WebGuard++:Interpretable Malicious URL Detection via Bidirectional Fusion of HTML Subgraphs and Multi-Scale Convolutional BERT

1st Ye Tian
Hangzhou Research Institute
Xidian University
Hangzhou, China
tianye@xidian.edu.cn

2nd ZhangYumin
Hangzhou Research Institute
Xidian University
Hangzhou, China
24241214904@stu.xidian.edu.cn

3rd Yifan Jia
Yantai Research Institute
Harbin Engineering University
Yantai, China
jiayf@hrbeu.edu.cn

4th Jianguo Sun*
Hangzhou Research Institute
Xidian University
Hangzhou, China
jgsun@xidian.edu.cn

5th Yanbin Wang*
Hangzhou Research Institute
Xidian University
Hangzhou, China
wangyanbin15@mails.ucas.ac.cn

***Abstract*—URL+HTML feature fusion shows promise for robust malicious URL detection, since attacker artifacts persist in DOM structures. However, prior work suffers from four critical shortcomings: (1) incomplete URL modeling, failing to jointly capture lexical patterns and semantic context; (2) HTML graph sparsity, where threat-indicative nodes (e.g., obfuscated scripts) are isolated amid benign content, causing signal dilution during graph aggregation; (3) unidirectional analysis, ignoring URL-HTML feature bidirectional interaction; and (4) opaque decisions, lacking attribution to malicious DOM components.**

To address these challenges, we present WebGuard++, a detection framework with 4 novel components: 1) Cross-scale URL Encoder: Hierarchically learns local-to-global and coarse to fine URL features based on Transformer network with dynamic convolution. 2) Subgraph-aware HTML Encoder: Decomposes DOM graphs into interpretable substructures, amplifying sparse threat signals via Hierarchical feature fusion. 3) Bidirectional Coupling Module: Aligns URL and HTML embeddings through cross-modal contrastive learning, optimizing inter-modal consistency and intra-modal specificity. 4) Voting Module: Localizes malicious regions through consensus voting on malicious subgraph predictions. Experiments show WebGuard++ achieves significant improvements over state-of-the-art baselines, achieving 1.1×–7.9× higher TPR at fixed FPR of 0.001 and 0.0001 across both datasets.

***Index Terms*—Malicious URL Detection, Multiscale Learning, ConvBERT, Pyramid Attention**

I. Introduction

Phishing attacks have become one of the most pervasive and damaging cyber threats in recent years, with the Anti-Phishing Working Group (APWG) reporting record-breaking volumes of 1,624,144 attacks in Q1 2023 [1] and approximately one million in Q1 2024 [2]. Modern phishing campaigns employ increasingly sophisticated evasion techniques, including homoglyphic domain spoofing (e.g., "facbook.com"), URL shortening services, and malicious content embedded within legitimate platforms. These deceptive practices enable attackers to bypass traditional URL detection methods, such as blacklists [3], [4], rules [5] and manual feature engineering [6], while effectively mimicking trusted websites to steal sensitive information. The growing sophistication and scale of these threats underscore the critical need for robust, accurate, and efficient phishing detection systems capable of operating at web scale with near-zero false positive rates to adequately protect users and infrastructure.

Most approaches to malicious URL detection have predominantly relied on URL-based features [7]. However, URLs present several critical limitations: 1) Limited Information Dimensions – URLs encode minimal contextual data, restricting detection models to surface-level patterns. 2) Vulnerability to Evasion – Attackers easily manipulate URLs through obfuscation, mimicry, or rapid replacement to bypass detection. 3) Lack of Structural Insight – URLs alone cannot reveal the underlying page behaviors that indicate malicious intent.

Integrating HTML structural analysis has the potential to address these shortcomings. Unlike URLs, HTMLs contain: 1) Rich Hierarchical Structure – DOM trees, nested iframes, and script dependencies expose hidden attack vectors (e.g., phishing content loaded via iframes). 2) Interaction Logic – Form actions, redirects, and domain mismatches reveal data exfiltration attempts (e.g., spoofed submission endpoints). By training models to recognize these structural anomalies, HTML+URL detection complements and reinforces URL-based features and mitigates evasion tactics, as structural manipulations are harder to disguise than URL alterations.

Existing approaches like Web2Vec and PhishDet demonstrate progressive advancements. Web2Vec proposes a deep hybrid network architecture that jointly processes URL strings,

*Yanbin Wang and Jianguo Sun are co-corresponding authors.

HTML content, and DOM structures. PhishDet combines Long Short-Term Memory (LSTM) networks with Graph Convolutional Networks (GCNs) to model URL patterns and structural HTML features. However, these methods exhibit 4 critical limitations:

- Incomplete URL Modeling: Existing methods process URLs as either lexical sequences or static tokens, failing to capture (i) local character-level manipulations (e.g., "facwook.com") and (ii) global semantic context (e.g. deceptive subdomains).
- HTML Graph Sparsity: Current GNN-based DOM analyzers suffer from signal dilution during neighborhood aggregation, as threat-indicative nodes (e.g., obfuscated <script> tags) account for <5% of typical DOM graphs, while benign content dominates attention weights.
- Deficient Cross-Modal Interaction: Existing methods have not modeled dynamic, bidirectional semantic relationships between modalities, missing mutually reinforcing signals. For instance, a suspicious URL (e.g., "login.paypa1.com") may align with HTML structures like payment forms, while anomalous DOM elements (e.g., fake brand-related notices) can clarify URL intent.
- Opaque Decisions: Black-box architectures lack attribution mechanisms to pinpoint malicious DOM components, hindering forensic analysis.

We propose WebGuard++ to overcome cross-modal phishing detection challenges with four interlocking technical: (1) Cross-scale URL Encoder: Combines ConvBERT's hierarchical representations with spatial pyramid fusion to jointly capture character-level obfuscational patterns (e.g., "paypa1") and semantic inconsistencies (e.g., deceptive subdomains). (2) Subgraph-aware HTML Encoder: Performs DOM subgraph partitioning with stabilized node grouping and iterative batch sampling, enabling localized malicious signal aggregation (e.g., form clusters) while mitigating benign node interference in full-graph processing. (3) Bidirectional Coupling Module: Employs stacked feature layers with both self and cross-attention to capture distinct semantic subspaces, enabling bidirectional URL-HTML feature interaction. (4) Voting Module: Adopts a minimum compromise voting policy - any malicious subgraph (≥ 1 in sampled rounds) triggers global malicious classification, while providing actionable forensic evidence through malicious subgraphs.

This work makes the following key contributions:

- We propose a cross-modal malicious URL detection that achieves 1.1-7.9× higher TPR at <0.01 FPR by jointly modeling URL and HTML features.
- We design a URL encoding method that captures both lexical obfuscation patterns and semantic inconsistencies by multi-layer ConvBERT with pyramidal feature fusion.
- We propose a subgraph learning method for HTML that employs effective subgraph partitioning to aggregate local malicious signals while preventing feature dilution.
- We use a hybrid attention network with both self/cross-attention to learn bidirectional, multi-view relationships between URLs and HTML content.
- Our method is the first subgraph-based maliciousness prediction that provides both fine-grained classification and component-level traceability.

II. Related Work

Early phishing detection methods mainly rely on extracting discriminative features from raw URLs. PhishDef [8] demonstrated that phishing links could be effectively identified using only static URL features, such as domain length and character composition, thereby reducing reliance on external resources. Nonetheless, it showed limited effectiveness against semantically natural and structurally sophisticated malicious URLs. PhishZoo [9] enhanced the ability to detect spoofed pages by analyzing visual similarities between webpages, but its stability degraded under dynamic content or slight layout changes. Whereupon, Sahingoz et al [10] modeled URL character features using various classifiers, significantly improving detection accuracy and generalization capabilities. Shraddha Parekh et al. [11] put forth a model by using the URL detection method using Random Forest algorithm. However, there are still some defects, such as the lack of fine-grained feature acquisition of URL text.

