applied sciences

MDPI

Article

MM-ConvBERT-LMS: Detecting Malicious Web Pages via Multi-Modal Learning and Pre-Trained Model

Xin Tong [1], Bo Jin [1,2,*], Jingya Wang [1], Ying Yang [2], Qiwei Suo [1] and Yong Wu [3]

[1] School of Information and Cyber Security, People's Public Security University of China, Beijing 100038, China
[2] The Third Research Institute of the Ministry of Public Security, Shanghai 200031, China
[3] Department of Information Management, The National Police University for Criminal Justice, Baoding 071000, China
* Correspondence: jinbo@gass.cn

Abstract: In recent years, the number of malicious web pages has increased dramatically, posing a great challenge to network security. While current machine learning-based detection methods have emerged as a promising alternative to traditional detection techniques. However, these methods are commonly based on single-modal features or simple stacking of classifiers built on various features. As a result, these techniques are not capable of effectively fusing features from different modalities, ultimately limiting the detection effectiveness. To address this limitation, we propose a malicious web page detection method based on multi-modal learning and pre-trained models. First, in the input stage, the raw URL and HTML tag sequences of web pages are used as input features. To help the subsequent model learn the relationship between the two modalities and avoid information confusion, modal-type encoding, and positional encoding are introduced. Next, a single-stream neural network based on the ConvBERT pre-trained model is used as the backbone classifier, and it learns the representation of multi-modal features through fine-tuning. For the output part of the model, a linear layer based on large margin softmax is applied to the decision-making. This activation function effectively increases the classification boundary and improves the robustness. In addition, a coarse-grained modal matching loss is added to the model optimization objective to assist the models in learning the cross-modal association features. Experimental results on synthetic datasets show that our proposed method outperforms traditional single-modal detection methods in general, and has advantages over baseline models in terms of accuracy and reliability.

Keywords: malicious web pages; multi-modal learning; pre-trained model; URL; HTML

Citation: Tong, X.; Jin, B.; Wang, J.; Yang, Y.; Suo, Q.; Wu, Y. MM-ConvBERT-LMS: Detecting Malicious Web Pages via Multi-Modal Learning and Pre-Trained Model. *Appl. Sci.* **2023**, *13*, 3327. https://doi.org/10.3390/app13053327

Academic Editor: Luis Javier García Villalba

Received: 19 January 2023
Revised: 23 February 2023
Accepted: 24 February 2023
Published: 6 March 2023

1. Introduction

The development of the Internet has undoubtedly revolutionized the way people work and live, with web browsers serving as the main gateway to the digital world. Unfortunately, the convenience of web services has also attracted cyber attackers who create phishing websites [1] for financial gain. While the specific motives and techniques of such attacks may vary, they all require an unsuspecting user to visit malicious web pages in order to achieve their objectives. In 2021, Rising [2] reported that their cloud security system intercepted a total of 62.79 million malicious websites worldwide, including 43.66 million Trojan-hanging websites and 19.13 million phishing websites. These malicious websites pose a serious threat to individuals' property and privacy security, making it imperative to develop effective approaches to identify them.

Although current machine learning or deep learning-based detection methods offer promising directions to address the limitations of traditional techniques. They often rely on single-modal features or simple stacking of classifiers based on different features. Consequently, it becomes challenging to semantic information from multiple views and may lead to information loss, which, in turn, would adversely affect the accuracy and

robustness of the methods. To solve these problems, this paper proposes a malicious web pages detection model MM-ConvBERT-LMS based on a pre-trained language model and multi-modal learning, with the following main contributions.

(1) **A malicious web page detection model based on a pre-trained model is constructed**. Our approach involves using ConvBERT as the backbone network to extract features from malicious websites. Compared to mainstream shallow learners, ConvBERT is able to leverage prior knowledge on a generic large-scale pre-trained dataset, leading to better training performance.

(2) **Multi-modal features are used to improve detection performance**. We use both the URL and HTML tag sequences of web pages as inputs and analyze the correlation of the two modal information using an attention-based single-stream neural network. Additionally, we introduce modal-type encoding to mitigate potential feature confusion.

