) which is used to classify URLs in the binary domain of phishing and legitimate URLs. Compares against a list of standard test URLs, a list of controlled structural perturbations, a list of evasion families, which have not been used to train the model, and a list of URLs generated by an LLM. The design enables a systematic evaluation of classification accuracy for clean, modified and independent URL samples.

1.1 Problem Statement

The task of detecting phishing URL involves the ability to differentiate between phishing and legitimate URLs in different input scenarios [10]. The structure and characters of a URL evolve in a manner that can cause phishing URLs to be misclassified by detection models. Furthermore, the performance on traditional data sets is not sufficient to guarantee the precision of a model when applied to URLs generated by a LLM or new evasion methods [11]. These limitations result in the requirement to evaluate and track detection performance for clean URLs, structurally altered samples, independently created URLs and previously undiscovered evasion patterns. So, the objective of the study is to check the classification accuracy of the RACA-Former in the above evaluation scenario, and to investigate its generalisation performance on the data of the general test set.

1.3 Research Motivation

The goal of this study is motivated by the need to assess phishing URL detection under a different set of classification conditions. A familiar URL might be correctly recognized by the detector, but might not be recognized if the URL is intentionally altered or the URL being detected is not a training example. This concern is more pertinent in the context of subdomain injection, URL padding, path manipulation, homoglyph substitution, and when URLs are created independently with LLMs. Analysis of these conditions individually can give a better understanding of the reliability of their detection and generalisation. Therefore, this study investigates RACA-Former, which combines character-level URL modelling with robustness-aware attention, to examine its classification behaviour on clean URLs, controlled structural perturbations, independent LLM-generated samples, and evasion families excluded from model development. The goal is to develop a valid score for the model under these conditions, not just based on traditional test scores.

1.4 Key Contributions

The proposed model RACA-Former for classification of phishing URLs is investigated in this study, specifically with regards to the effect of changes made to the architecture and generation of independent URLs. The important aspects of the contribution are summarized as follows:

- Introduces RACA-Former, combining character-level URL representation, Transformer encoding, and robustness-aware attention for binary phishing classification.
- Defines systematic procedures for duplicate removal, malformed URL handling, label verification, and cross-dataset overlap detection.
- Evaluates robustness against subdomain injection, URL padding, path manipulation, and homoglyph substitution using validated perturbations.
- Reserves ASA_Phishing_URL_Dataset for independent external testing of generalisation to LLM-generated phishing and legitimate URL samples.
- Assesses generalisation on previously withheld evasion families alongside clean and perturbed URL test samples externally.

1.5 Organization of the Study

The rest of the study is structured in five parts. In Section 2, the existing research regarding phishing URL detection, LLM phishing URL detection and adversarial robustness is reviewed, including methods pertinent to the present study. The proposed RACA-Former framework, including the dataset collection, URL preprocessing and label validation, dataset partitioning, structural perturbation generation, character-level URL representation, model architecture, training and evaluation procedures, are explained in Section 3. The experimental results are then presented and analysed in Section 4, covering classification accuracy on standard test URLs, ability to resist changes in the structure of URLs, generalisation to URLs generated independently by LLM, and performance when faced with new evasion techniques. Section 5 concludes the study by summarising the study's findings and discussing directions for future research.

2. Related Works

Trad and Chehab [12] has explored if a general-purpose LLM could be used instead of a task-specific model for phishing URL detection. Prompt-Engineering techniques are being compared with GPT-3.5-turbo and Claude 2 models, and fine-tuned GPT-2, Bloom, Baby LLaMA and DistilGPT-2 models. The experiment was conducted using 1000 URL samples as test set. The best F1-score of the prompt-based approach was 92.74%, while the F1-score of the fine-tuning approach was 97.29% with highest value of AUC. The results indicate the impact of task-specific adaptation. However, generalization inference in practical application is hampered by small sample size and requires further adversarial robustness testing.

Nasution et al. [13] compared 21 LLMs trained using open-source methodology to check their efficiency in detecting phishing websites with the help of prompts. We explored the concept of zero-shot, few-shot and role-playing prompts and checked LLMs belonging to Llama, Gemma, Qwen, Phi, DeepSeek and Mistral classes. The dataset consisting of 1,000 URLs included 500 phishing and 500 genuine ones. The few-shot prompting was the top performing F1 score in the table, with the Llama 3.3 70B model achieving a score of 91.24%, while the chain-of-thought prompting reduced the accuracy and increased the inference time. However, the generalizability of the conclusions is limited because of the small sample size and the static sample attacks.

Kibriya et al. [14] developed an algorithmic approach for automatically detecting malicious URLs by employing the power of pre-trained language models to create URL embeddings for the purpose of reducing feature engineering manually. The embeddings may be categorized as defacement, malware, benign, and phishing. This is done by a customized DL classifier consisting of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) layers which learns sequential dependency among embeddings. The accuracy of the BERT-based configuration was reported as 97.5%, with around 0.5 million parameters of the classifier and an inference time of 0.119 milliseconds. However, the embedded generation requires GPU resources, and the combined datasets don't capture all the existing malicious URLs either, which can impact the generalization of the results for new malicious URLs.

Chaudhuri and B [15] presented a new Adversarial-Resilient Lightweight Random Forest (AR-LRF) which can resist the attack of the phishing URL detection and can identify realistic attack. The method uses a combination of features of the URLs – including lexical, structural and metadata features – alongside the simulated adversarial transformations during training. The data set consisted of 650,000 URLs, with an imbalanced ratio of about 91% (benign) to 9% (phishing) URLs being used for evaluation. The model was found to be as accurate as 99.78% and reached 0.9999 ROC-AUC in clean conditions and had less performance degradation for considered adversarial perturbations. However, such clean-data statistics are not enough to ensure robustness to all attacks, and temporal drift and new evasion strategies have not yet been covered.

Yıldırım [16] investigated whether the “feature-space attacks” are properly modeling the realistic attacks on phishing URL classifiers. The study examined both feature manipulation and problem-space transformations to real URL and examined robust features like lexical validity, homoglyph disguises, brand impersonation, sub domain depth, character entropy, and transport-related signals. The experiments included comparisons with a different set of real-URLs in the experiments, and a benchmark feature dataset. The proposed feature representation achieved clean-data F1-score of 0.985 compared to 0.994, and showed to be robust against the assessed attacks. However, there were some features that were suggested which an adaptive attacker could exploit and the study highlighted the lack of functionality for live URL verification and host related information.

Prasad et al. [17] presented StealthPhisher, a defensive framework that utilizes complementary neural network architectures and generative AI for the improvement of phishing recognition. This uses a dataset with 336749 rows, each of which is accompanied by the attributes associated with the URL of the row (such as Shannon entropy, Kolmogorov Complexity, and HTML associated attributes). Class-Specific Principal Component Analysis (CSPCA) is used to reduce dimension and a wide and deep neural network and feed-forward neural network learn complementary feature patterns. The accuracy reached in the detection was higher accuracy, and it was robust against Fast Gradient Sign Method (FGSM) attack with 98.70% accuracy. Generative AI can also be used to generate easy detection reports. The provided description does not, however, provide independent cross-dataset verification nor robustness against a wide range of adaptive attacks.

Han et al. [18] explored the capability of LLMs for the generation of elaborate phishing emails and designed a framework to address the threats. It creates deceptive emails with the help of Chain of Thought reasoning, MetaPrompt techniques and domain-specific information - their Phish-Master approach. For the evaluated campus-network environment, high accuracy (higher than 98% accuracy) of evasion was reported, using 100 malicious emails in experiments. A separate integrated machine learning framework which was trained using a phishing email dataset from Kaggle was reported with high accuracy in detecting LLM-generated phishing emails in the test set. The conclusions drawn are limited by the few generators tested, the testing environment, and the fact that the conclusion is compared to the models tested and not to other models and to other phishing attacks.

Brissett and Wall [19] explored the use of supervised learning, unsupervised clustering and watermark based signals for detection of phishing emails through AI. The suggested technique uses a combination of logistic regression models, Term Frequency-Inverse Document Frequency (TF-IDF), k-means clustering and watermarking techniques to detect linguistic indicators and content indicators that can be traced. The study also considers methods that are used to deceive the user in order to make him think that he is receiving a phishing

message. Under controlled conditions the reported experiments resulted in a perfect classification when the combined signals are available. However, the descriptions given do not provide enough quantitative evidence to quantify the size of the data sets used or to provide enough quantitative evidence to establish performance on independent data sets. Finally, there might be a scenario in which someone paraphrases or generates text without watermarking, which could reduce the effectiveness of the system.

Heiding et al. [20] compared the vulnerability to being targeted by phishing emails by automatically constructed phishing emails, manually constructed phishing emails, and hybrid phishing emails (auto/manual). The study tested 112 participants and evaluated phishing emails created with GPT-4 and V-Triad framework and generic control emails. Click-throughs differed in the various experimental conditions, reflecting the effect of message design and personalization. The researchers proceeded to assess the detection accuracy levels of GPT, Claude, PaLM, and LLaMA in detecting phishing and their respective accuracies compared to that of the human population for the purpose of detection. It was found that the performances of the LLMs on detection of the affected regions were satisfactory. Nonetheless, the limitations to this study lie within the scope of the participants used for conducting the experiment.

Ghalechyan et al. [21] neural networks were determined through deterministic and probabilistic approaches using publicly available data and production data gathered using EasyDMARC. Data sources were not only the PhishTank and OpenPhish lists, but also the data gathered from private operational sources, and beyond one static benchmark evaluation. The accuracy of the daily validation was reportedly at 97% on average. The study also revealed that probability predictions can provide helpful information on uncertainty and can improve selected precision and F1 score metrics over the same predicted by a deterministic method. The study mentioned, however, that there may be some labels which are not properly labeled and hence cannot be fully verified. Furthermore, probabilistic inference was far more complex to compute and thus was not as useful for resource-limited real-time applications.