Despite these advancements, single-modality detection methods [12] remain vulnerable to adaptive attacks that exploit their limited perceptual scope. With the advent of deep learning, more expressive models emerged. URLNet [13] combined character-level and word-level embeddings via convolutional neural networks (CNNs) to capture morphological patterns in URLs, offering significant improvements over handcrafted features. This line of research was extended through CNN-based architectures [14] and attention-based transformers such as TransURL [15], which demonstrated stronger robustness against adversarial obfuscation. Concurrently, models such as PhishGuard [16] and Fed-urlBERT [17] integrated federated training and transformer encoders to support privacy-preserving and scalable learning. PhishBERT [18] further explored pre-trained language models for URL representation learning, yielding enhanced generalization.

In response to the continuously evolving phishing techniques, researchers have gradually shifted towards multi-modal fusion and structure-aware modeling [19]–[27]. Yoon et al. [28] proposed a detection framework that integrates HTML DOM graphs and URL features based on graph convolutional and transformer networks. Lee et al. [29] introduced a brand consistency verification mechanism, effectively enhancing performance under adversarial attacks, although challenges remain in identifying emerging niche brands. PhishAgent [30] further improved detection robustness by leveraging a multi-modal large language model to integrate webpage text, visual, and structural information. Lihui Meng et al. [31] proposes DPMLF (Deep Learning Phishing Detection Model with Multi-Level Features), which integrates URL character-level and HTML word-level semantic features. However, feature fusion using fully connected layers cannot closely match the modality, thus affecting the model performance.

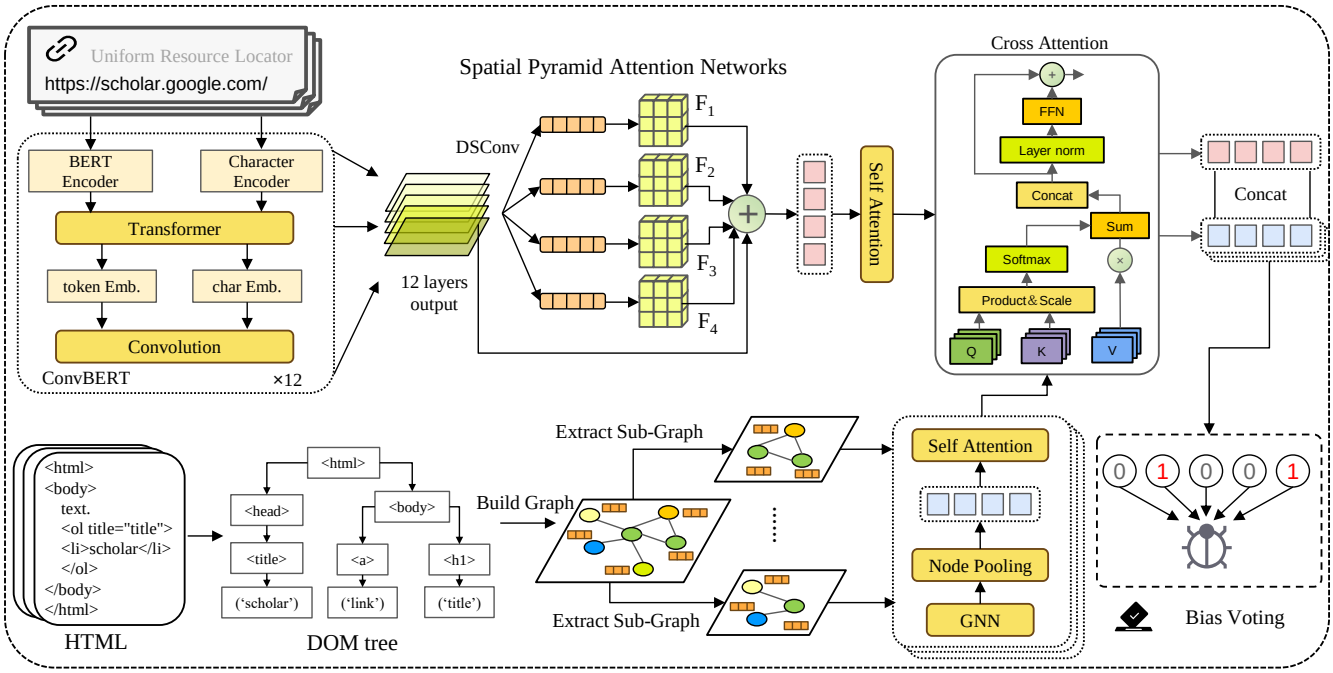

Fig. 1. Framework diagram of the model structure of WebGuard++.

In parallel, PhishIntention [21] and Phishpedia [20] emphasized visual understanding. A broader evaluation by Bushra Sabir et al. [32] revealed that many state-of-the-art models exhibit drastic performance degradation when facing adversarial URL samples, highlighting the fragility of current systems.

Our approach fundamentally advances malicious URL detection by differentiating itself from prior work through four key dimensions: (1) fine-grained URL feature extraction, (2) subgraph-level HTML structure learning, (3) bidirectional modal coupling, and (4) malicious segment localization.

III. Methodology

We organize this section as follows: First introducing data preprocessing, then detailing three core components (URL encoder, HTML encoder, and their bidirectional coupling), and finally presenting the phishing detection mechanism. See Figure 1 for overview.

A. Cross-scale URL Encoder

URLs serve as a fundamental indicator for malicious webpage detection, as attackers frequently manipulate URL structures to mimic benign pages while evading traditional pattern-matching techniques and retaining malicious intent. However, extracting discriminative URL features requires careful consideration of two key challenges: (1) structural ambiguity; and (2) adversarial noise—embedded homoglyphs or Base64-encoded payloads. This necessitates fine-grained URL feature extraction capable of capturing multiscale information at both character-level and semantic levels.

To address this, we integrate CharBERT and ConvBERT models, where the former extracts character-level URL features, while the latter, equipped with global and local context learning, refines local and global semantic representations. Furthermore, to enable coarse-to-fine representation learning, we extract embeddings from all 12 hidden layers of the ConvBERT model to construct a feature matrix, which is then processed via a spatial pyramid multiscale feature fusion module for hierarchical feature learning.

The spatial pyramid module first applies DSConv3×3 (depthwise separable convolution) to preliminarily transform input features, as follows:

$$X = DSConv3 \times 3(x) \tag{1}$$

where the input is x and the output is X. Next, multiple DSConv3x3 convolutional kernel branches with different expansion rates are defined, where different expansion rates can capture contextual information at different scales. Smaller expansion rates can focus on local features, while larger expansion rates can capture more global features. Each branch uses the depth-separable convolution DSConv3x3 with expansions d_1, d_2, d_3, d_4.

$$D_1 = DSConv3 \times 3(X, d_i), i = 1, 2, 3, 4 \tag{2}$$

where D_i denotes the output of the i^{th} branch and d_i is the expansion rate of the branch.

Sum the outputs of all branches with the output of the initial convolution to achieve feature fusion.

$$DX = X + \sum_{i=1}^{4} D_i \tag{3}$$

Next, along the lines of Spatial Pyramid Attention Networks [33], pooling operations are performed on the feature

maps using different sizes of Adaptive Average Pooling Layers (AdaptiveAvgPool2d) to extract features at different spatial scales.

$$y_1 = AdaptiveAvgPool2d(1)(DX) \tag{4}$$

$$y_2 = AdaptiveAvgPool2d(2)(DX) \tag{5}$$

$$y_3 = AdaptiveAvgPool2d(4)(DX) \tag{6}$$

where AdaptiveAvgPool2d(k) denotes the adaptive pooling of the feature map to a size of $k \times k$.

Finally, these feature vectors are spliced and input to the fully connected layer for feature fusion, and the attention weights are obtained using the sigmoid function.

$$y = concat(y_1, y_2, y_3) \tag{7}$$

$$y = FC_2(ReLU(FC_1(y))) \tag{8}$$

$$y = sigmoid(y) \tag{9}$$

where FC_1 and FC_2 denote the fully connected layers, respectively.