(3) **A more robust classification layer is adopted**. Drawing inspiration from the large margin classifier, we replace the commonly used softmax in the output layer of our model with large margin softmax (LMS). This substitution enlarges the class margin between positive and negative samples, leading to improved model performance, particularly in cases where datasets are unbalanced or small-scale.

Experimental results on a multi-modal malicious web pages dataset synthesized from public datasets demonstrate that MM-ConvBERT-LMS achieves a detection accuracy of 98.72% and outperforms traditional single-modal detection techniques in metrics such as precision and F1 score, as well as outperforms current mainstream machine learning and deep learning models. Notably, our method does not rely on web content such as text as input, making it capable of detecting malicious web pages in multiple languages and more generalizable.

2. Related Work

Malicious web pages pose a great threat to the privacy and property security of users as they can steal private information without users' knowledge, often by disguising themselves as legitimate web pages or embedding malicious scripts in the pages. Various approaches have been proposed to identify such malicious web pages, including blacklist-based approaches [3], heuristic rule-based approaches [4], and interactive host behavior approaches [5]. Although these techniques have been widely used in anti-virus tools or browser security plug-ins, they can only detect known types of malicious web pages and rely on large-scale features or rule bases. As a result, they require high maintenance and update costs and are easily bypassed by encryption and obfuscation, leading to high rates of false positives and false negatives. To solve these problems, researchers have proposed self-learning techniques for malicious web page detection based on various machine learning or deep learning detection methods. These techniques have the potential to improve flexibility and accuracy, making them promising for real-world applications.

2.1. Detection Methods Based on Single-Modal Features

Uniform Resource Locator (URL) is a widely used representation for specifying the location of information on the World Wide Web. As shown in Figure 1, a complete URL consists of six parts, namely protocol, hostname, port, path, query, and anchor. URLs serve as interfaces for users to access various resources on the Internet. However, attackers may create phishing links that visually resemble the target URL, and then bind fake web pages generated using tools such as setoolkit to the URL, thereby tricking users into visiting them. Malicious URLs may have a high visual similarity to benign ones, but they may contain more spelling errors and digits that are semantically irrelevant. Therefore, some researchers have used URLs as a feature to detect malicious web pages.

Figure 1. The main structure of a typical URL.

Atrees [6] used AdaBoost to integrate SVM, Naive Bayesian algorithm, and decision tree model, and used lexical features obtained through CfsSubsetEval filtering as the input of the model, which can effectively detect four types of malicious URLs including spam, phishing, malware, and defacement URLs. Wang [7] proposed a malicious URL detection method based on the fusion of word-level features and character-level features. They used dynamic convolutional layers for feature extraction, which enables deeper feature mining from a larger perceptual view than the original convolutional layers. Furthermore, based on word-level and character-level features, Yuan [8] et al. used Bi-IndRNN and CapsNet as the backbone networks to achieve richer feature mining from different perspectives. Luo [9] et al. proposed a malicious URL detection method based on a composite neural network. Their approach involves using an auto-encoder to generate feature vectors, followed by a convolutional neural network that incorporates skip connections to classify the extracted features. The experimental results on the HTTP CSIC2010 dataset show that this approach performs well. To construct task-relevant word vectors, Yan [10] et al. proposed an unsupervised URL embedding model. Unlike traditional feature engineering, these pre-trained word vectors obtained from URL data can better help the downstream model complete the detection task.