Table 1. Comparative Analysis of Existing Phishing Detection Studies

Author(s)	Methodology	Advantages	Disadvantages
Trad and Chehab [12]	Prompt engineering and fine-tuned LLMs	Demonstrates the benefits of task-specific fine-tuning	Limited test size and insufficient adversarial evaluation
Nasution et al. [13]	Benchmarking 21 LLMs with four prompting strategies	Broad comparison of open-source models and prompting strategies	Limited sample diversity and higher inference costs for some models
Kibriya et al. [14]	BERT embeddings with LSTM and GRU	Automated feature representation and lightweight classification head	GPU-dependent embedding generation and dataset limitations
Chaudhuri and B [15]	Adversarially trained Random Forest with lexical, structural, and metadata features	High clean-data accuracy and improved evaluated adversarial resilience	Emerging attacks and temporal drift remain concerns
Yıldırım [16]	Problem-space evaluation and robust feature engineering	More realistic adversarial assessment and improved clean-data F1-score	Adaptive manipulation remains possible; live functionality was not verified
Prasad et al. [17]	CSPCA, Wide and Deep neural network, feed-forward network, and GenAI	Combines complementary learning pathways and interpretable reporting	Independent cross-dataset validation is not established in the supplied description
Han et al. [18]	LLM-based phishing email generation and integrated ML detection	Evaluates LLM-generated threats and specialized email detection	Limited generation experiment and deployment-specific evaluation
Brissett and Wall [19]	Logistic Regression, TF-IDF, K-Means, and watermarking	Combines linguistic signals, clustering, and content provenance	Watermark availability and paraphrasing may affect detection
Heiding et al. [20]	Human experiments and four-LLM phishing-intent detection	Compares human susceptibility with LLM detection	Limited participant sample and experimental setting
Ghalechyan et al. [21]	Deterministic and probabilistic neural networks	Production-oriented evaluation and prediction uncertainty estimation	Potential label errors and higher computational requirements

Table 1. show the effects of language models, neural architectures, adversarial training, robust feature engineering, and uncertainty estimation on phishing detection. Nonetheless, there are still challenges with diversity of datasets, computational efficiency, and adaptability to attacks, and generalization to real-world

scenarios [26]. These gaps inspire the development of a detection framework that can achieve a trade-off between accuracy in classification, robustness, and ease of deployment.

Existing DL-based phishing URL detection methods require further evaluation to determine how reliably the study classify URLs when attackers deliberately modify their structures through subdomain injection, URL padding, path manipulation, and homoglyph substitution. The strength of the classification on the conventional datasets does not imply resistance to the evasion strategies used here. Furthermore, the score of the models trained on previously collected data might not be transferable to phishing URLs created through LLMs or to evasion attacks that were not included in the development of the models. In this study, these limitations are explored using the RACA-Former language model, classifying URLs with controlled structural perturbations, generated by independently trained LLM-based URLs, and against evasion families not used during training and model selection.

3. Proposed RACA-Former Framework for Robust Phishing URL Detection

The proposed Robustness-Aware Character Attention Transformer Encoder (RACA-Former) aims at detecting phishing URLs based on learned patterns on the character level and their robustness against any structural perturbations or evasion from LLMs. The complete framework is illustrated in Fig. 1. Two different URL data sets are collected for model training and external generalisation testing purposes: labeled conventional URLs and independent URL set from LLM models. The gathered data are preprocessed and validated for labels such as: removing the same records, processing invalid URLs, verifying class-labels, saving informative characters and checking for overlaps across datasets. The conventional dataset is subsequently split into training, validation, and clean testing sets, with the external set of LLM-generated data kept for external assessment. To study robustness against changes in URL structure, URL structure is altered by subdomain injection, URL padding, path manipulation and homoglyph substitution, and label validity is tested for the altered samples. These URLs are then translated into a string of characters and fed into the RACA-Former architecture. It uses a Transformer encoder to understand the context of all characters in the URL, and the proposed robustness-aware attention mechanism focuses on the interactions between characters that are still meaningful when the structure changes. The learned representations are combined and passed to a classification head to classify phishing URLs as such. After that, the model is evaluated using clean URLs, perturbed data samples, independently created LLM URLs, and previously revoked evasion family URLs to measure its accuracy and generalisation capability.

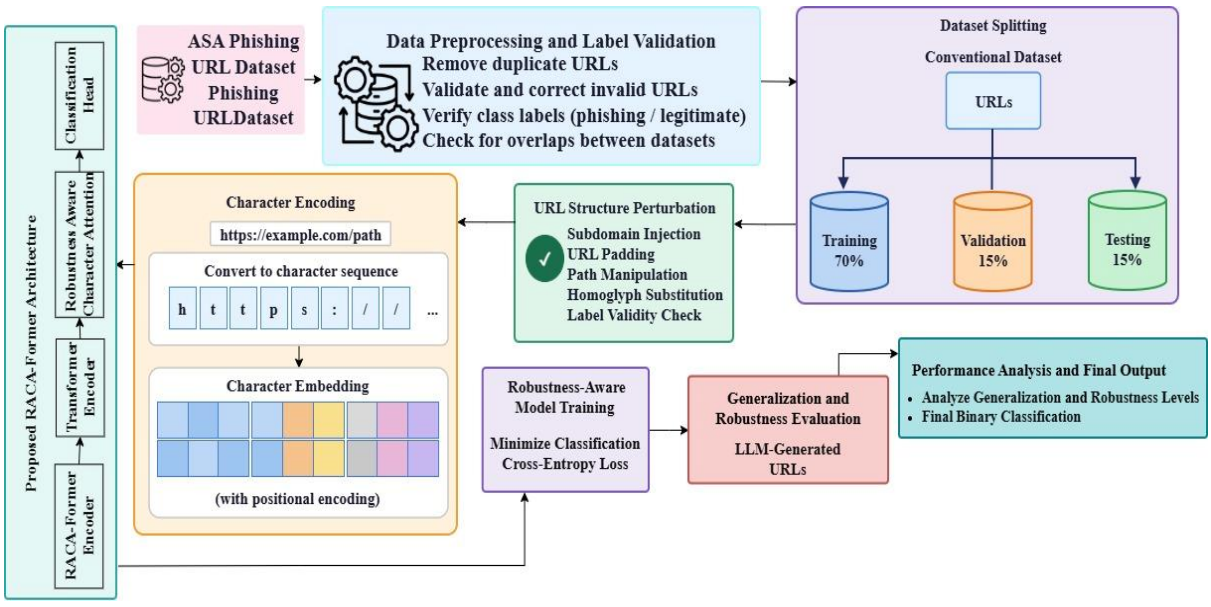

Fig. 1 Proposed RAC-Former Methodology for Robust Phishing URL Detection

3.1 Dataset Collection

The data set used for this study consists of two labeled sets of URLs sourced from Mendeley Data. The first data set includes 450,176 URL instances, out of which 104,438 belong to phishing URLs and 345,738 belong to legitimate URLs, obtained from major sources like PhishTank, Majestic Million Ranking and others URL feeds. In addition, the same data repository also offers a second data set which consists solely of phishing URLs, and is comprised of 54,807 URL instances [22]. The second one is the ASA_Phishing_URL_Dataset which has 5,000 synthetically created URLs comprising 2500 phishing URLs and 2500 legit URLs [23]. These URLs are generated through the use of a LLM-based prompting technique and include typosquat, IP-address based host, suspicious top-level domain, and legitimate URLs. Both data sets include URLs along with the labels of the classes, thus allowing character-level modelling without the need of manually crafted features. The conventional dataset is used for model development and evaluation of clean data, the ASA dataset is reserved for independent

external testing to investigate generalisation to LLM-generated URLs. Records are duplicated and overlaps between the datasets are checked prior to experimentation to minimize the possibility of data leakage.

Table 1. Composition and statistics of the datasets

Dataset	Source	Total URLs	Phishing	Legitimate	Phishing share (%)	Mean length (chars)
Phishing URL dataset: URL dataset.csv	PhishTank, Majestic Million, other sources	450,176	104,438	345,738	23.2	60.2
Phishing URL dataset: Phishing URLs.csv	Phishing-only compilation	54,807	54,807	0	100.0	66.6
ASA_Phishing_URL_Dataset	LLM-generated	5,000	2,500	2,500	50.0	30.8

Table 1 provides summary information about the files in terms of sources, total URLs, URLs by class type, percentage of phishing URLs, and average URL length in characters. The first file is huge and unbalanced; the second one consists exclusively of phishing samples; while the ASA set is small and balanced.

3.2 Data Preprocessing and Label Validation

Data preprocessing and label validation ensure a uniform and stable input representation for the proposed RACA-Former framework. The recorded URLs are analyzed to detect the duplicate URLs, invalid strings, incorrect class assignment, and overlapping samples between datasets. URL characters are carefully preserved since punctuation, domain elements, and character substitutions can contain valuable phishing related data. For the structurally modified URLs, further validation is performed to maintain the proper meaning of their labels. These operations can be helpful to ensure consistency of the data and to limit the possibility of mis-estimating model performance.

3.2.1 Duplicate URL Detection and Removal

Duplicate URL detection is performed to prevent repeated records from disproportionately influencing model learning and evaluation. The comparison representation is used to compare URLs, but the original URL strings are not modified for character-level analyses. A representation of each unique URL is kept in the dataset collected, and any duplicates are detected in the collected dataset. The resulting dataset is described by Eq. (1).

$$\mathcal{D}_{\text{unique}} = \{(u_i, y_i) : u_i \not\equiv u_{j \text{ for } j} < i\} \quad (1)$$

Here, $\mathcal{D}_{\text{unique}}$ represents the data set without any duplication, u_i denotes the i-th URL, y_i is its class label, and $\not\equiv$ indicates equivalence under the selected URL comparison rule. The condition $j < i$ retains the first occurrence of each equivalent URL.

3.2.2 Invalid and Malformed URL Record Handling

The URL records are analysed to detect missing data, empty string, truncation, and mal-formed structure which cannot be used reliably. A validation function is applied before the record entry in character form representation. Records which do not satisfy the set of inputs defined are not entered or corrected until reliable data is available. The selection of valid records is given by Eq. (2).

$$v_i = \begin{cases} 1, & if\ u_i\ satisfies\ the\ validation\ rules \\ 0, & otherwise \end{cases} \quad (2)$$

Where,v_i is the validity indicator for URL u_i. A value of 1 denotes a record that satisfies the predefined validation criteria, whereas 0 indicates an invalid record. The validation rules should be specified consistently and should not reject unusual URLs merely because their structures appear suspicious.