The attention weights are multiplied with the fused feature map to obtain the attention weighted feature map. The attention weighted feature map is summed with the original input feature map to realize the residual join to preserve the information of the original input.

$$O_{url} = DX \times y + x \tag{10}$$

B. Subgraph-aware HTML Encoder

The input HTML document is parsed using the Beautiful Soup library (v4.12.3) to construct a Document Object Model (DOM) tree representation. This tree preserves the hierarchical structure of HTML elements, including all tags, attributes, and text content. The DOM tree is traversed using a depth-first search (DFS) algorithm to generate: A diagram object for the NetworkX package (v3.1) where: Nodes represent HTML elements (tags) with unique identifiers. Edges encode parent-child relationships between elements. A node attribute list containing structured metadata for each node: Tag type (e.g., , <a>), Key-value pairs of HTML attributes (e.g., {"class": "container"}), Raw text content. Node features consist of node "tag", "attributes", "text" attributes text content. For each node, we use a pretrained Word2Vec model (minimum vocabulary size) to generate a 100-dimensional embedding.

Then, the corresponding edge matrices, node neighbor lists, and maximum number of neighbors are computed based on the edge and node data of the networkx.DiGraph() graph object, and all the data are integrated into the model-acceptable S2VGraph object structure.

For HTML content, our objective is to learn effective representations of malicious signals embedded within its structure. However, since such signals are often concealed within normal HTML elements, they tend to be obscured during graph-based learning. To address this challenge, we propose subgraph-aware HTML graph learning, which transforms the HTML DOM into a graph structure and employs node-level subgraph partitioning to extract localized subgraphs. This approach enables subgraph-level feature learning, ensuring that malicious signals remain distinguishable and are not diluted by benign structural patterns.

Algorithm 1 Biased voting mechanism process.

$subgraphs \leftarrow sum(division\_func(graph))$
0count $\leftarrow 0$
1count $\leftarrow 0$
0scores $\leftarrow 1$
1scores $\leftarrow 0$
for 0 to $iters\_per - 1$ **do**
  selected_subgraphs $\leftarrow random.sample(subgraphs, 4)$
  outputs $\leftarrow model(selected\_subgraphs, *url)$
  scores $\leftarrow$ softmax$(outputs)$
  _, predicted_classes $\leftarrow \max(outputs, 1)$
  if $predicted == 0$ **then**
    $0count \leftarrow 0count + 1$
    $0scores \leftarrow \min(0scores, scores)$
  else
    $1count \leftarrow 1count + 1$
    $1scores \leftarrow \max(1scores, scores)$
  end if
end for
$y\_pred \leftarrow []$
$y\_scores \leftarrow []$
if $1count \geq 2$ **then**
  1 add to y_pred
  $1scores$ add to y_scores
else
  0 add to y_pred
  $0scores$ add to y_scores
end if
return: y_pred, y_scores

Specifically, we partition the graph into N subgraphs using a hash function H. The function takes a node ID string (S_v) as input and assigns nodes to distinct subsets, where each subgraph retains only the nodes and edges belonging to a specific group.

$$X^t = \{X_v | H(S_v)\%T_f + 1 = t\}, t = 1, 2, ..., T_f \tag{11}$$

Where G=(V,E,X), G denotes the input graph, V denotes the node union, E denotes the edge union, X denotes the node feature matrix and X_v denotes the feature vector of node v.

Next, we will select batch subgraphs to input into the model many times for neighbor aggregation, MLP nonlinear transformation and BatchNorm operations.

$$X_{concat} = Concat(G_1.node_features, ..., G_B.node_features) \tag{12}$$

$$H_0 = X_{concat} \tag{13}$$

$$H^{(l)} = A_{block} \cdot H^{(l-1)} \tag{14}$$

$$H^{(l)} = ReLU(BatchNorm(MLP_l(H^{(l)}))) \tag{15}$$

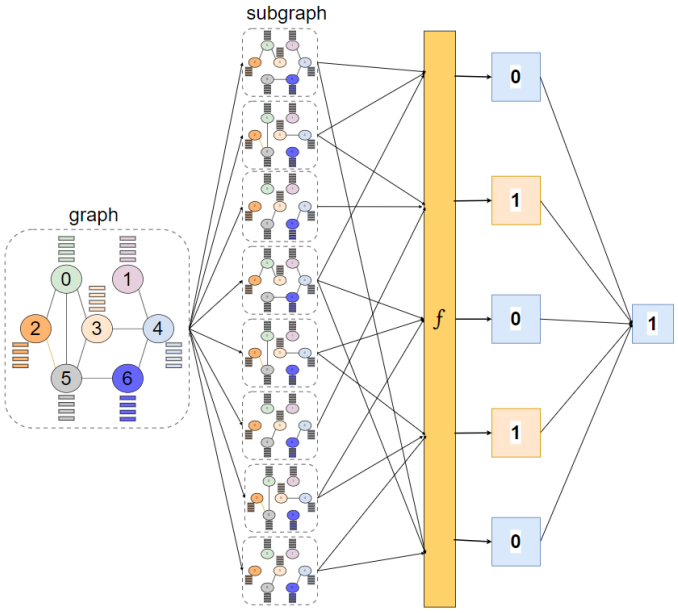

Fig. 2. Biased voting mechanism process.

Where B denotes the batch size, G_1 represents the 1^{st} subgraph, l denotes the l^{th} propagation, $H^{(l)}$ denotes the hidden representation of each layer, A^{block}denotes the block diagonal sparse matrix.

Finally, all sub-graphs of each layer are pooled at the subgraph level, and the features of each layer are concatenated to obtain the final features of these sub-graphs.

$$H_{pooled}{}^{(i)} = P \cdot H^{(i)}, i = 0, 1, ...l \tag{16}$$

Where P represents the pooling operation.

C. Bidirectional Coupling Module

To learn joint URL-HTML representations, we propose bidirectional multi-view coupling ([34]) for fusing multimodal URL and HTML features. Unlike simple concatenation or unidirectional cross-attention layers, our bidirectional coupling module stacks multiple hybrid attention layers. Each layer combines self-attention and cross-attention mechanisms: self-attention independently models intra-modal dependencies within each modality, while cross-attention captures inter-modal interactions between URL and HTML features.

The HTML modality features are processed through self-attention to capture intra-modal relationships:

$$SelfAttention(Q_h, K_h, V_h) = softmax(\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right))V_h \tag{17}$$

Similarly, URL features undergo self-attention:

$$SelfAttention(Q_u, K_u, V_u) = softmax(\left(\frac{Q_u K_u^T}{\sqrt{d_k}}\right))V_u \tag{18}$$

We establish bidirectional cross-modal attention for inter-modal information exchange:

HTML-to-URL attention extracts domain semantics from URL features:

$$CrossAttention(Q_u, K_h, V_h) = softmax(\left(\frac{Q_u K_h^T}{\sqrt{d_k}}\right))V_h \tag{19}$$

URL-to-HTML attention captures structural patterns from HTML features:

$$CrossAttention(Q_h, K_u, V_u) = softmax(\left(\frac{Q_h K_u^T}{\sqrt{d_k}}\right))V_u \tag{20}$$

The fused features are subsequently output.

D. phishing bias voting mechanism

we propose a biased voting mechanism for phishing website detection. If the number of malicious predictions across multiple rounds of batch subgraph extraction exceeds one, we directly classify the corresponding HTML-URL pair as a phishing website. Unlike conventional approaches, our mechanism operates at the subgraph level, enabling not only final detection but also localization of malicious regions within the HTML structure. The voting process is illustrated in Figure 2, and the algorithmic workflow is detailed in Algorithm 1.

Specifically, for each HTML graph divided into num_groups of subgraphs, we will perform iter_per rounds to randomly extract iter_num subgraphs and URL data corresponding to the current HTML graph from the subgraphs and input them into the WebGuard++ model for feature extraction. WebGuard++'s Subgraph-aware HTML Encoder will batch process the subgraph features without affecting each other. Subsequently, WebGuard++ fuses the output subgraph features from the Subgraph-aware HTML Encoder to represent the total features of the subgraph collection in this round of extraction.