While JavaScript provides good scalability for web pages, it also introduces security risks. Malicious code written in JavaScript can lead to attacks such as account theft and traffic hijacking. Therefore, several studies have attempted to distinguish malicious web pages from benign ones by analyzing the properties of JavaScript. Khan [11] et al. tested the effectiveness of four machine learning models including Naive Bayesian, KNN, SVM, and J48 decision tree for malicious JavaScript tasks and the results proved that KNN has the best detection effect on their dataset. Wang [12] et al. developed a malicious JavaScript detection framework based on stacked autoencoders and logistic regression algorithms. In this framework, stacked autoencoders are used to extract high-level semantic features from the input JavaScript code, and logistic regression is used for the classification task. The framework achieved a detection accuracy of up to 95%. Huang [13] et al. proposed a malicious JavaScript detection method called JSContana, which is based on a TextCNN model. In the data processing stage, the JavaScript code is transformed into a sequence of syntactic units. These units can then be used to generate dynamic word embeddings, which can effectively enhance the performance of downstream tasks. Alex [14] introduced S-BSA (Spider-based Bird Swarm Algorithm), a malicious JavaScript detection method that uses Deep Belief Network (DBN) with Spider Monkey Optimization (SMO) and Bird Swarm algorithm for classification. The method takes JavaScript code features such as execution time, break statements, and function calls as input and achieves detection accuracy of over 94% on public datasets. Fang [15] et al. proposed a detection method based on LSTM, which uses an improved word vector to extract features from bytecode and outperforms traditional models such as SVM and random forest in detection accuracy. In subsequent work [16], an attention mechanism was introduced to extract the significant features in syntactic unit sequences generated based on JavaScript abstract syntax tree, and the detection was further improved. JStrong [17] was the first to apply GNN models to malicious JavaScript detection tasks. Their method generates program dependency graphs based on abstract syntax tree, data flow, and control flow information, and then uses GNN for analysis. Compared to other neural networks, GNN is better at capturing the association relations between code fragments. The above studies focus on detection techniques on balanced datasets, whereas malicious JavaScript samples and normal samples in real-world scenarios are commonly imbalanced. To address this challenge, Phung [18] proposed a malicious JavaScript detection model Doc2Vec based on oversampling technique, which also has excellent performance on unbalanced datasets.

In addition, the HTML tags on web pages can serve as evidence for detection as they reflect the constituent elements of the pages. Hou [19] et al. developed a classifier by extracting a mixture of statistical features extracted from DHTML data. In their work, keyword frequencies of Native Java functions, frequency and length of HTML elements,

and count of the use of each ActiveX object were selected as input to the model, and experiments showed that the boosted decision tree performed the best among these feature sets, achieving an accuracy of 96.14%. Altay [20] extracted words, keyword frequencies, and keyword density from HTML content as features, and then used three supervised learning models, including SVM, maximum entropy, and extreme learning machine for classification, all of which demonstrated promising detection results.

2.2. Detection Methods Based on Hybrid Features

Several studies have shown that detecting malicious web pages based on multiple features leads to better results compared to single-feature approaches. For instance, Kazemian [21] et al. used URLs, page links, frequency of keywords contained in web content, and screenshots of web pages as features to verify the detection effectiveness of supervised and unsupervised learning. The classifier they built can be combined with the Google Chrome extension and applied to real scenarios. Wang [22] et al. proposed a two-stage detection method that combines static and dynamic features. In the first stage, the frequencies of elements contained in HTML, JavaScript, and URLs are used as static features, and then a decision tree model is used to classify them and label samples with confidence less than the threshold as "unknown". In the second stage, these unknown samples are extracted from the API call information by a dynamic analysis tool and then analyzed using a shellcode detector. Experiments show that the hybrid feature-based approach improves the F-score by 8.8% and 11.2%, respectively, compared to the static and dynamic feature-based approaches. However, not all elements in the hybrid features are significant for prediction. To filter out redundant features, Deng [23] used the feature selection method based on information gain, which was shown to remove redundant features. In addition, considering the effectiveness of malicious web detection models on computationally constrained devices, Amrutkar [24] proposed a lightweight mobile malicious web detection method called kAYO. This approach adds mobile-specific statistical features in addition to the frequency features contained in HTML, JavaScript, and URLs. The detection accuracy can reach 90% and has a fast-running speed.