3.2.3 Phishing and Legitimate URL Label Verification

Every retained URL is analysed to confirm whether the class label of the URL is consistent with the binary classification task. The label correctly identifies the phishing URLs, and also shows that the URL is uncertain or conflicting, hence requires verification before classification. Mathematically, the verified data set can be expressed using Eq. (3).

$$\mathcal{D}_{\text{verified}} = \{(u_i, y_i) : y_i \in \{0,1\}\} \quad (3)$$

Here, $\mathcal{D}_{\text{verified}}$ denotes the dataset containing accepted URL-label pairs, u_i represents a URL, and y_i denotes its verified binary label. The values 0 and 1 represent legitimate and phishing URLs, respectively, according to the adopted label convention. A binary value alone does not establish label correctness; verification requires a reliable labelling procedure.

3.2.4 Cross-Dataset Duplicate and Overlap Detection

Cross-dataset overlap check is conducted to detect URLs which appear in multiple datasets. This step is necessary when a conventional dataset is used for building the model and an additional dataset generated by the

LLM is reserved for external validation. Consistently processed URL representations allow to detect overlaps before splitting the dataset. Intersection of two datasets can be calculated according to Eq. (4).

$$\mathcal{O}_{\mathcal{A},\mathcal{B}} = \mathcal{U}_{\mathcal{A}} \cap \mathcal{U}_{\mathcal{B}} \quad (4)$$

Here, $\mathcal{O}_{\mathcal{A},\mathcal{B}}$ represents the set of overlapping URLs, while $\mathcal{U}_{\mathcal{A}}$ and $\mathcal{U}_{\mathcal{B}}$ denote the URL sets in datasets $\mathcal{A}$ and $\mathcal{B}$, respectively. The intersection identifies records shared by both collections under the selected comparison rule. Any detected overlap should be handled according to the experimental protocol to preserve the independence of external evaluation.

3.2.6 Label-Preserving URL Validation for Structural Perturbations

Structure-based perturbations are applied to analyse the impact of the changes made to URLs on phishing identification. But alteration of a URL does not ensure that the new URL will belong to the same class. It can point to some other domain or object. Consequently, every newly created instance must be labelled properly. The acceptance criterion for label-preserving perturbation is given by Eq. (5).

$$P_{valid} = \{(T_k(u_i), y_i) : y(T_k(u_i)) = y_i\} \quad (5)$$

Here, P_{valid} denotes the set of accepted perturbed URL samples, T_k represents the k-th structural perturbation operation, u_i is the original URL, and y_i is its original label. The term $y(T_k(u_i))$ denotes the verified label of the modified URL. A perturbed record is retained with its original label only when the label remains valid; otherwise, it must be relabelled using reliable evidence or excluded from the label-preserving set.

3.3 Dataset Partitioning and Generalisation Protocol

Dataset partitioning is meant to facilitate the process of model development and evaluation of generalization capabilities for the developed RACA-Former architecture. The dataset is divided into training, validation, and clean test sets in order to differentiate between model learning and parameter optimization and testing stages. Duplicate instances and any overlaps are checked before dataset partitioning in order to reduce data leakage. The ASA_Phishing_URL_Dataset is used exclusively as an external dataset for evaluation purposes and is not used for training and optimization of the model. Such approach allows to test the model on both conventional URLs and independently generated LLM-based URLs. The conventional dataset is partitioned into three disjoint subsets, as shown in Eq. (6).

$$D_c = D_{train} \cup D_{val} \cup D_{test} \quad (6)$$

Where D_c denotes the conventional URL dataset, D_{train} represents the training subset, D_{val} denotes the validation subset, and D_{test} represents the clean testing subset. The union of these subsets constitutes the conventional dataset, with each URL assigned to only one subset to prevent direct record overlap. The independent external dataset is kept separate from the conventional dataset partitions, as defined in Eq. (7).

$$D_{external} \cap (D_{train} \cup D_{val} \cup D_{test}) = \emptyset \quad (7)$$

Here, $D_{external}$ denotes the ASA_Phishing_URL_Dataset, and $\emptyset$ indicates an empty intersection. This condition specifies that no identical URL records should be shared between the external dataset and the conventional training, validation, or testing subsets. Maintaining this separation supports an independent assessment of generalisation to LLM-generated URLs.

3.4 Controlled Structural URL Perturbation Generation

The URL structures are altered in a controlled way, and the proposed RACA-Former framework is evaluated for determining whether it can learn to identify the phishing-related patterns given the alteration of URL structures. The alterations of URLs being considered include subdomain injection, URL padding, path alteration, and homoglyph substitution. The selected URL is perturbed in accordance with certain rules with the original URL preserved for comparison. The validity of the labels of the generated data samples is verified. The perturbation process is illustrated in Eq. (8).

$$u_i^{(k)} = T_k(u_i) \quad (8)$$

Where, $u_i^{(k)}$ denotes the URL generated after applying the k-th perturbation to the original URL u_i, and T_k represents the corresponding transformation operation. The transformation may modify a URL's subdomain, length, path, or character appearance, depending on the selected perturbation type. Each URL that has been modified is evaluated to see if the original class label is still applicable, to ensure the reliability of the resulting dataset. Only samples with valid labels are retained for label-preserving robustness experiments, as formulated in Eq. (9).

$$P_{valid} = \left\{(u_i^{(k)}, y_i) | y(u_i^{(k)}) = y_i\right\} \quad (9)$$

Here, P_{valid} denotes the set of accepted perturbed URL samples, $u_i^{(k)}$ represents the modified URL, y_i is the original class label, and $y(u_i^{(k)})$ denotes the verified label of the modified URL. A sample is included only when the modified URL retains the original label. If the transformation changes the URL's actual classification, the sample must be relabelled using reliable evidence or excluded from the label-preserving set.

3.5 Character-Level URL Representation

Character-based representation of URLs refers to the representation where each URL is represented as a sequence of individual characters. With character-level URL representation, it becomes possible for the proposed RACA-Former framework to learn structural patterns from the actual URL text itself. Character-level URL representation preserves all the discriminative characters in the URL, including domain separators, path

symbols, digits, and others, without using any handcrafted features. All the characters are represented as integers based on a predefined vocabulary, and the character-level sequence length policy is applied, resulting in Eq. (10).

$$x_i = \left(c_{i,1}, c_{i,2}, \ldots, c_{i,L_i}\right) \quad (10)$$

Where x_i denotes the character sequence of URL u_i, $c_{i,t}$ represents the character at position t, and L_i indicates the total number of characters in the URL. The ordered sequence preserves character-level structural information for subsequent representation learning. To enable neural network processing, each character is mapped to its corresponding integer index in the character vocabulary. The conversion creates a numerical sequence but preserves the order of the characters as expressed in Eq. (11).

$$z_i = \left(V(c_{i,1}), V(c_{i,2}), \ldots, V(c_{i,L_i})\right) \quad (11)$$

Here, z_i denotes the numerical representation of URL u_i, and $V\left(c_{i,t}\right)$ represents the vocabulary index assigned to character $c_{i,t}$. The function $V(\cdot)$ maps each character to an integer identifier, enabling the character sequence to be processed by subsequent embedding layers. Any padding or truncation applied to obtain a fixed sequence length should follow the predefined preprocessing procedure.

3.6 Proposed RACA-Former Architecture

The proposed Robustness-Aware Character Attention Transformer Encoder (RACA-Former) combines the character-level contextual learning with the robustness-aware attention mechanism to classify phishing and legitimate URLs. The architecture encodes the numerical character sequence with a Transformer to learn the relationships between the URL components and character position. The proposed attention mechanism filters out potentially informative patterns under structural variations to obtain the resulting contextual representations. The simplified features are then fed into a classification head which produces the final prediction. The entire calculation process of RACA-Former is summarized as follows in Eq. (12).

$$\hat{y}_i = Softmax(f_{cls}(f_{RA}(f_{Tr}(z_i)))) \quad (12)$$

Where, $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ denotes the predicted class-probability vector for URL u_i, z_i represents its numerical character sequence, $f_{\mathrm{Tr}}(\cdot)$ denotes the Transformer encoder, $f_{\mathrm{RA}}(\cdot)$ represents the robustness-aware attention mechanism, and $f_{\mathrm{cls}}(\cdot)$ denotes the classification head. The Softmax function converts the final classification scores into probabilities for the phishing and legitimate classes.

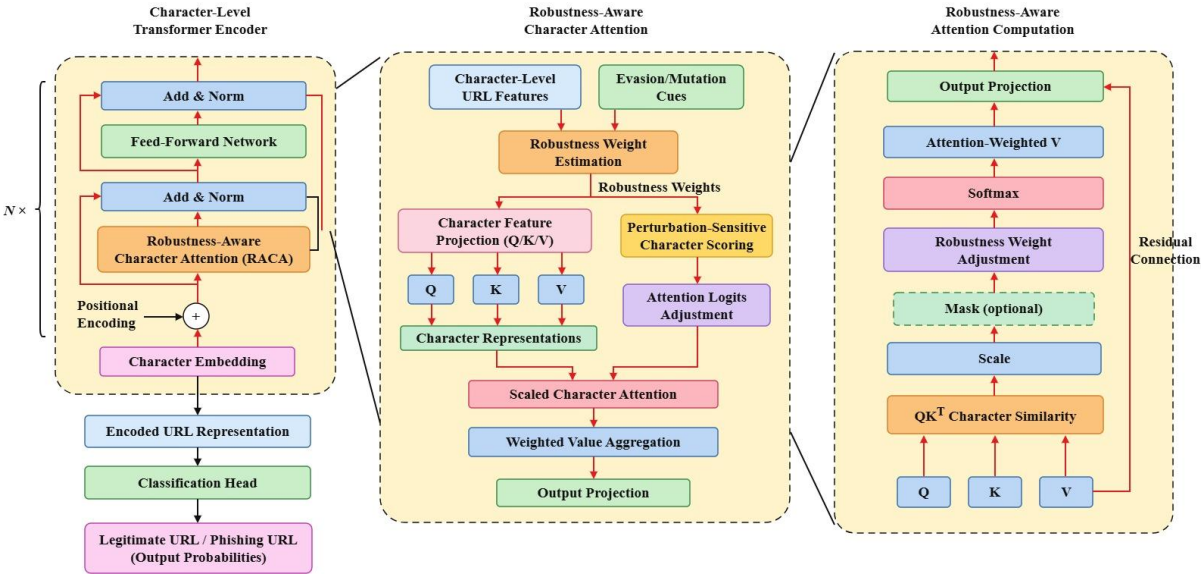

Fig. 2. Proposed RAC-Former Architecture for Phishing Detection

The proposed RAC-Former is designed to recognize the phishing URLs by character-level representation learning and robustness-aware attention. Initially, URL characters are converted into embeddings combined with positional information. The encoder features Robustness-Aware Character Attention (RACA) that focuses on informative character patterns and addresses URL perturbations such as subdomain injection, URL padding, path manipulation and homoglyph substitution. The representations of query, key, and value are projected into character features and the attention computation is modulated by robustness weights. The representations of query, key, and value are mapped into character features, and the robustness weights modulate the attention computation. Weighted value aggregation generates enhanced character representations. These features are further enhanced by residual connections, normalization and feed-forward layers. Finally, the classification head predicts whether each URL is legitimate or phishing.