Suppose the current is a phishing site, num_group=5, iter_per=5, iter_num=4, a subgraph contains malicious. Then, 4 images are randomly selected inside 5 images, and the probability that 4 images contain malicious subgraphs is 80%; the probability that none of the set of subgraphs of the extraction site contains malicious subgraphs in each of the 5 rounds of random selection is 0.032%, which is close to zero.

$$\text{Contains malicious} = \frac{C(4,3)}{C(5,4)} = \frac{4}{5} = 80\% \tag{21}$$

$$\text{Without malicious} = 20\% \tag{22}$$

$$\text{All without malice} = (20\%)^5 = 0.032\% \tag{23}$$

Therefore, our method can basically extract subgraphs with full coverage and input them to the model for prediction.

To ensure the accuracy of the predictions, we set the condition for determining URL-HTML data pairs as malicious to be when the number of predictions as malicious in all extraction rounds is greater than 1. When a batch of subgraph features is predicted to be malicious, the current URL-HTML data pair is tagged as malicious. If a subsequent batch of subgraph features

TABLE I
MTLP Dataset 10000 Evaluation metrics at scale: WebGuard++ vs other models.

Method	TN	FP	FN	TP	ACC	Precision	Recall	F1	PR-AUC	ROC-AUC	MCC	Weighted F1
BiLSTM	940	54	922	84	0.5120	0.6086	0.0834	0.1468	0.5118	0.5145	0.0575	/
TextCNN	814	180	104	902	0.8580	0.8336	0.8966	0.8639	0.9472	0.9434	0.7179	0.8577
URLNet	829	165	148	857	0.9269	0.8385	0.8527	0.8455	0.7891	0.8433	0.6869	0.8434
TransURL	953	41	20	986	0.9695	0.9600	0.9801	0.9699	0.9934	0.9937	0.9391	0.9694
PMANet	991	26	22	961	0.9760	0.9736	0.9776	0.9756	0.9933	0.9952	0.9519	0.9760
URLBERT	910	42	11	1037	0.9735	0.9610	**0.9895**	0.9750	**0.9965**	**0.9967**	0.9472	0.9734
DEPHIDES	966	36	76	922	0.9440	0.9624	0.9238	0.9427	0.9824	0.9827	0.8887	0.9439
Semi-GAN	790	151	17	948	0.9118	0.8626	0.9823	09186	/	0.9826	0.8316	0.9113
WebGuard++	989	21	24	966	**0.9775**	**0.9787**	0.9757	**0.9772**	0.9959	0.9949	**0.9549**	**0.9772**

TABLE II
The Abdelhakim Dataset phishing dataset Evaluation metrics : WebGuard++ vs other models.

Method	TN	FP	FN	TP	ACC	Precision	Recall	F1	PR-AUC	ROC-AUC	MCC	Weighted F1
BiLSTM	115	2	58	13	0.6808	0.8666	0.1830	0.3023	0.4671	0.5830	0.2970	/
TextCNN	103	14	42	29	0.7021	0.6744	0.4084	0.5087	0.6462	0.7647	0.3333	0.6814
URLNet	90	26	40	31	0.8128	0.5438	0.4366	0.4843	0.4513	0.6062	0.2240	0.6378
TransURL	101	16	20	51	0.8085	0.7611	0.7183	0.7391	0.8210	0.8737	0.5886	0.8073
PMANet	98	14	14	62	0.8510	0.8157	0.8157	0.8157	0.8827	0.8956	0.6907	0.8510
URLBERT	88	18	16	66	0.8191	0.7857	0.8048	0.7951	0.8772	0.9004	0.6334	0.8193
DEPHIDES	104	1	67	16	0.6382	**0.9411**	0.1927	0.3200	0.7898	0.7840	0.3172	0.5621
Semi-GAN	88	24	34	35	0.6795	0.5932	0.5072	0.5468	0.4759	0.7017	0.3035	0.6738
WebGuard++	109	8	12	59	**0.8936**	0.8805	**0.8309**	**0.8550**	**0.9072**	**0.9256**	**0.7719**	**0.8550**

TABLE III
MTLP Dataset 10000 in terms of TPR@FPR metrics: WebGuard++ vs other models.

Method	TPR@FPR (0.0001)	TPR@FPR (0.001)	TPR@FPR (0.01)	TPR@FPR (0.1)
BiLSTM	0	0	0	0.0834
TextCNN	0	0.2345	0.4771	0.8429
URLNet	0	0	0	0
TransURL	0.3797	0.3797	0.8876	0.9960
PMANet	0.2177	0.2553	0.9369	0.9979
URLBERT	0.4122	0.4122	0.9408	**0.9990**
DEPHIDES	0.1002	0.1002	0.8086	0.9829
Semi-GAN	0	0	0.7709	0.9544
WebGuard++	**0.6848**	**0.7939**	**0.9515**	0.9929

TABLE IV
The Abdelhakim Dataset phishing dataset in terms of TPR@FPR metrics: WebGuard++ vs other models.

Method	TPR@FPR (0.0001)	TPR@FPR (0.001)	TPR@FPR (0.01)	TPR@FPR (0.1)
BiLSTM	0	0	0	0.1830
TextCNN	0	0	0.0704	0.3521
URLNet	0	0	0	0
TransURL	0.1126	0.1126	0.3521	0.6338
PMANet	0.2105	0.2105	0.4605	0.7368
URLBERT	0.3414	0.3414	0.3536	0.6707
DEPHIDES	0.1927	0.1927	0.2771	0.5421
Semi-GAN	0	0	0	0.3623
WebGuard++	**0.3802**	**0.3802**	**0.5492**	**0.8591**

is also predicted to be malicious, the previous prediction is confirmed to be true, indicating that the HTML graph does indeed contain a subgraph with malicious content, and the current URL-HTML data pair is malicious.

IV. Experiments

In this section, we conduct extensive experiments to evaluate the proposed method. First, we describe the experimental setup, including the dataset, evaluation metrics, environment and equipment. Then, we compare the phishing site detection performance of our proposed method with other state-of-the-art methods. Finally, we perform some ablation experiments, cross-dataset tests and model robustness tests.

A. EXPERIMENTAL SETUP

Dataset. Our experiments focus on using the MTLP Dataset [35] phishing site detection dataset, the Abdelhakim Dataset phishing site detection dataset, and the course-cotrain-data course categorization dataset.

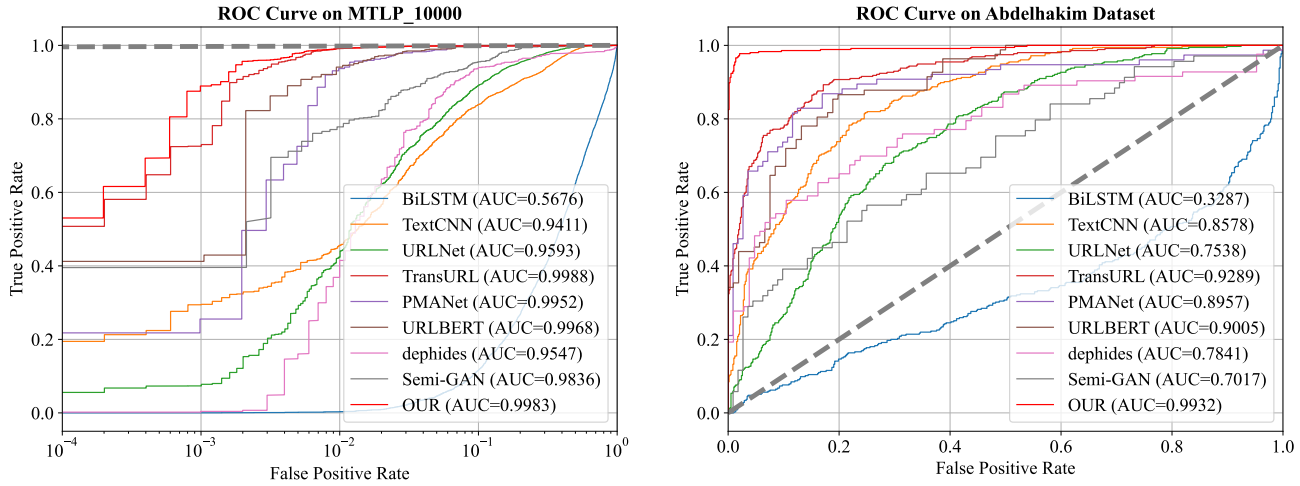

Fig. 3. Performance of our model against other models on the ROC curve.