2.3. Motivation

While there are many methods for detecting malicious web pages, a significant number of them are trained from scratch on labeled data, without utilizing any prior knowledge. These approaches can lead to a more challenging training process and results that are highly dependent on the size and quality of the dataset. Although some studies have attempted to use word2vec and fastText to extract semantic features, these static word vector methods struggle with issues, such as polysemy, making them less suitable for malicious web analysis tasks. Furthermore, most mainstream methods focus on feature engineering for single-modal features and do not consider mining for cross-modal features, which can result in additional processing costs, information loss, and insufficient learned representations. Therefore, to solve these problems, we propose a malicious web page detection method based on ConvBERT and multi-modal learning. Our approach improves upon existing work, as shown in Table 1.

Table 1. Comparison of the proposed method and the existing mainstream methods. ✓ represents that the technology is used, and × represents that the technology is not involved.

Methods	Models	Prior Knowledge	Features	Feature Engineering	Multimodal Matching	Analysis
Atrees [6]	AdaBoost	×	URL	✓	×	**Methods based on a single feature and statistical machine learning**: 1. They perform well on small-scale datasets and are able to respond quickly to meet real-time requirements. 2. They require feature engineering, which can lead to additional data preprocessing costs and information loss. 3. Due to their relatively small number of model parameters, these models can quickly reach convergence on large-scale datasets, thereby entering a plain stage in a shorter time frame. 4. They mainly learn statistical features, which may be less effective in mining high-level semantic features.
Khan [11]	KNN et al.	×	JS	✓	×	
Hou [19]	Decision Tree	×	HTML	✓	×	
Wang [7]	Dynamic Conv.	×	URL	×	×	**Deep learning models based on a single feature and supervised learning**: 1. It is proved that the representation ability of the deep learning models is stronger than the statistical machine learning models in malicious web page detection tasks. 2. Complex feature engineering is often not required. 3. Models' performance may be limited in small data set scenarios due to a lack of prior knowledge. 4. The single-modal and multi-view work represented by the research [7] is enlightening for the method based on multimodal learning.
Luo [9]	Composite NN	×	URL	✓	×	
Alex [14]	DBN	×	JS	✓	×	

Table 1. *Cont.*

Methods	Models	Prior Knowledge	Features	Feature Engineering	Multimodal Matching	Analysis
Yuan [8]	IndRNN + CapsNet	Word2Vec	URL	×	×	**Deep learning models based on a single feature and mixed-supervised learning**: 1. Pre-training with unsupervised learning can effectively leverage prior knowledge to improve the performance of these models on downstream tasks. 2. Static word embeddings generated by most methods struggle to handle polysemy. 3. JSContana [13] is inspired by the ELMo [25] model and employs a Bi-LSTM to transform word2vec into dynamic word embedding. However, the use of an LSTM-based backbone limits parallel computing, which negatively impacts training efficiency. 4. Most methods lack an attention mechanism that can assist the models in extracting more relevant features while filtering out noise.
Yan [10]	URL embed	×	URL	×	×	
Wang [12]	SAE + LR	Layer-wise Pre-training	JS	✓	×	
JSContana [13]	TextCNN	Word2Vec	JS	✓	×	
Fang [15]	LSTM	Word2Vec	JS	✓	×	
Fang [16]	Att-LSTM	fastText	JS	✓	×	
JStrong [17]	GNN	Word2Vec	JS	✓	×	
Doc2Vec [18]	SVM	Doc2Vec	JS	✓	×	
Altay [20]	SVM et al.	×	HTML	✓	×	**Statistical machine learning models based on mixed features**: 1. It has been demonstrated that the integration of these hybrid features can significantly improve detection performance. 2. These approaches primarily concentrate on feature extraction. However, the complexity of the data can escalate the difficulty and cost of feature engineering, potentially leading to the curse of dimensionality. 3. Statistical machine learning models face inherent challenges in capturing correlations between different features, especially in the context of hybrid features. Furthermore, these models often lack auxiliary optimization objectives to facilitate the mining of information across modalities.
Kazemian [21]	SVM et al.	×	Page Link, Screenshot, URL, Keywords	✓	×	
Wang [22]	Decision Tree	×	URL, HTML, JS	✓	×	
Deng [23]	Rotation Forest	×	Content, JS, URL	✓	×	
kAYO [24]	SVM et al.	×	URL, HTML, JS, Statistical features	✓	×	