3.7 Robustness-Aware Model Training

The RACA-Former model is trained with labelled URL sequences to leverage the character-level patterns of distinguishing phishing URLs from legitimate ones. The training data can contain clean URLs and validated and appropriately labelled noisy URLs to present the model with structural variations. Minimizing the classification loss on training samples to update the model parameters, and using the validation subset for monitoring learning and selecting configurations of the model. The classification objective is described in Eq. (13).

$$\mathscr{L}_{cls} = -\frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}\sum_{\mathrm{c}=1}^{\mathrm{C}} \mathrm{y}_{\mathrm{i,c}} log\left(\hat{p}_{i,c}\right) \quad (13)$$

Where, $\mathscr{L}_{cls}$ denotes the classification loss, N represents the number of training samples, C is the number of classes, $\mathrm{y}_{\mathrm{i,c}}$ is the ground-truth indicator for class c of sample i, and $\hat{p}_{i,c}$ denotes the predicted probability assigned to that class. Minimising this cross-entropy loss encourages the model to assign higher probabilities to the correct URL labels. Clean and validated perturbed samples can be merged in a single training set to include structural variations during training. The combined objective is represented in Eq. (14).

$$D^*_{train} = D_{clean} \cup D_{perturbed} \quad (14)$$

Here, D^*_{train} denotes the combined training set, D_{clean} represents the clean training samples, and $D_{perturbed}$ contains structurally modified URLs whose labels have been validated. Their union provides training examples with different URL structures, potentially helping the model learn patterns that are less dependent on a single URL form.

3.8 Generalisation Testing with LLM-Generated and Unseen Evasion URLs

The generalisation test is performed to evaluate the ability of the RACA-Former model to perform phishing URL detection when faced with URLs from sources and structures different from what were seen during training. This is done using the independent ASA_Phishing_URL_Dataset for evaluation of the performance on LLM-generated URLs and the perturbation families not included in training data for evaluation of performance under novel evasion attacks. The model performs classification on every test example and the results are compared against the ground truth. The classification accuracy on the evaluation set is defined by Eq. (15).

$$Accuracy = \frac{N_{correct}}{N_{total}} \times 100 \quad (15)$$

Where, $N_{correct}$ denotes the number of URLs classified correctly by RACA-Former, and N_{total} represents the total number of evaluated URLs. Accuracy measures the proportion of correct predictions in the selected evaluation set and can be calculated separately for LLM-generated URLs and samples from held-out evasion families. To examine the effect of structural evasion on detection, the model is tested against the perturbed URLs and compared against the corresponding clean URLs. The relative change in accuracy is expressed in Eq. (16).

$$\triangle A = A_{clean} - A_{perturbed} \quad (16)$$

Here, $\triangle A$ denotes the change in classification accuracy, A_{clean} represents the accuracy obtained on clean URLs, and $A_{perturbed}$ denotes the accuracy obtained on the corresponding perturbed evaluation set. A positive value indicates that accuracy decreased under perturbation, while zero indicates no observed change. A negative value indicates higher accuracy on the perturbed set. Both accuracies should be calculated using the same metric definition and comparable evaluation samples.

3.9 Network URL Classification

In the last step of classification, the RACA-Former model that was already trained is used to classify the input URL as phishing or legitimate. Each URL is first processed and encoded to numbers at the character level, just like during the training of the model, and passed through the Transformer encoder, robustness-aware attention mechanism, and classification head. The output of the model is a set of probabilities for each class, and the class with the maximum probability is selected. The decision process can be summarized by Eq. (17).

$$\hat{y}_i = \arg\max_{c\in\{0,1\}} \hat{p}_{i,c} \quad (17)$$

Where, $\hat{y}_i$ denotes the predicted class for URL u_i, c represents a candidate class, and $\hat{p}_{i,c}$ is the probability assigned to class c by the trained model. The class labels are defined as 0 for legitimate URLs and 1 for phishing URLs. The $arg\ max$ operation selects the class with the highest predicted probability. For a binary classification task, the predicted class can also be expressed using a probability threshold, as shown in Eq. (18).

$$\hat{y}_i = \begin{cases} 1, \hat{p}_{i,1} \geq \tau \\ 0, \hat{p}_{i,1} < \tau \end{cases} \quad (18)$$

Here, $\hat{p}_{i,1}$ denotes the predicted probability that URL u_i belongs to the phishing class, and τ represents the decision threshold. If the probability of phishing exceeds the value of the threshold, then the URL is identified as a phishing one; otherwise, it will be recognized as a legitimate one. Selection of the threshold value needs to be done based on the chosen decision rule and if required based on the validation data.

Algorithm 1. RACA-Former for Phishing URL Detection and Robustness Evaluation

Input: Labelled URL dataset (D), external ASA dataset (D_ASA), perturbation families (P), maximum sequence length (L=128), classification threshold ($\tau = 0.5$).
Output: Predicted URL classes and evaluation metrics.
Begin

Load the labelled URL dataset (D).
Remove invalid records, conflicting-label records, and duplicate URLs.
IF a URL record is invalid THEN discard the record.
END IF
IF a URL has conflicting labels that cannot be reliably resolved THEN exclude the record.
END IF
Divide the cleaned dataset into stratified training, validation, and clean test subsets.
FOR EACH perturbation family ($p \in P$) DO
Generate modified URLs using perturbation family (p).
IF the modified URL is valid and retains its original class label THEN accept the sample.
ELSE exclude the sample or relabel it only when reliable evidence supports the new label.
END IF
END FOR
Convert each URL into a character sequence and map its characters to vocabulary indices.
Apply the predefined padding and truncation policy to obtain sequences of length (L).
Feed the encoded sequences into the Transformer encoder to learn contextual character representations.
Apply the robustness-aware attention mechanism to aggregate the character representations.
Pass the aggregated representation through the classification head and Softmax layer to obtain class probabilities.
Train RACA-Former using labelled training samples and cross-entropy loss.
Monitor validation performance and retain the model selected using the validation results.
FOR EACH evaluation dataset (D_e), including the clean test set and accepted perturbation sets, DO
Predict class probabilities for every URL in (D_e).
IF the phishing-class probability is greater than or equal to (τ)THEN classify the URL as phishing.
ELSE classify the URL as legitimate.
END IF
Calculate the required classification metrics for (D_e).
END FOR
Evaluate the trained model on the external ASA dataset without using it for model selection.
FOR EACH held-out perturbation family DO
Train and validate the model without that perturbation family.
Evaluate the resulting model on the withheld family.
Record the classification metrics.
END FOR
Compare results across clean, perturbed, external, and held-out evaluation settings.
End

Algorithm 1 describes the complete workflow of the proposed RACA-Former for phishing URL detection and robustness evaluation. It starts with URL preprocessing, data partitioning, and structurally perturbed samples validation, then character-level encoding and representation learning using the Transformer model. A robust-aware attention network is included to aggregate the character features before the phishing and legitimate URL classifier network. The trained network is evaluated on clean, perturbed, external, and held out evasion samples to ascertain the classification performance and limitations of the generalisation process.

The novelty of the proposed RACA-Former is that it combines the character-level Transformer representation with a robustness-aware attention mechanism for phishing URL classification in the presence of structural variations. The model treats the URL as a text string and learns contextual patterns for the legitimate and malicious URLs instead of relying on manually designed URL characteristics. The attention module is an element that aggregates character representations to be classified and provides a model architecture that should deal with variability induced by URL manipulation. Furthermore, the work extends evaluation beyond the conventional clean test framework by evaluating against 4 perturbation families: subdomain injection, URL padding, path manipulation and homoglyph substitution. Generalization across sources and sensitivity to new evasion strategies are tested through external evaluation on the ASA phishing URL dataset and perturbation experiments on held-out data. An ablation study on masked mean pooling reveals whether the attention mechanism is useful. The findings reveal remarkable performance under clean test conditions as well as weaknesses in the presence of distribution shift and out-of-distribution subdomain injection.

3.10 Statistical Learning Theory and Robustness Analysis

Generalisation and robustness are analysed using statistical learning theory. For a classifier f with 0–1 loss, the true risk $R(f)$ is bounded by the empirical risk on n held-out URLs through Hoeffding's inequality, with probability at least $1-\delta$:

$$R(f) \leq \hat{R}_n(f) + \sqrt{\frac{\ln(1/\delta)}{2n}} \qquad (19)$$

Robustness to the k-th perturbation family is measured by the gap Γ_k, which equals $\Delta A/100$ from Eq. (16), and by the worst-case risk over families:

$$\Gamma_k = R_k(f) - R(f), \quad R_{\text{rob}}(f) = \max_k R_k(f) \qquad (20)$$

Under shift from a source domain S to a target domain T, the target risk is bounded by the source risk, a divergence term d between the domains, and the combined error λ^* of the best single classifier:

$$R_T(f) \leq R_S(f) + \frac{1}{2} d(\mathcal{D}_S, \mathcal{D}_T) + \lambda^* \qquad (21)$$

Training on seen perturbation families constrains only R_S, so it gives no guarantee on unseen families or external data.