The MTLP dataset is compiled from two different sources: benign samples from the top 2,000 URLs in the Alexa rankings, and additional randomly selected benign and malicious URLs from OpenPhish. The dataset consists of 50,000 benign URLs and 50,000 malicious URLs, which contain HTML content, whois information, and screenshots. Due to the nature of the experiments and equipment limitations, we cleaned the MTLP Dataset and evenly sampled 10,000 data as the training evaluation dataset for this study.

The Abdelhakim Dataset includes 11430 URLs. The dataset are designed to be used as a benchmark for machine learning based phishing detection systems. The datatset is balanced, it containes exactly 50% phishing and 50% legitimate URLs. Datasets are constructed on May 2020. The dataset contains a list a URLs together with their DOM tree objects that can be used for replication and experimenting new URL and content-based features overtaking short-time living of phishing web pages.

The course-cotrain-data dataset consists of 1051 pages, with 230 in the course category and 821 in the non-course category. This data set contains a subset of the WWW-pages collected from computer science departments of various universities in January 1997 by the World Wide Knowledge Base (Web->Kb).

Evaluation Metrics. To evaluate the model in a more comprehensive and detailed way, we used the following 'TN', 'FP', 'FN', 'TP', 'ACC', 'Precision', 'Recall', 'F1', 'ROC-AUC', 'PR-AUC', 'MCC', 'Weighted F1', 'TPR@FPR=0.0001', 'TPR@FPR=0.001', 'TPR@FPR=0.01', 'TPR@FPR=0.1' assessment metrics.

TP: The number of samples where the model predicts a positive class and the true value is also positive. TN: Number of samples where the model predicts a negative category and the true value is also negative. FP: Number of samples where the model predicts a positive class but the true value is negative. FN: Number of samples where the model predicts a negative class but the true value is positive.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{24}$$

$$Precision = \frac{TP}{TP + FP} \tag{25}$$

$$Recall = Sensitivity = TPR = \frac{TP}{TP + FN} \tag{26}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN} \tag{27}$$

ROC-AUC is the area under the ROC curve, which measures the model's ability to distinguish between positive and negative classes.

PR-AUC is the area under the PR curve, which measures the model's precision at different recall rates.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{28}$$

If the denominator is zero, the MCC is defined as zero.

$$Weighted\ F1 = \sum_{i=1}^{n} (w_i \times F1_i) \tag{29}$$

where n is the number of categories, $F1_i$ is the F1 score of the i^{th} category, and w_i is the weight of the i^{th} category in the real labeling

TPR@FPR=0.0001, TPR@FPR=0.001, TPR@FPR=0.01, TPR@FPR=0.1: These metrics represent the True Positive Rate (TPR) for a given False Positive Rate (FPR).

Environment and Parameter setting. The batch size during model training was 4, Adam optimizer (initial learning rate: 2e-5, weight decay: 5e-4), dropout rate 0.1, dataset n-fold cross-validation division random seed: 42 and 10 training epochs. We used PyTorch 1.12.1, NVIDIA CUDA12.0 and Python 3.8.20 for training on an NVIDIA 3090.

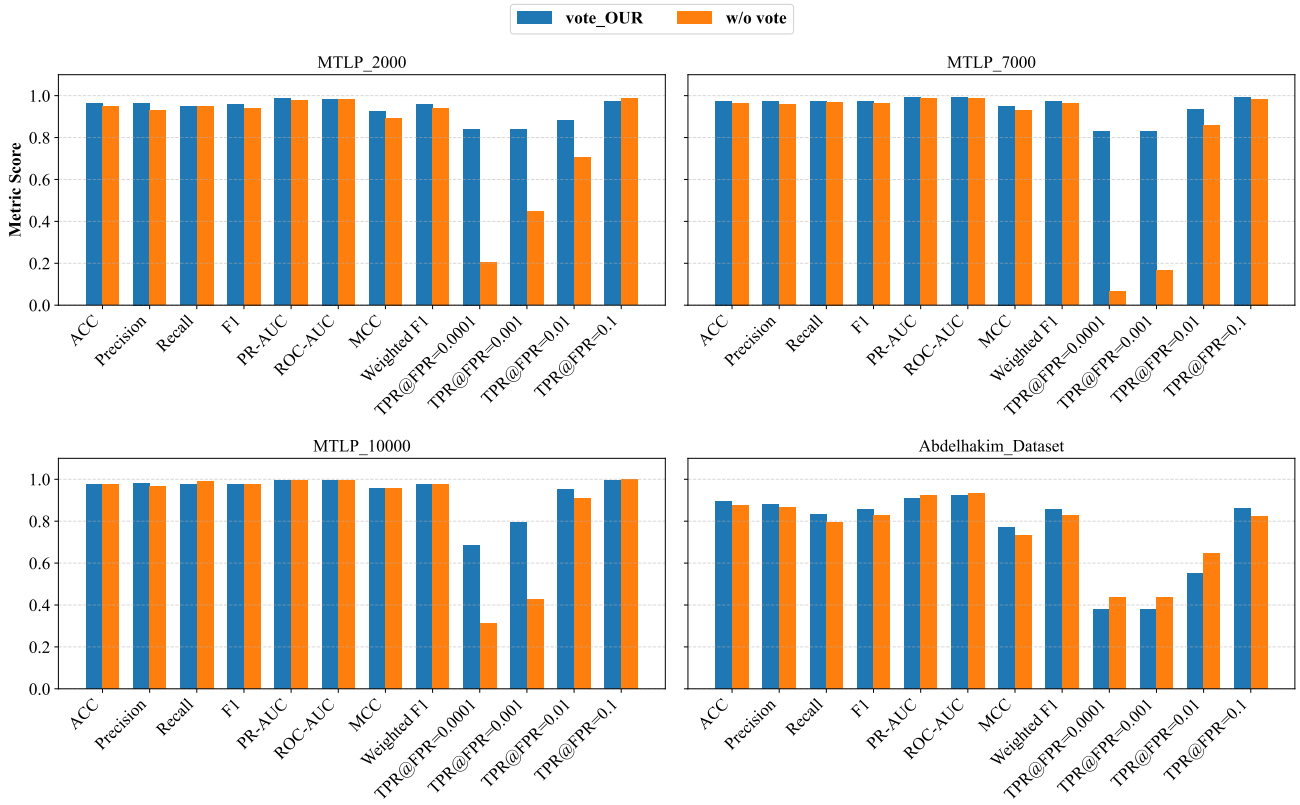

Fig. 4. The ablation experiments of our model against the voting mechanism were done on the MTLP dataset (uniformly sampled 2000, 7000, and 10000 data sizes) and the Abdelhakim dataset, respectively.

B. EXPERIMENTAL RESULTS

1) Phishing Detection Capabilities: In order to comprehensively evaluate the performance of our proposed model WebGuard++ in the field of phishing website detection, we conducted experiments on two publicly available datasets and compared it with several state-of-the-art models. The experimental results include results n the MTLP Dataset with 10,000 data points, (see Table I, III), results on Abdelhakim Dataset (see Table II, IV), and ROC graphs of different models on both datasets (Figure 3).

According to Table I, III and the ROC curve on the left side of Figure 3), our model on the MTLP Dataset with 10,000 data points, has key evaluation metrics such as Accuracy, Precision, F1, MCC, Weighted F1, TPR@FPR(0.0001), TPR@FPR(0.001), TPR@FPR(0.01), and so forth outperformed other models.

Our model achieves the highest values in both ACC and Precision, indicating that it exhibits higher accuracy and both accuracy and false alarm rate in categorizing phishing websites and normal websites. Meanwhile, the optimal values of F1 and MCC further validate the balance and robustness of our model in categorizing positive and negative samples, especially in the case of unbalanced category assignment. Among them, our model is significantly ahead of other models in low FPR scenarios such as TPR@FPR(0.0001), TPR@FPR(0.001), and TPR@FPR(0.01), which shows a unique advantage in the phishing detection task.