Table 1. *Cont.*

Methods	Models	Prior Knowledge	Features	Feature Engineering	Multimodal Matching	Analysis
Ours	ConvBERT	Dynamic Embed.	URL + HTML	×	✓	**A method based on multimodal learning and pre-trained model**: 1. This method accepts the raw URL string and HTML tag sequence directly as input and does not require any feature engineering. 2. To learn dynamic text embeddings using pre-training based on prior knowledge, ConvBERT was chosen as the backbone. 3. The model incorporates modal matching as an auxiliary task to enable cross-modal information learning. 4. The Transformer-based architecture is more suitable for parallel computing than sequential models such as LSTM. 5. The self-attention mechanism is used to better focus on meaningful features.

3. Method

3.1. Overall Structure

The term "modal" refers to a specific type of information and its representation, and the Internet is a carrier of multi-modal data. For instance, a typical web page contains various modal data, including the URL, text content, multimedia data such as images and audio contained on the web page, and the sequence of HTML tags. These distinct forms of information may be semantically connected, while data from different modalities may also provide complementary information to each other, revealing implicit features that are not apparent in single-modal data.

Several multi-modal deep learning models [26,27] have been developed to address the challenges of processing multi-modal data. By incorporating multiple types of data as input and capturing correlations between modalities, these models have been widely used in various fields such as sentiment analysis [28] and fake news detection [29]. Building upon this existing research, we present a novel multi-modal learning model for detecting malicious web pages. Our objective is to exploit the distinct features of different types of web page data to elevate detection accuracy and expand the range of possible applications for our model.

Given that the multi-modal data we use is essentially string-based (HTML tag sequences and URLs), despite representing information in different dimensions, we chose the ConvBERT [30] rather than the visual-text models as the backbone. The model has also been enhanced to improve its applicability to the task of analyzing malicious web pages. The overall framework, shown in Figure 2, consists of three key components: dual input, feature analysis network, and multi-task optimized output layer. During the input phase, the model accepts URL strings and HTML tag sequences as input, and a modal-type encoding is used to help the model learn the boundary between the two types of data. In the feature analysis phase, a single stream backbone network built on ConvBERT is selected for the feature interaction task of the two modalities. In the output stage, a linear layer based on LMS is utilized to carry out decision classification. For the optimization objective, in addition to detecting web page types as the main task, a coarse-grained modal matching loss is introduced as an optimization objective to further assist the model in learning cross-modal correlated features.