4. Result and Discussion

This section analyses the classification performance and generalisation capabilities of the RACA-Former framework for detecting phishing URLs in clean, perturbed, and independently constructed settings. Specifically, the analysis considers the held-out clean-test set, four label-consistent perturbations of the structure of the URLs, the independent ASA dataset that contains URLs created using LLMs, and the set of evasion families that were not seen during training and model selection. If not stated otherwise, all reported results are averaged mean ± SD over three seeds (0, 1, and 2), except for the held-out evasion families which utilise a single seed 0. The results are interpreted to investigate clean-test classification performance, robustness to perturbations of structural URLs, external dataset generalisation and evasion family performance. A total of ten figures are used to illustrate the experimental results where each figure is focused on a different part of the evaluation. As no statistical significance test was conducted, all reported differences are descriptive.

Table 2. Experimental configuration and evaluation parameters

Parameter	Setting
Partition	Stratified record-level 70/15/15, seed 42: train 347,675; validation 74,502; clean test 74,501
Training data used	24,000 clean + 24,000 label-validated perturbed URLs = 48,000 per epoch (compute-limited subset)
Validation data used	6,000 clean + 6,000 perturbed URLs (seen families only) = 12,000
Character vocabulary	95 printable ASCII characters + PAD + UNK = 97 symbols (fixed)
Sequence length	128 characters (3.88% of pool URLs truncated); length buckets 48/64/96/128
Robustness-aware attention	Per-character sigmoid gate + 4 learned attention queries pooled over encoder outputs; head: LayerNorm, dense 64, dense 2; 111,299 parameters
Loss / optimiser	Cross-entropy; Adam (β = 0.9, 0.999), weight decay 0.01, peak learning rate 2×10^{-3}, 100-step warm-up, cosine decay
Batch size / epochs	128 / 3; epoch with highest validation accuracy retained
Decision threshold	0.5
Random seeds	Model seeds 0, 1, 2; split seed 42; perturbation seeds fixed per URL
Perturbations	Subdomain: 1–3 labels from an 18-word list + optional 4-8-character token. Padding: 20–60 random alphanumeric characters in query or path. Path: insert 1–2 segments, change extension, or add dot-segment.
Label validation	Registered domain preserved (phishing homoglyph: altered domain must not equal any legitimate registered domain); invalid, unchanged and inapplicable outputs excluded
Perturbed test set	10,000 stratified clean test URLs; accepted counts 9,898 / 10,000 / 9,749 / 8,895 (subdomain / padding / path / homoglyph)
External test	ASA_Phishing_URL_Dataset, 5,000 URLs (2,500 / 2,500), evaluation only
Unseen-evasion protocol	Leave-one-family-out: three families in training and validation, the fourth only at test; seed 0

Table 2 contains the summary of experimental settings in RACA-Former. This table provides information on the following aspects: dataset split, training procedure, character representation, model design, optimization procedure, decision threshold, adversarial generation, label validity, external assessment, random seeds, and testing of novel evasion families.

4.1. Dataset Characteristics and Preprocessing

The database went through a systematic process of pre-processing for improvement in terms of data quality and modelling purposes. Invalid records, conflicting labels, and duplicate URLs were removed before partitioning the data into training, validation, and testing subsets. The resulting dataset provides the basis for assessing RACA-Former's phishing URL classification performance.

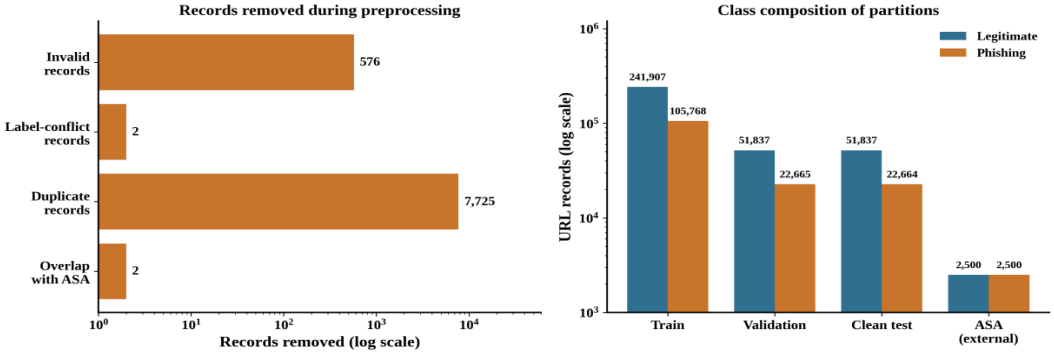

Fig. 3. Records removed during preprocessing and class composition of the resulting dataset partitions.

Fig 3 illustrates the preprocessing results and distribution of class among the training, validation, clean-test, and external ASA data sets. Duplicates were the most removed, followed by invalid entries. The internal data sets have both classes, whereas the external data set has an equal number of phishing and legitimate URLs.

4.2. Overall Phishing URL Detection Performance

The phishing URL detection performance of RACA-Former was evaluated on a held-out clean-test dataset comprising 74,501 URLs, including 22,664 phishing and 51,837 legitimate samples. The evaluation is based on the accuracy of the model to distinguish between the two classes based on a fixed classification threshold of 0.5, where the positive class are the phishing URLs. The results serve as a reference point to evaluate the effectiveness of detections in non-modified input conditions.

Table 3. Clean-test classification performance of RACA-Former on the held-out test partition

Metric	RACA-Former
Accuracy	99.16 ± 0.08
Precision	99.45 ± 0.04
Recall	97.79 ± 0.30
F1-score	98.61 ± 0.13
Specificity	99.76 ± 0.02

Table 3 below shows RACA-Former's clean-test performance for five classification metrics. The results show that phishing and legitimate URLs can be well distinguished, with high precision and specificity values. The lower rate of recall, however, suggests that there are still phishing URLs that haven't been detected and it's important to assess performance under the structural evasion condition.

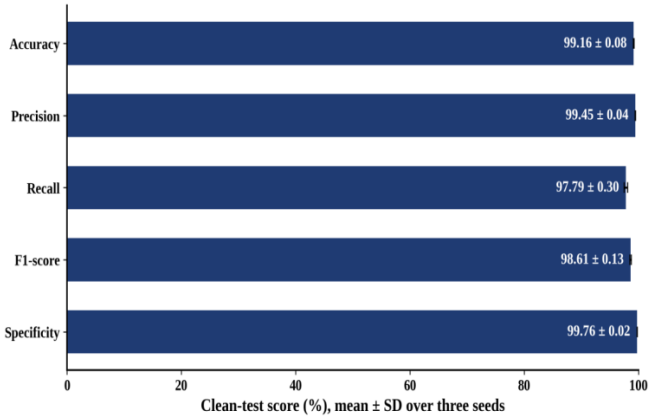

Fig. 4. Clean-test classification performance of RACA-Former.

Fig. 4 shows the clean-test performance of the RACA-Former and its capability to classify the phishing sites as 99.16% accuracy and 99.76% specificity. While these results satisfy the requirement for reliable classification in clean conditions, the low recall of 97.79% shows that there is still a chance of errors, which must be verified with the use of structural evasion techniques.

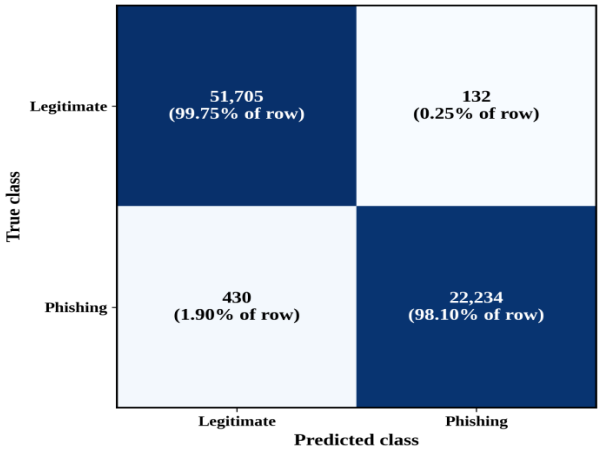

Fig. 5. Clean-test confusion matrix of RACA-Former

Fig. 5 shows the seed-0 confusion matrix for the clean-test set for RACA-Former. The model has correctly classified 22234 phishing URLs and 51705 legitimate URLs with 132 misclassifications of legitimate URLs and 430 failure to detect phishing URLs. The results show good clean-data classification, but more robust testing should be performed to determine resistance to evasion methods.

4.3. Robustness Evaluation Against Structural URL Perturbations

RACA-Former was evaluated against four structural URL perturbations, including subdomain injection, URL padding, path manipulation, and homoglyph substitution. To test the model's performance under conditions of evolving URL structure with no change to the intended label, these experiments explore the robustness of the model under non-clean-test conditions.

Table 4. RACA-Former classification performance on four structural URL perturbation families

Perturbation	URLs	Accuracy	Precision	Recall	F1-score	Specificity
Subdomain injection	9,898	97.56 ± 0.08	98.72 ± 0.40	92.99 ± 0.67	95.77 ± 0.17	99.49 ± 0.16
URL padding	10,000	98.83 ± 0.27	99.39 ± 0.03	96.73 ± 0.90	98.04 ± 0.45	99.74 ± 0.01
Path manipulation	9,749	99.11 ± 0.15	99.42 ± 0.03	97.65 ± 0.51	98.53 ± 0.25	99.75 ± 0.01
Homoglyph substitution	8,895	99.25 ± 0.09	99.80 ± 0.03	97.99 ± 0.28	98.89 ± 0.14	99.90 ± 0.02

Table 4 shows how RACA-Former rectifies the research gap of evaluating phishing detection not only with clean input, but with structural URL modifications as well. Overall, accuracy and recall are very similar across all four perturbation families, though the greatest drop in accuracy and recall is seen for subdomain injection, showing that there is still work to be done to identify altered phishing URLs.