Comparing with other models (see Table I, III and Figure 3), left), traditional models such as BiLSTM and TextCNN have significantly lower performance, indicating their shortcomings in large-scale complex features; whereas advanced models such as PMANet and TransURL outperform our model in F1, MCC, and low FPR metrics, although their AUCs are close. This suggests that our model is not only able to accurately categorize but also provides strong detection capabilities in scenarios such as front and low false alarm rate.

According to Table II, IV and the ROC curve on the right side of Figure 3, our model significantly outperforms the other models in ROC-AUC on Abdelhakim Dataset and outperforms the other models in Accuracy, Recall, F1, Weighted F1, PR-AUC, MCC, TPR@FPR(0.0001), TPR@FPR(0.001), TPR@FPR(0.01), TPR@FPR(0.1) and many other key metrics outperform other models. Evidently, our model almost comprehensively overwhelms the other models.

The comparison with other models (see Table II, IV and Figure 3 right) shows that the traditional model BiLSTM and the base model TextCNN perform significantly worse on the dataset; whereas models such as PMANet [36] and URL-BERT [37] outperform in some of the metrics, they still do not perform as well as our model in terms of comprehensive performance and low FPR. In addition, some generative models

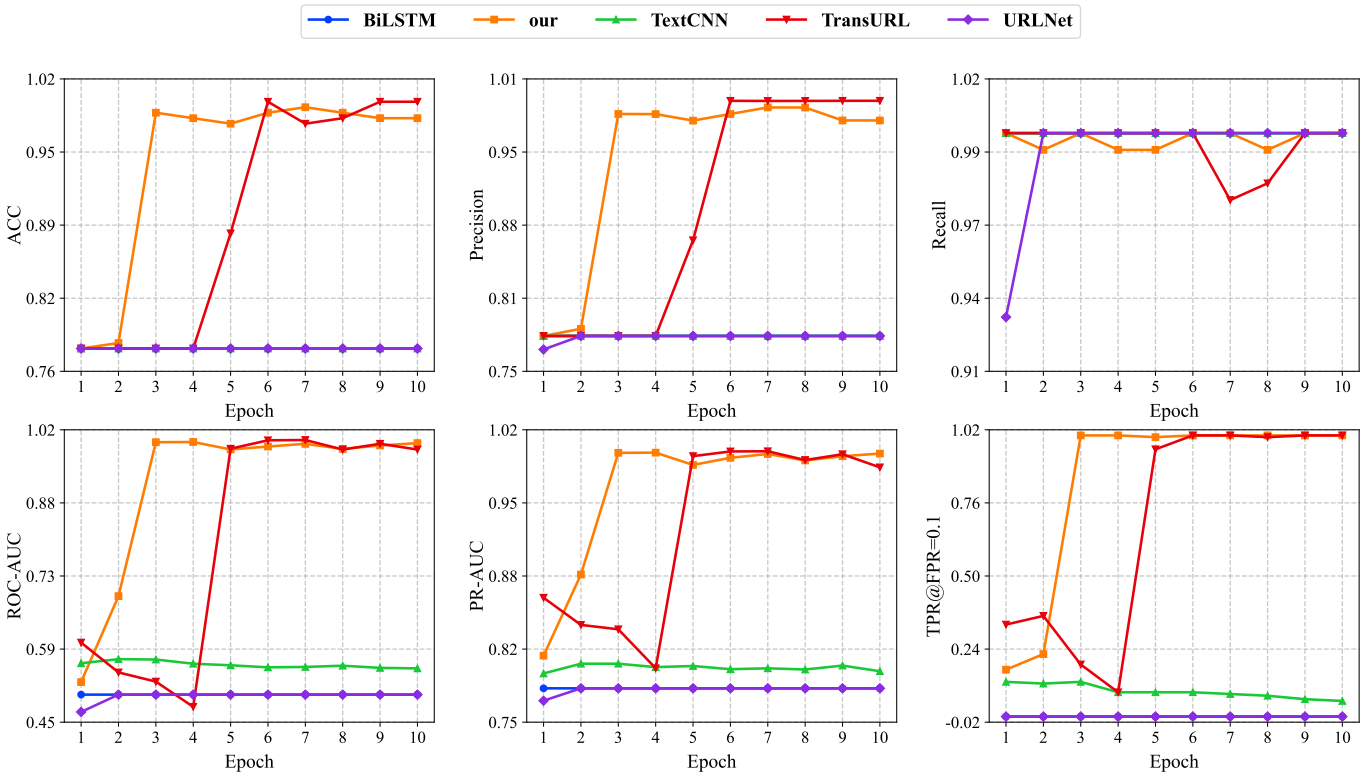

Fig. 5. The generalisation performance of BiLSTM, TextCNN, TransURL, URLNet, and the proposed model in this paper in cross-dataset scenarios is visually demonstrated by the trend changes of six classical evaluation metrics under 10 training epochs.

such as Semi-GAN [38] and dephides [39] perform relatively poorly on Abdelhakim Dataset, which further highlights the strong correlation and generalization ability of our model.

To summarize, our model shows excellent performance in the phishing website detection task, both in terms of classification accuracy, low false alarm rate detection capability, and consistency with unbalanced data, which meets the task requirements.

2) Voting Mechanism Ablation Experiment: In order to verify the actual enhancement effect of the proposed innovative mechanism on model performance, we compare and analyze the performance of adding the voting mechanism with and without adding the voting mechanism under several key evaluation metrics in four data (MTLP_2000, MTLP_7000, MTLP_10000, and Abdelhakim Dataset), as shown in Figure 4). From the overall trend, vote_OUR outperforms w/o vote on most of the assessment metrics, and performs better on most of the key metrics. This result indicates that the innovative mechanism has a significant positive effect in improving the overall recognition performance of the model and enhancing the robustness and generalization ability. In addition, We observed that after introducing this mechanism, the model's TPR significantly improved at all set FPR thresholds (0.0001, 0.001, 0.01, and 0.1), demonstrating comprehensive and stable detection advantages. This phenomenon indicates that the mechanism can effectively enhance the model's ability to identify positive samples even under extremely low false positive rate requirements, thereby improving the model's practicality and robustness in real-world high-risk scenarios. In summary, the experiments demonstrate that the proposed innovative method significantly improves the performance of the model under the key indicators while maintaining the overall performance balance, which validates its practical application value in high-reliability task scenarios.As shown in Figure 4.

3) Cross-dataset generalisability test: We conducted a generalization test on the cross-dataset course_data, and the results are shown in the Figure 5. The experimental results presented in the six subplots (a-f) demonstrate the performance of multiple models (BiLSTM, our model, TextCNN, TransTRL, and URLNet) over 10 epochs across six evaluation metrics: Accuracy, Recall, ROC-AUC, PR-AUC, TPR@FPR=0.1 and Precision. Below is a detailed analysis highlighting the advantages of our model compared to the other baselines.

(a) Accuracy. Our model achieves near-perfect accuracy ($\approx$ 1.0) after just 2 epochs, significantly outperforming all other models. TransURL also converges to high accuracy but requires 5 epochs, demonstrating slower convergence. TextCNN and BiLSTM plateau at lower accuracy values, while URLNet remains constant at a much lower level ($\sim$ 0.8). Our model demonstrates faster convergence and higher overall performance, indicating its robustness and efficiency in learning.