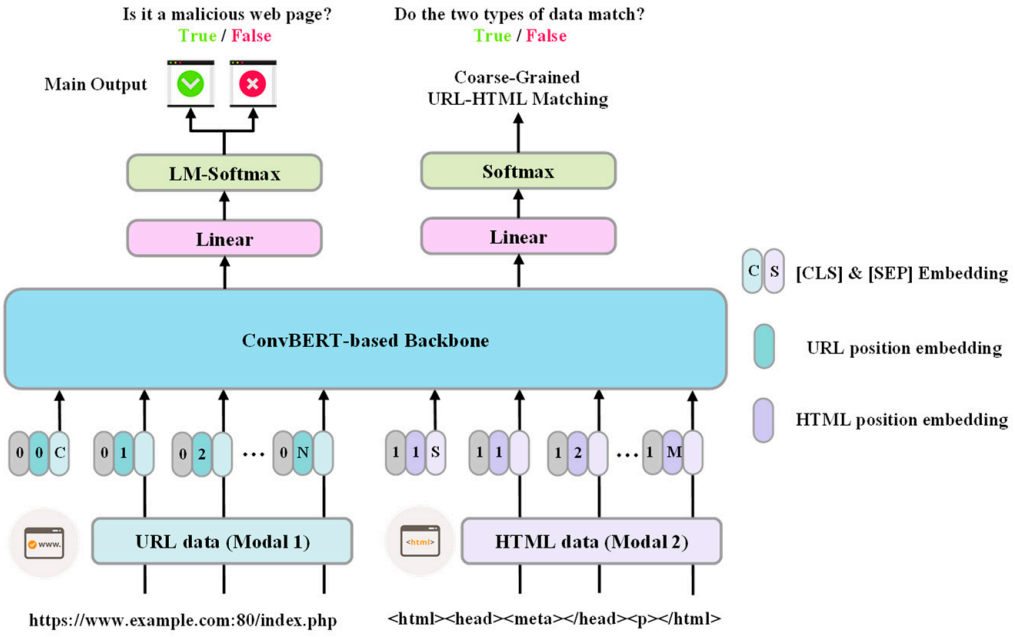

Figure 2. The overall structure of the MM-CovBERT-LSM model.

3.2. Multi-Modal Input and Embedding

Since we use a single-stream neural network based on the Transformer architecture [31] to analyze input data from two modalities, HTML tag sequences, and URL strings, additional processing is required at the input stage to help the model distinguish between two modalities and learn sequential features.

First, the order of elements in the input data reflects their features. For example, the ordered combinations of letters in a URL can form words with certain semantics, and the order of HTML tags can describe the overall structure of the page. Since the Transformer is mainly composed of self-attention mechanisms and feed-forward layers, it cannot inherently represent sequential features, such as recurrent neural networks. To avoid losing important features, we use positional encoding in the input phase to add sequential features to word vectors.

Second, ConvBERT, the backbone of our model, was originally designed to process single-modal text data. Concatenating HTML and URL data and feeding it directly to the ConvBERT would lead to feature confusion, which in turn affects the detection effect. Therefore, we also introduce modal-type encoding in the input stage. Specifically, we mark HTML data as 0 and URL data as 1 and add a special token [SEP] as a segmentation marker between the two types of data.

Ultimately, the input process of this model is shown in Equation (1), where h and u denote the HTML and URL data, respectively. H and U represent the modal-type encoding of both data, P represents positional encoding, and the special token [CLS] represents the global features of the input sequence.

$$\begin{aligned} \overline{h} &= [[\text{CLS}]; h_1; \cdots; h_N] + H^{type} \\ \overline{u} &= [[\text{SEP}]; u_1; \cdots; u_M] + U^{type} \\ x &= \text{Concat}(\overline{h}, \overline{u}) + P \end{aligned} \tag{1}$$

3.3. ConvBERT-Based Backbone Network

To incorporate prior knowledge into the model, we use a pre-trained ConvBERT as the backbone. ConvBERT is an improved BERT-based [32] pre-trained model, which has been optimized in both pre-training methods and model structure to achieve a balance of effectiveness and efficiency.

Regarding the pre-training approach, ConvBERT uses a similar training process as Electra [33], both of which refer to the generative adversarial network and use Replaced Token Detection as a pre-training task, the main process of which is depicted in Figure 3.

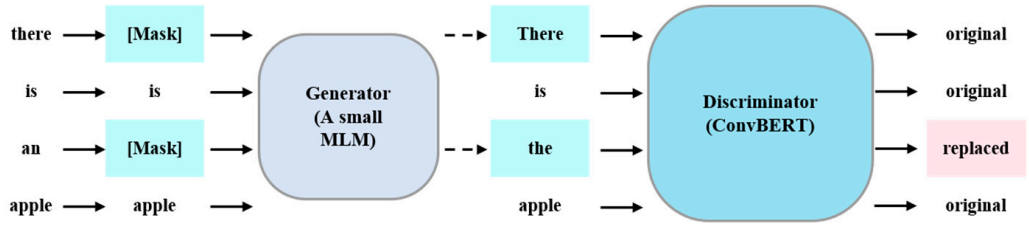

Figure 3. The pre-training process of ConvBERT.