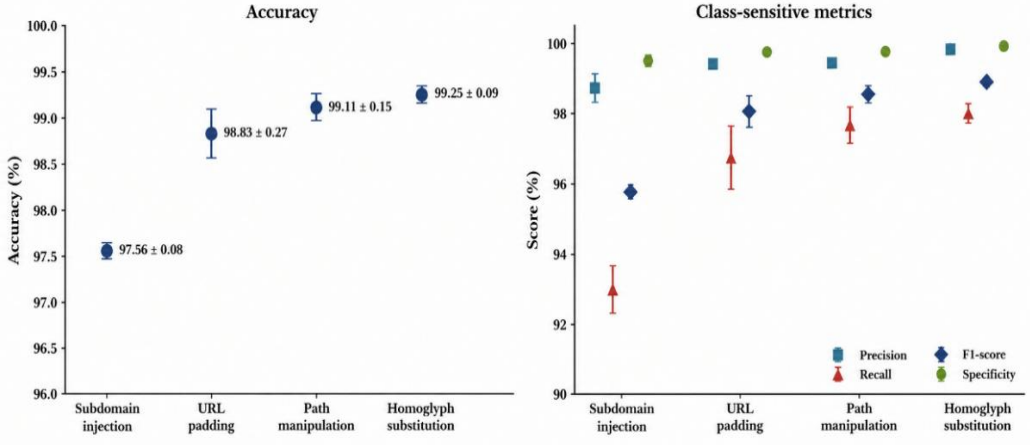

Fig. 6. Performance of RACA-Former across structural URL perturbation families

Fig. 6 shows the accuracy and class-sensitive metrics for the four structural URL perturbations of RACA-Former, filling a missing link in the evaluation of the reliability of detection for non-clean inputs. The most significant drop is seen in case of subdomain injection, which falls by 92.99% in recall, whereas path

manipulation and homoglyph substitution maintain a relatively high performance. These findings demonstrate robustness to known perturbations, but also reveal vulnerability to specific URL modifications.

4.4. Performance Degradation Analysis Under URL Perturbations

The degradation in performance was evaluated by comparing the classification accuracy of RACA-Former on clean URLs with the accuracy after structural perturbations. This analysis quantifies the effect of URL changes on detection reliability, pinpoints the families of perturbations that cause the most loss of performance, and explains the limitations of the model during the change of URL.

Table 5. Clean-to-perturbed accuracy change and accuracy retention

Perturbation	A_clean (%)	A_perturbed (%)	ΔA (pp)	Relative change (%)	Retention (%)
Subdomain injection	99.15 ± 0.11	97.56 ± 0.08	1.59 ± 0.05	1.61 ± 0.05	98.39 ± 0.05
URL padding	99.16 ± 0.11	98.83 ± 0.27	0.33 ± 0.16	0.34 ± 0.16	99.66 ± 0.16
Path manipulation	99.16 ± 0.11	99.11 ± 0.15	0.05 ± 0.03	0.05 ± 0.03	99.95 ± 0.03
Homoglyph substitution	99.18 ± 0.13	99.25 ± 0.09	-0.07 ± 0.03	-0.07 ± 0.03	100.07 ± 0.03

Table 5 presents quantitative results of the effect of structural perturbations in the URL on RACA-Former's accuracy, filling the gap in the literature regarding the quantification of detection stability under modified inputs. Subdomain injection generates the greatest loss of accuracy, at 1.59 percentage points, while path manipulation has the least change. The effect depends on the type of perturbation, and a slight improvement in accuracy is achieved by homoglyph substitution.

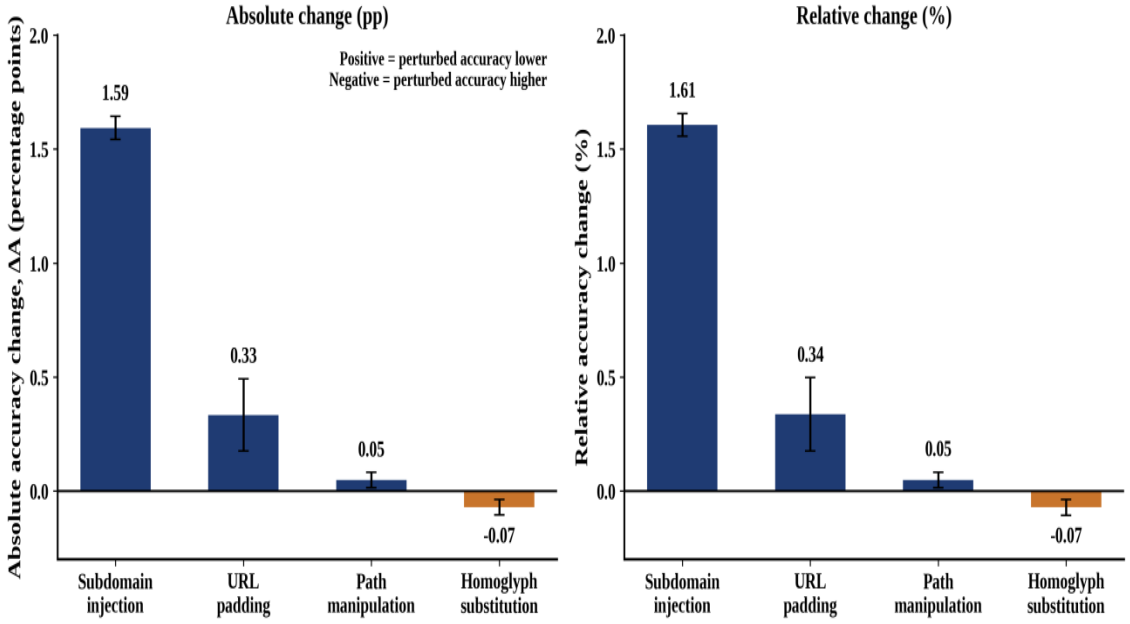

Fig. 7. Absolute and relative accuracy change of RACA-Former under perturbation

Fig. 7 represents the accuracy changes based on the structural URL perturbations which is the research gap for measuring the detection stability under modified inputs. The two highest losses are for subdomain injection at 1.59 percentage points followed by URL padding at 0.33 points. The effects of path manipulation are modest; robustness for homoglyph substitution is marginally better, indicating that perturbation-type plays a role in robustness.

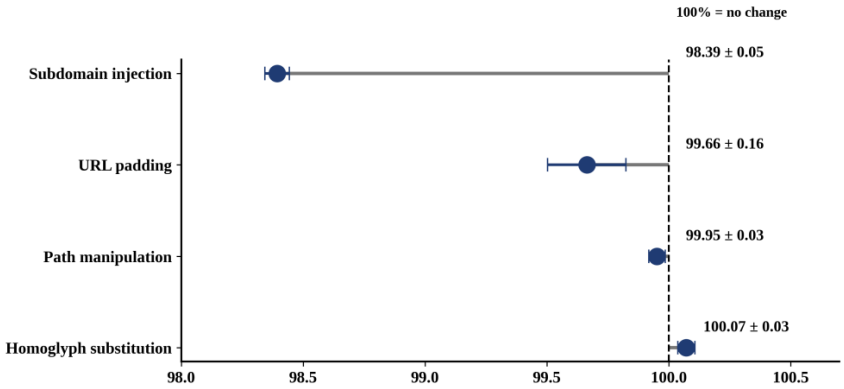

Fig. 8. Accuracy retention of RACA-Former across perturbation families

Fig. 8 addresses the research gap in evaluating phishing detection robustness by quantifying how much accuracy RACA-Former retains when URL structures are modified. Retention ranges from 98.39% for subdomain injection to high accuracy for homoglyph substitution, identifying subdomain injection as the principal weakness among the tested perturbations. These findings quantify stability under known modifications but do not establish robustness against unseen evasion techniques.

4.5. Generalisation Performance on Independently Generated LLM-Based URLs

The external evaluation examines whether RACA-Former maintains phishing detection performance on the ASA dataset of 5,000 LLM-related URLs, comprising 2,500 phishing and 2,500 legitimate samples. This evaluation is a cross-dataset generalisation evaluation and will highlight whether performance is seen across traditional data transfers or across a collection of URLs sourced from a different data source.

Table 6. RACA-Former classification performance on the independent ASA external dataset

Metric	RACA-Former
Accuracy	62.33 ± 0.18
Precision	57.03 ± 0.12
Recall	100.00 ± 0.00
F1-score	72.64 ± 0.10
Specificity	24.67 ± 0.36

Table 6 addresses the gap in external generalisation by evaluating RACA-Former on the independently sourced ASA dataset rather than relying solely on conventional test data. The model is highly phishing recall with a low specificity of 24.67% that shows that it may classify a lot of URLs as phishing without there being any phishing in them. Therefore, it is difficult to make a good discrimination between various URL sources.

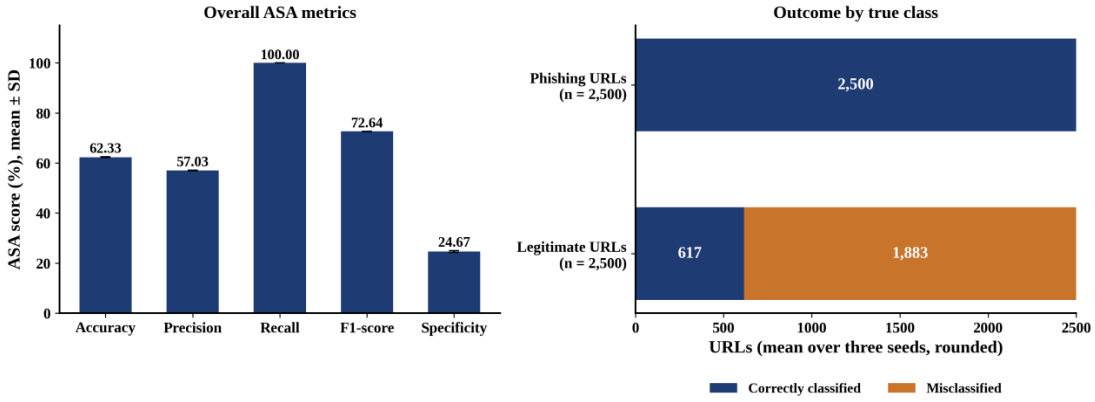

Fig. 9. Independent ASA external-test performance of RACA-Former

Fig. 9 addresses the research gap in cross-dataset generalisation by evaluating RACA-Former on independently sourced ASA URLs. The phishing recall is still high, but the specificity is 24.67%, which indicates that there are a lot of false positives when it comes to legitimate URLs. This finding reveals some cross-source discrimination, suggesting that generalisation across sources is still not possible.