(b) Recall. All models except for URLNet achieve high recall (>0.99) by the end of training. However, our model

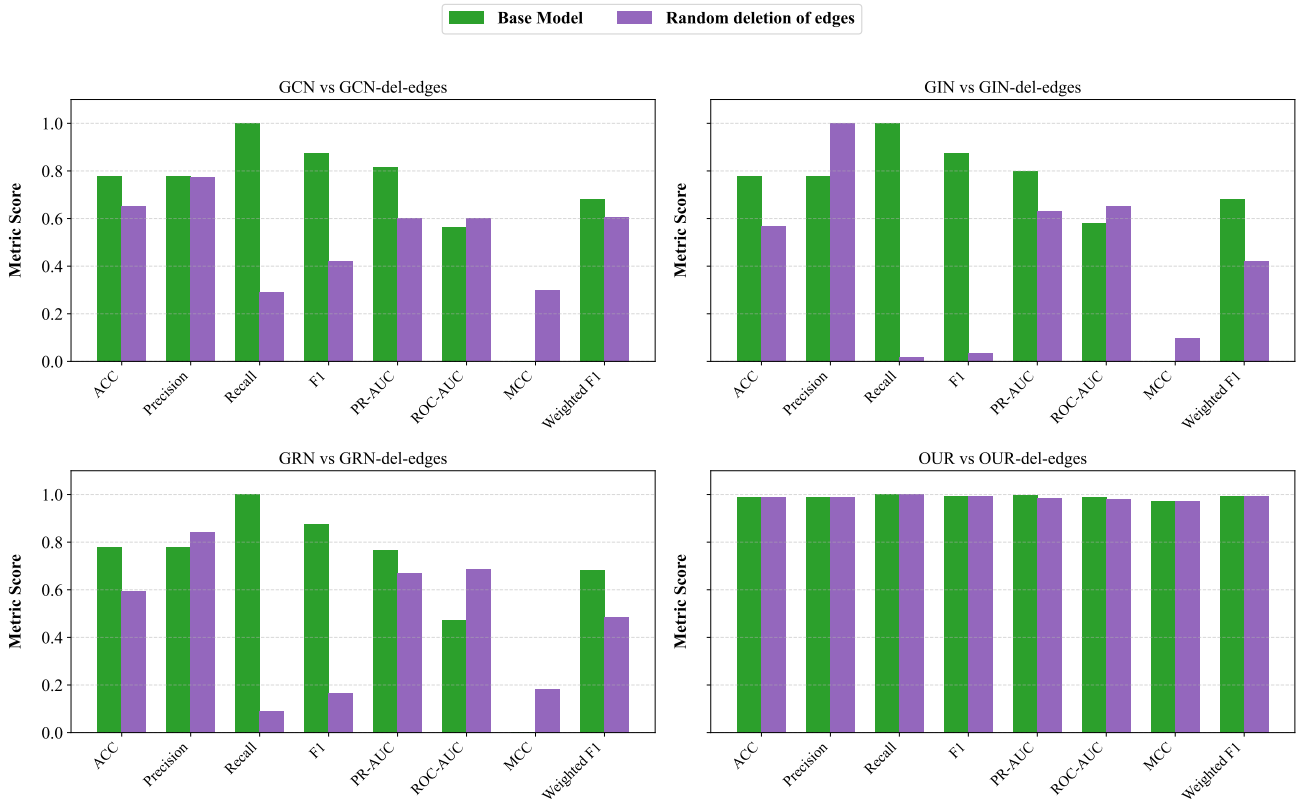

Fig. 6. Compare the robustness test of our model and the basic GNN model in the random edge deletion scenario.

maintains a consistently high recall throughout the epochs, with minimal fluctuations. TransURL shows instability during the training process, with notable drops around epochs 5-7. URLNet lags significantly behind with poor recall (~ 0.93). Our model ensures stability and reliability in recall, a critical metric for minimizing false negatives.

(c) ROC-AUC. Our model reaches a perfect ROC-AUC of 1.0 within just 2 epochs, outperforming all baselines. TransURL converges to a similar level but requires 5 epochs, while the other models (TextCNN, BiLSTM, URLNet) fail to surpass 0.75. URLNet performs the worst, stagnating near 0.5, indicating poor discriminatory capability. The superior ROC-AUC of our model highlights its exceptional ability to distinguish between classes.

(d) PR-AUC. Our model achieves a PR-AUC of 1.0 by epoch 2, outperforming all other models in both convergence speed and final performance. TransURL achieves similar results but requires more epochs (5), while TextCNN and BiLSTM stabilize at much lower levels (~ 0.8). URLNet shows the weakest performance, stabilizing at ~ 0.75. The rapid and consistent optimization of PR-AUC demonstrates our model's strength in handling imbalanced datasets by balancing precision and recall.

(e) TPR@FPR=0.1. Our model achieves a TPR@FPR=0.1 of 1.0 by epoch 2, significantly outperforming all baselines. TransURL shows delayed convergence, reaching similar performance only after 5 epochs. TextCNN and BiLSTM remain stagnant at lower values (~ 0.3), while URLNet fails to perform effectively, staying near 0.0. The ability of our model to achieve a perfect TPR at low FPR demonstrates its precision in detecting true positives under stringent conditions.

(f) Precision. Our model achieves precision values approaching 1.0 by epoch 2, maintaining stability throughout subsequent epochs. TransURL converges to similar precision levels but requires additional epochs (5). TextCNN and BiLSTM plateau at much lower precision levels (~ 0.8), while URLNet exhibits the poorest performance (~ 0.75). The high precision of our model emphasizes its ability to minimize false positives, an essential property in high-stakes applications.

The experimental results clearly establish the superiority of our model over others. Its faster convergence, higher overall accuracy, and consistent performance across all metrics make it a reliable and efficient solution for the task at hand. Compared to competing models, our approach demonstrates significant advancements in both learning efficiency and classification accuracy, solidifying its position as the state-of-the-art in the given application.

4) Robustness Testing: In the robustness evaluation, we applied random edge deletion with a probability of 50% to the input graph structures, aiming to assess the resilience of different models under structural perturbations. As illustrated in the bar chart in Figure 6, our proposed model demonstrates remarkable stability across various evaluation metrics (includ-

ing ACC, Precision, Recall, F1, PR-AUC, ROC-AUC, MCC, and Weighted F1) with negligible performance degradation and consistently high accuracy.

In contrast, baseline models such as GIN, GRN, and GCN exhibit substantial sensitivity to structural perturbations, with significantly larger performance drops, particularly in Recall and F1 scores. These results, as visualized in Figure 6, underscore the superior structural robustness and generalization ability of our model, which remains effective even when the graph structure is heavily disrupted.

V. Conclusion

In this paper, we propose a novel malicious URL detection framework, WebGuard++, consisting of a cross-scale URL semantic encoder, a subgraph-aware HTML encoder, and a bidirectional multi-view coupling module. The subgraph-aware model allows malicious region signals to aggregate with each other and is less likely to be diluted. At the same time, it also provides ideas for the interpretability of the model, which can be traced to a subregion when maliciousness is detected. In addition, the feature extraction and fusion for multimodal features enable the model to better understand the URL and HTML information and improve the detection performance. In this paper, after extensive experiments, we confirm that our model outperforms the benchmark methods as well as previously proposed state-of-the-art techniques on different data sizes and datasets, and maintains excellent robustness, generalization. Looking ahead, we will endeavor to improve the performance of "WebGuard++" and explore more new approaches.

Acknowledgment

This work was supported by the Basic Research Program (No.JCKY2023110C079).

References

[1] Ali Aljofey, Saifullahi Aminu Bello, Jian Lu, and Chen Xu. Comprehensive phishing detection: A multi-channel approach with variants tcn fusion leveraging url and html features. *Journal of Network and Computer Applications*, 238:104170, 2025.

[2] Zilaing Zhang, Jinmin Wu, Ning Lu, Wenbo Shi, and Zhiquan Liu. Adaptpud: An accurate url-based detection approach against tailored deceptive phishing websites. *Computer Networks*, page 111303, 2025.

[3] Ye Cao, Weili Han, and Yueran Le. Anti-phishing based on automated individual white-list. In *Proceedings of the 4th ACM Workshop on Digital Identity Management*, DIM '08, page 51–60, New York, NY, USA, 2008. Association for Computing Machinery.

[4] Nureni Ayofe Azeez, Sanjay Misra, Ihotu Agbo Margaret, Luis Fernandez-Sanz, and Shafi'i Muhammad Abdulhamid. Adopting automated whitelist approach for detecting phishing attacks. *Computers & Security*, 108:102328, 2021.