During the pre-training process, both the generator and discriminator are trained simultaneously. The generator is a small masked language model (MLM) whose main goal is to reconstruct the masked tokens in the input, whereas the discriminator is a classifier that operates at the token level to identify whether each word in a sentence has been replaced or not. The pre-training objective of the ConvBERT model is shown in Equation (2), where G represents the generator, D represents the discriminator, θ represents the model parameters, and t is the position in the input vector marked by the token [MASK]. x^{masked} represents the sequence after masking and $x^{corrupt}$ is the input reconstructed by the generator. By conducting large-scale corpus-based pre-training, the model can become more sensitive to

some elements in URL or HTML data that have a specific meaning. This prior knowledge can help improve the accuracy of detection.

$$\begin{aligned}\mathcal{L}_{pt} &= \min_{\theta_G \theta_D} \sum \mathcal{L}_{MLM}(x, \theta_G) + \lambda \times \mathcal{L}_{Disc}(x, \theta_D) \\ &= \mathbb{E}\left(\sum_i -\log pG(x_i \Big| x^{masked})\right) \\ &+\lambda \times \mathbb{E}\left(\sum_t -\mathbb{I}(x_t^{corrupt} = x_t) \log D(x^{corrupt}, t) - \mathbb{I}(x_t^{corrupt} \neq x_t) \log(1 - D(x^{corrupt}, t))\right)\end{aligned} \tag{2}$$

From a structural perspective, ConvBERT makes significant improvements to the multi-head self-attention mechanism included in the Transformer architecture. This mechanism, which is a core part of the Transformer, is shown in Equation (3), where Q is the query matrix, and K and V correspond to the key matrix and the value matrix, respectively. Through the use of multi-head attention, ConvBERT is able to learn features across multiple representation subspaces, providing a more comprehensive understanding of the input data. As shown in Equation (4), where *head* represents the input vector projected into the subspace, W^O is the parameter matrix used for projection, and d_k is the dimension of the vector K. The multi-headed self-attention has an expansive perceptual field of view that covers the entire output. This allows it to effectively extract global features, enabling analysis of not only long-term relationships in HTML sequences but also for modeling cross-modal relationships.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{3}$$

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{Concat}(head_1, head_2, \ldots, head_n)W^O \\ \text{where } head_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)\end{aligned} \tag{4}$$

However, for URL data, its local features are more useful for the detection task. For example, adjacent elements can form words with independent semantics, which in turn reflect the characteristics of the corresponding web page. Lightweight convolution is shown to compensate for the lack of local information modeling by multi-headed self-attentiveness, which works as shown in Equation (5), where X is the input vector, W represents the weights of the convolution kernel, i represents the position of the feature map currently being processed, and K represents the size of the convolution kernel. The improved dynamic convolution uses variable convolution kernel parameters and thus enhances the diversity of the representation. As shown in Equation (6), W_f represents the parameter matrix of the linear layer.

$$\text{LConv}(X, W, i) = \sum_{j=1}^{K} W_j \bullet X_{(i+j-\lceil \frac{K+1}{2} \rceil)} \tag{5}$$

$$\text{DConv}(X, W_f, i) = \text{LConv}(X, \text{softmax}(W_f X_i), i) \tag{6}$$

Drawing inspiration from both convolutional and self-attention mechanisms, ConvBERT finally uses span-based dynamic convolution to efficiently learn local features. This is achieved through the calculation method shown in Equation (7), K_S is the span-aware generated by the depth-wise separable convolution, and $\odot$ is the point-wise multiplication. The structure comparison of the self-attention mechanism, dynamic convolution, and span-based dynamic convolution is shown in Figure 4.

$$\text{SDConv}(Q, K_s, V; W_f, i) = \text{LConv}(V, \text{softmax}(W_f(Q \odot K_s)), i) \tag{7}$$

To represent different levels of features, ConvBERT incorporates a mixed attention module that combines multi-head self-attention and span-based dynamic convolution. The underlying principles of this module are depicted in Figure 5 and the mathematical formulation is presented in Equation (8). As such, this model is better suited for detecting malicious web pages across multiple modalities, as it can effectively learn features at both local and global levels.

$$\text{MixedAttention}(K, Q, K_s, V; W_f) = \text{Concat}(\text{SelfAttention}(Q, K, V), \text{SDConv}(Q, K_s, V; W_f) \tag{8}$$