4.6. Evaluation on Unseen Evasion Families

In this experiment, RACA-Former is tested on families of URLs that are structurally perturbed but not included in the training and validation sets. In contrast to tests conducted with modifications that were seen before, this

evaluation quantifies detection performance on unseen evasion patterns, and it asks whether the model generalises beyond the familiar attacks and also allows for the identification of its weaknesses when it is presented with unfamiliar URL structures.

Table 7. RACA-Former performance on held-out structural URL perturbation families

Held-out family	URLs	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Specificity (%)	FP	FN
Subdomain injection	9,898	33.20	30.60	98.47	46.69	5.62	6,567	45
URL padding	10,000	97.68	95.71	96.71	96.21	98.10	132	100
Path manipulation	9,749	99.09	99.38	97.61	98.49	99.73	18	71
Homoglyph substitution	8,895	99.17	99.80	97.75	98.76	99.90	6	68

Table 7 shows the evaluation of the performance of RACA-Former when the structural URL perturbation families not presented during training and validation are used, which is another research gap in detecting unseen evasion patterns. The three families have reasonable classification performance, but accuracy is reduced to 33.20% for subdomain injection, indicating that there is a significant generalisation weakness.

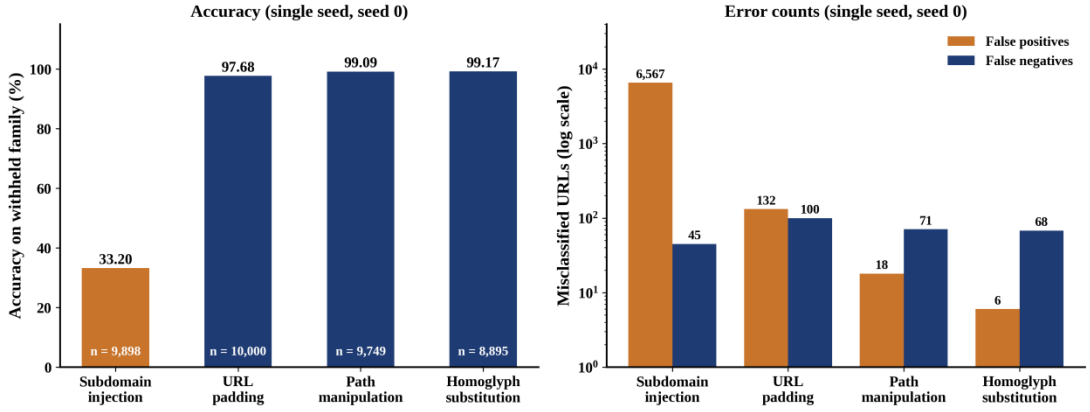

Fig. 10. Held-out evasion-family performance

Fig. 10 shows the assesses RACA-Former's ability to generalise to unseen evasion patterns by excluding each perturbation family from training and validation. Subdomain injection reduces accuracy to 33.20% and specificity to 5.62%, revealing a severe detection weakness that validation on familiar perturbations would not expose.

4.7 Ablation analysis of robustness-aware attention

An ablation study was performed to check if robustness-aware attention plays any role in the classification performance of the phishing URLs in RACA-Former. The proposed architecture was compared with a similar transformer encoder but without the attention mechanism and instead having the masked mean pooling mechanism, keeping all other parameters the same.

Table 8. Ablation of RACA-Former vs the Transformer without robustness-aware attention

Evaluation	RACA-Former	Transformer without robustness-aware attention
Clean-test accuracy	99.16 ± 0.08	99.19 ± 0.07
Clean-test F1-score	98.61 ± 0.13	98.66 ± 0.12
Clean-test recall	97.79 ± 0.30	97.87 ± 0.27
Clean-test specificity	99.76 ± 0.02	99.77 ± 0.01
Subdomain injection accuracy	97.56 ± 0.08	97.67 ± 0.04
URL padding accuracy	98.83 ± 0.27	98.97 ± 0.21
Path manipulation accuracy	99.11 ± 0.15	99.15 ± 0.10
Homoglyph substitution accuracy	99.25 ± 0.09	99.27 ± 0.05
ASA accuracy	62.33 ± 0.18	62.39 ± 0.13
ASA specificity	24.67 ± 0.36	24.79 ± 0.26
Withheld subdomain injection accuracy	33.20	32.72

(single seed)		
Withheld URL padding accuracy (single seed)	97.68	97.65
Withheld path manipulation accuracy (single seed)	99.09	99.13
Withheld homoglyph substitution accuracy (single seed)	99.17	99.24

Table 8 compares RACA-Former to a Transformer with masked mean pooling, to see the importance of robustness-aware attention. The other model performs slightly better in almost all evaluation tests, suggesting that the attention mechanism does not significantly improve performance, including against adversarial examples from unseen evasion classes.

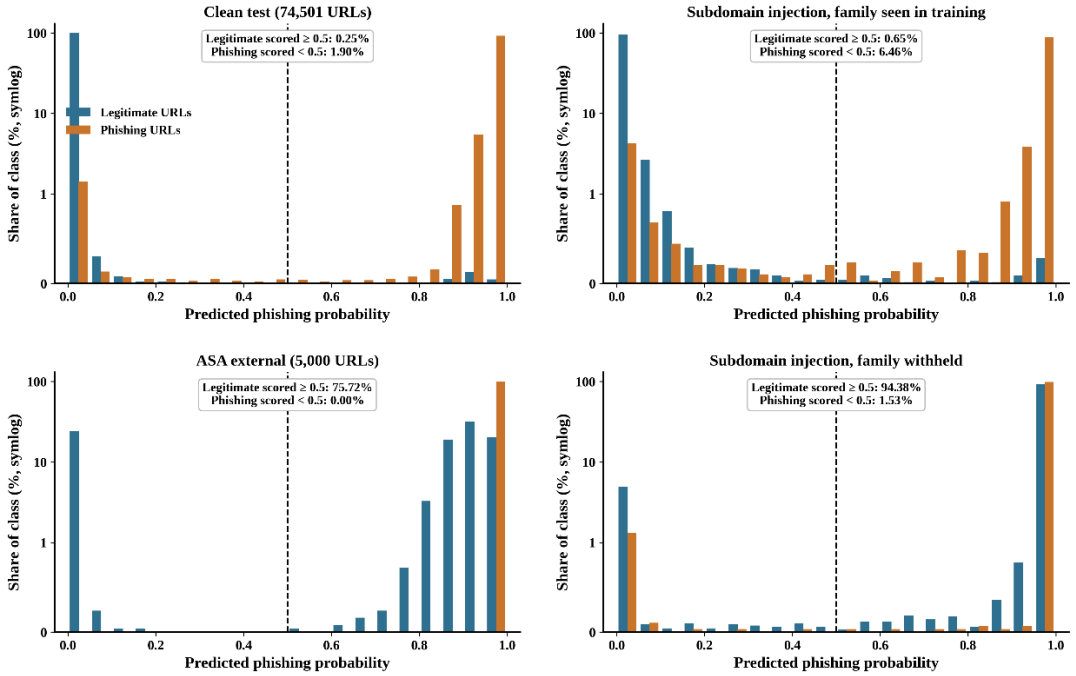

Fig. 11. Ablation comparison of RACA-Former and Transformer without robustness-aware attention**.**

Fig. 11 evaluates whether the use of attention helps RACA-Former in phishing detection task for clean, perturbation, external, and evasion unseen datasets. The results obtained by both the models were quite close, with the other model performing better than the RACA-Former in majority of the measures.

4.8. Comparative Evaluation with Existing Phishing URL Detection Methods

The comparative study compares RACA-Former against three current techniques for phishing detection in terms of their published metrics. The differences in the datasets, data split, class definition, and metric calculation impact the validity of comparison. These factors are taken into account during the interpretation of the results, whereas the results on clean-test and ASA external-test for RACA-Former are presented separately.

Table 9. Comparison with existing methods

Method	**Accuracy**	**Precision**
Deterministic model (BERT-based neural network) [21]	87.76	93.71
SemanticPhishNet [24]	96.0	96.0
Adaptive Deep URL Intelligence Network (ADUIN) [25]	95	93
RACA-Former	99.16	99.45

Table 9 presents the comparison between the accuracy and precision measures of RACA-Former and three existing methods of detecting phishing sites. RACA-Former has higher accuracy and precision scores of 99.16% and 99.45%, respectively, compared to the existing methods numerically. Nevertheless, different evaluation datasets can affect comparisons among these methods.

4.9 Statistical Learning Theory and Robustness Results

Applying Eq. (19) with $\delta = 0.05$ and $n = 74{,}501$ gives a sampling margin of 0.45 pp, so the true clean error is at most 0.84% + 0.45 pp = 1.29% with 95% confidence. The worst-case risk over the seen families is $R_{\mathrm{rob}} = 2.44\%$ (subdomain injection, $\Gamma = 0.0159$). On the ASA dataset the risk is 37.67%, so by Eq. (21) the

divergence and λ^* terms together are at least 36.8 pp. Recall achieves the maximum of 100% with specificity at only 24.67% - the false positives must therefore be on valid URLs. This confirms covariate shift rather than label shift, the latter of which would have caused the false-positive rate to remain unaffected: a mean URL length of 30.8 versus 60.2 was determined, rather than 27.53% of new families experiencing subdomain injection. If the latter evasion method is implemented, the probability is estimated to jump from 2.44% to 66.80%, around 27 times higher than the bounds deduced with clean data for previously encountered families. The bounds obtained are therefore unlikely to be guaranteed with shifting distributions or unseen attacks.