[5] Mahmood Moghimi and Ali Yazdian Varjani. New rule-based phishing detection method. *Expert Systems with Applications*, 53:231–242, 2016.

[6] Raniyah Wazirali, Rami Ahmad, and Ashraf Abdel-Karim Abu-Ein. Sustaining accurate detection of phishing urls using sdn and feature selection approaches. *Computer Networks*, 201:108591, 2021.

[7] Doyen Sahoo, Chenghao Liu, and Steven C. H. Hoi. Malicious url detection using machine learning: A survey, 2019.

[8] Anh Le, Athina Markopoulou, and Michalis Faloutsos. Phishdef: Url names say it all. In *2011 Proceedings IEEE INFOCOM*, pages 191–195. IEEE, 2011.

[9] Sadia Afroz and Rachel Greenstadt. Phishzoo: Detecting phishing websites by looking at them. In *2011 IEEE fifth international conference on semantic computing*, pages 368–375. IEEE, 2011.

[10] Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, and Banu Diri. Machine learning based phishing detection from urls. *Expert Systems with Applications*, 117:345–357, 2019.

[11] Shraddha Parekh, Dhwanil Parikh, Srushti Kotak, and Smita Sankhe. A new method for detection of phishing websites: Url detection. In *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 949–952, 2018.

[12] YunDa Tsai, Cayon Liow, Yin Sheng Siang, and Shou-De Lin. Toward more generalized malicious url detection models, 2024.

[13] Hung Le, Quang Pham, Doyen Sahoo, and Steven CH Hoi. Urlnet: Learning a url representation with deep learning for malicious url detection. *arXiv preprint arXiv:1802.03162*, 2018.

[14] Alsadig Hadi Alsadig and Md Oqail Ahmad. Phishing url detection using deep learning with cnn models. In *2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI)*, pages 768–775. IEEE, 2024.

[15] Ruitong Liu, Yanbin Wang, Zhenhao Guo, Haitao Xu, Zhan Qin, Wenrui Ma, and Fan Zhang. Transurl: Improving malicious url detection with multi-layer transformer encoding and multi-scale pyramid features. *Computer Networks*, 253:110707, 2024.

[16] Md Robiul Islam, Md Mahamodul Islam, Mst Suraiya Afrin, Anika Antara, Nujhat Tabassum, and Al Amin. Phishguard: A convolutional neural network-based model for detecting phishing urls with explainability analysis. In *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*, pages 1–6. IEEE, 2024.

[17] Yujie Li, Yanbin Wang, Haitao Xu, Zhenhao Guo, Fan Zhang, Ruitong Liu, and Wenrui Ma. Fed-urlbert: Client-side lightweight federated transformers for url threat analysis, 2023.

[18] Yanbin Wang, Weifan Zhu, Haitao Xu, Zhan Qin, Kui Ren, and Wenrui Ma. A large-scale pretrained deep model for phishing url detection. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[19] Yuexin Li, Chengyu Huang, Shumin Deng, Mei Lin Lock, Tri Cao, Nay Oo, Hoon Wei Lim, and Bryan Hooi. KnowPhish: Large language models meet multimodal knowledge graphs for enhancing Reference-Based phishing detection. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 793–810, Philadelphia, PA, August 2024. USENIX Association.

[20] Lei Zhang, Peng Zhang, Luchen Liu, and Jianlong Tan. Multiphish: Multi-modal features fusion networks for phishing detection. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3520–3524, 2021.

[21] S. Kavya and D. Sumathi. Multimodal and temporal graph fusion framework for advanced phishing website detection. *IEEE Access*, 13:74128–74146, 2025.

[22] Ruofan Liu, Yun Lin, Xiwen Teoh, Gongshen Liu, Zhiyong Huang, and Jin Song Dong. Less defined knowledge and more true alarms: Reference-based phishing detection without a pre-defined reference list. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 523–540, Philadelphia, PA, August 2024. USENIX Association.

[23] Haijun Zhang, Gang Liu, Tommy W. S. Chow, and Wenyin Liu. Textual and visual content-based anti-phishing: A bayesian approach. *IEEE Transactions on Neural Networks*, 22(10):1532–1546, 2011.

[24] Aron Laszka, Yevgeniy Vorobeychik, and Xenofon Koutsoukos. Optimal personalized filtering against spear-phishing attacks. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 958–964. AAAI Press, 2015.

[25] Yunji Liang, Qiushi Wang, Kang Xiong, Xiaolong Zheng, Zhiwen Yu, and Daniel Zeng. Robust detection of malicious urls with self-paced wide & deep learning. *IEEE Transactions on Dependable and Secure Computing*, 19(2):717–730, 2022.

[26] Marzieh Bitaab, Haehyun Cho, Adam Oest, Zhuoer Lyu, Wei Wang, Jorij Abraham, Ruoyu Wang, Tiffany Bao, Yan Shoshitaishvili, and Adam Doupé. Beyond phish: Toward detecting fraudulent e-commerce websites at scale. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2566–2583, 2023.

[27] Zhen Guo, Jin-Hee Cho, Ing-Ray Chen, Srijan Sengupta, Michin Hong, and Tanushree Mitra. Safer: Social capital-based friend recommendation to defend against phishing attacks. *Proceedings of the International AAAI Conference on Web and Social Media*, 16(1):241–252, May 2022.

[28] Jun-Ho Yoon, Seok-Jun Buu, and Hae-Jung Kim. Phishing webpage detection via multi-modal integration of html dom graphs and url features based on graph convolutional and transformer networks. *Electronics*, 13(16):3344, 2024.

[29] Jehyun Lee, Peiyuan Lim, Bryan Hooi, and Dinil Mon Divakaran. Multimodal large language models for phishing webpage detection and identification. *arXiv preprint arXiv:2408.05941*, 2024.

[30] Tri Cao, Chengyu Huang, Yuexin Li, Wang Huilin, Amy He, Nay Oo, and Bryan Hooi. Phishagent: a robust multimodal agent for phishing webpage detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27869–27877, 2025.

[31] Lihui Meng, Zhujuan Ma, and Erzhou Zhu. Phishing detection model integrating url characters and html word semantic deep features. In *2024 4th International Conference on Communication Technology and Information Technology (ICCTIT)*, pages 468–473, 2024.

[32] Bushra Sabir, M. Ali Babar, Raj Gaire, and Alsharif Abuadbba. Reliability and robustness analysis of machine learning based phishing url detectors. *IEEE Transactions on Dependable and Secure Computing*, pages 1–18, 2022.

[33] Xuefeng Hu, Zhihan Zhang, Zhenye Jiang, Syomantak Chaudhuri, Zhenheng Yang, and Ram Nevatia. Span: Spatial pyramid attention network for image manipulation localization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 312–328, Cham, 2020. Springer International Publishing.

[34] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 38–55, Cham, 2025. Springer Nature Switzerland.

[35] Furkan Çolhak, Mert İlhan Ecevit, and Hasan Dağ. Transfer learning for phishing detection: Screenshot-based website classification. In *2024 9th International Conference on Computer Science and Engineering (UBMK)*, pages 1–6, 2024.

[36] Ruitong Liu, Yanbin Wang, Haitao Xu, Zhan Qin, Fan Zhang, Yiwei Liu, and Zheng Cao. Pmanet: Malicious url detection via post-trained language model guided multi-level feature attention network. *Information Fusion*, 113:102638, 2025.

[37] Yujie Li, Yanbin Wang, Haitao Xu, Zhenhao Guo, Zheng Cao, and Lun Zhang. Urlbert:a contrastive and adversarial pre-trained model for url classification, 2024.

[38] Sharif Amit Kamran, Shamik Sengupta, and Alireza Tavakkoli. Semi-supervised conditional gan for simultaneous generation and detection of phishing urls: A game theoretic perspective. *arXiv preprint arXiv:2108.01852*, 2021.

[39] Ozgur Koray Sahingoz, Ebubekir BUBEr, and Emin Kugu. Dephides: Deep learning based phishing detection system. *IEEE Access*, 12:8052–8070, 2024.