4.10 Discussion

The accuracy of RACA-Former on held-out URLs was 99.16%, while the precision was 99.45%, with a recall rate of 97.79% indicating that some phishing URLs are undetected even without modification. All the four label-validated perturbations showed that the accuracy remained between 97.56% and 99.25% while the recall decreased to 92.99% (subdomain injection). These measurements quantify stability rather than assuming it, which is typically the case with earlier phishing URL detectors, where the behaviour of the detector on a modified test data set was not measured. Generalisation was the less strong side, though. When the withheld families were independent LLM generated URLs, accuracy dropped to 62.33% (with 24.67% specificity), while the other three withheld families remained above 97%. The generalisation gap is thus not addressed, but left as a measurement, because the issues of source shift and unfamiliar subdomain structures are left unresolved. The attention module did not provide any measurable advantage compared to mean pooling, and hence is not responsible for the robustness. Comparisons with published methods are only indicative, due to differences in datasets; no significance testing was attempted. A high clean accuracy is not enough to determine reliability in varying URL structures and sources. Subdomain-aware training and source-diverse data will be targeted for future work.

5. Conclusion and Future Work

The URLs generated or modified by LLMs can pose a threat to traditional methods of URL detection by altering the structure of the URL, which could impact the effectiveness of learned classification patterns. The problem was investigated by testing RACA-Former on clean URLs, four structural perturbation families, an external test set, and an unseen-evasion test set. The accuracy, precision, and recall scores were 99.16%, 99.45%, and 97.79% respectively on the clean test set. In the perturbation families known to us, the accuracy ranged between 97.56% and 99.25%, with the largest drop in accuracy being in subdomain injection, where the phishing recall dropped to 92.99%. When the external phishing set was used for evaluation, however, accuracy was 62.33% and specificity 24.67% while phishing recall was high. The subdomain experiment with the held out set also revealed a significant vulnerability as the accuracy dropped to 33.20% and specificity to 5.62%. Furthermore, the results of the ablation study revealed no measurable consistent gain in robustness with robustness-aware attention compared to masked mean pooling. These results show that high clean-test performance does not guarantee reliable generalisation to independently generated URLs or previously unseen structural evasion patterns. These results highlight the need for high quality results in the absence of any evasion in previously unseen perturbed URLs, or across different data sources. Due to the limitations in the study, in particular the single seed held-out evaluation and the lack of statistical significance testing results for the ablation experiments, it is unclear what the specific contribution of the attention mechanism is, and what is the variability of unseen-evasion performance.

Future work should investigate subdomain-aware representations and training strategies that improve generalisation to previously unseen URL modifications. To evaluate the performance in more distribution shifts, other independently collected datasets and LLM-generated phishing samples should be used. Future experiments should be conducted using repeated runs and incorporate statistical significance testing comparing attention and pooling mechanisms. Finally, threshold selection and false-positive reduction should be explored using validation data to improve the distinction between phishing and legitimate URLs while maintaining high phishing recall. These directions are particularly important for improving cross-source generalisation and reducing the severe performance degradation observed under unseen subdomain-based evasion

Reference

1. R. Goenka, M. Chawla, and N. Tiwari, “A comprehensive survey of phishing: mediums, intended targets, attack and defence techniques and a novel taxonomy,” *Int. J. Inf. Secur.*, vol. 23, no. 2, pp. 819–848, Apr. 2024, doi: 10.1007/s10207-023-00768-x.
2. S. Remya, M. J. Pillai, K. K. Nair, S. Rama Subbareddy, and Y. Y. Cho, “An Effective Detection Approach for Phishing URL Using ResMLP,” *IEEE Access*, vol. 12, pp. 79367–79382, 2024, doi: 10.1109/ACCESS.2024.3409049.
3. M. A. Tamal, M. K. Islam, T. Bhuiyan, and A. Sattar, “Dataset of suspicious phishing URL detection,” *Front. Comput. Sci.*, vol. 6, p. 1308634, Mar. 2024, doi: 10.3389/fcomp.2024.1308634.
4. A. Kulkarni, V. Balachandran, and T. Das, “Phishing Webpage Detection: Unveiling the Threat Landscape and Investigating Detection Techniques,” *IEEE Commun. Surv. Tutor.*, vol. 27, no. 2, pp. 974–1007, 2025, doi: 10.1109/COMST.2024.3441752.

5. E. Kytidou, T. Tsikriki, G. Drosatos, and K. Rantos, “Machine learning techniques for phishing detection: A review of methods, challenges, and future directions,” *Intell. Decis. Technol.*, vol. 19, no. 6, pp. 4356–4379, Nov. 2025, doi: 10.1177/18724981251366763.
6. S. Asiri and N. Alasmari, “Phishing webpage detection using structured URL generation,” *Sci. Rep.*, vol. 16, no. 1, p. 31327, Jul. 2026, doi: 10.1038/s41598-026-63981-3.
7. S. K and M. Naik S, “A novel framework for effective phishing URL detection using an LSTM-based siamese network,” *Knowl.-Based Syst.*, vol. 329, p. 114271, Nov. 2025, doi: 10.1016/j.knosys.2025.114271.
8. F. Ji and D. Kim, “How Can We Effectively Use LLMs for Phishing Detection?: Evaluating the Effectiveness of Large Language Model-based Phishing Detection Models,” 2025, *arXiv*. doi: 10.48550/ARXIV.2511.09606.
9. D. Chiba, H. Nakano, and T. Koide, “PhishLumos: From a Single URL to Campaign-Level Phishing Mitigation,” *IEEE Access*, vol. 14, pp. 79716–79739, 2026, doi: 10.1109/ACCESS.2026.3696597.
10. Er. Kritika, “A comprehensive literature review on phishing URL detection using deep learning techniques,” *J. Cyber Secur. Technol.*, vol. 9, no. 4, pp. 315–343, Oct. 2025, doi: 10.1080/23742917.2024.2378552.
11. J. Lee and H. Kwon, “Crafting evasive phishing URLs: exploiting tokenizer vulnerabilities in transformer-based detection systems,” *Int. J. Mach. Learn. Cybern.*, vol. 17, no. 8, p. 390, Aug. 2026, doi: 10.1007/s13042-026-03239-6.
12. F. Trad and A. Chehab, “Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models,” *Mach. Learn. Knowl. Extr.*, vol. 6, no. 1, pp. 367–384, Feb. 2024, doi: 10.3390/make6010018.
13. A. H. Nasution, W. Monika, A. Onan, and Y. Murakami, “Benchmarking 21 Open-Source Large Language Models for Phishing Link Detection with Prompt Engineering,” *Information*, vol. 16, no. 5, p. 366, Apr. 2025, doi: 10.3390/info16050366.
14. H. Kibriya, R. Amin, S. S. Alshamrani, S. Rehman, M. Hassan, and F. S. Alsubaei, “Lightweight malicious URL detection using deep learning and large language models,” *Sci. Rep.*, vol. 15, no. 1, p. 43044, Dec. 2025, doi: 10.1038/s41598-025-26653-2.
15. A. Chaudhuri and M. B, “Adversarial-resilient lightweight phishing url detection: Evaluating lexical & metadata features under evasion techniques,” *Sci. Rep.*, vol. 16, no. 1, p. 26668, Jul. 2026, doi: 10.1038/s41598-026-60046-3.
16. M. Yıldırım, “Adversarial Robustness in URL-Based Phishing Detection: Problem-Space Evaluation and Robust Feature Engineering,” *Appl. Sci.*, vol. 16, no. 17, p. 8737, Sep. 2026, doi: 10.3390/app16178737.
17. A. Prasad, V. Yadav, C. Solanki, H. Goswami, T. Jha, and D. Nagal, “StealthPhisher: A defensive framework against phishing attack using hybrid deep learning and GenAI,” *Expert Syst. Appl.*, vol. 299, p. 130205, Mar. 2026, doi: 10.1016/j.eswa.2025.130205.
18. W. Han, J. Zhu, C. Zhang, Z. Zhang, Y. Mei, and L. Wang, “Phish-Master: Leveraging Large Language Models for Advanced Phishing Email Generation and Detection,” *Appl. Sci.*, vol. 15, no. 22, p. 12203, Nov. 2025, doi: 10.3390/app152212203.
19. A. Brissett and J. Wall, “Machine Learning and Watermarking for Accurate Detection of AI-Generated Phishing Emails,” *Electronics*, vol. 14, no. 13, p. 2611, Jun. 2025, doi: 10.3390/electronics14132611.
20. F. Heiding, B. Schneier, A. Vishwanath, J. Bernstein, and P. S. Park, “Devising and Detecting Phishing Emails Using Large Language Models,” *IEEE Access*, vol. 12, pp. 42131–42146, 2024, doi: 10.1109/ACCESS.2024.3375882.
21. H. Ghalechyan, E. Israyelyan, A. Arakelyan, G. Hovhannisyan, and A. Davtyan, “Phishing URL detection with neural networks: an empirical study,” *Sci. Rep.*, vol. 14, no. 1, p. 25134, Oct. 2024, doi: 10.1038/s41598-024-74725-6.
22. J. K. S. KAITHOLIKKAL, “Phishing URL dataset.” Mendeley Data, Apr. 02, 2024. doi: 10.17632/VFSZBJ9B36.1.
23. A. B. Antu, A. Paul, and S. Rahman, “ASA_Phishing_URL_Dataset.” Mendeley Data, Aug. 03, 2026. doi: 10.17632/KPMVHMTR5F.1.
24. Snousi, H. M., & Aleej, F. A. (2025). Machine learning-augmented partial differential equation solvers for high-fidelity engineering design optimization. Journal of Applied Mathematical Models in Engineering, 34-40.
25. R. Gobinath and S. Manikandan, “Deep learning-based phishing classification framework for accurate detection using optimized URL intelligence,” *Sci. Rep.*, vol. 16, no. 1, p. 15794, Apr. 2026, doi: 10.1038/s41598-026-46481-2.
26. E.-H. Qazi, M. H. Faheem, and A. Almorjan, “Phishing URL Detection Using Transformer-Based Architecture and Contextual Content Features,” *Computers*, vol. 15, no. 6, p. 335, May 2026, doi: 10.3390/computers15060335.