Moreover, to further reduce the training costs, ConvBERT also modifies the feed-forward network with more parameters and proposes a grouped feed-forward network. Specifically, the grouped linear (GL) operator is introduced, as shown in Equation (9). H is the input vector of dimension d, g represents the number of groupings, m is the dimension of the temporary variables M and M', and $\text{Linear}_{a \to b}(\bullet)$ denotes a fully connected network with a-dimensional input and b-dimensional output. This is achieved by feature grouping to improve efficiency and learning ability

without affecting the model performance. Finally, the mixed attention and grouping feed-forward networks are stacked several times to form the ConvBERT model.

$$\begin{cases} M = \text{Concat}_{i=0}^{g}\left(\text{Linear}_{\frac{d}{g}\to\frac{m}{g}}^{i}(H_{[:,i-1:i\times\frac{d}{g}]})\right) \\ M' = \text{GeLU}(M) \\ H' = \text{Concat}_{i=0}^{g}\left(\text{Linear}_{\frac{m}{g}\to\frac{d}{g}}^{i}(M'_{[:,i-1:i\times\frac{m}{g}]})\right) \end{cases} \tag{9}$$

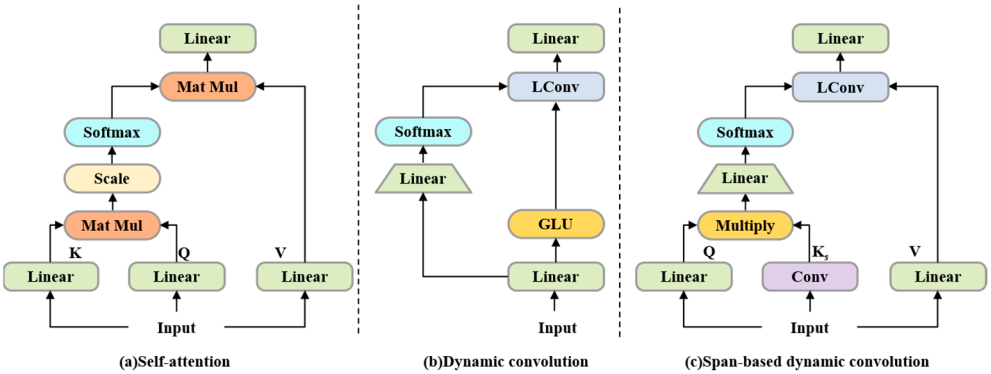

Figure 4. Self-attention mechanism, dynamic convolution, and span-based dynamic convolution module.

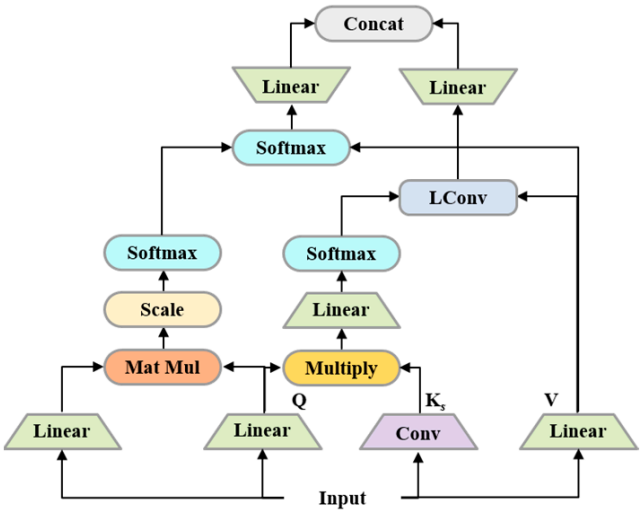

Figure 5. Mixed attention module based on multi-head self-attention and span-based dynamic convolution.

3.4. Multi-Task Optimization Objective

For the output section of the model, an optimization objective based on multi-task learning is used. Specifically, we use a linear layer as the main output, whose aim is to determine whether the input corresponds to a malicious web page. To increase the classification spacing between positive and negative samples, we employ an efficient large margin softmax (LMS) [34] activation function in place of the original softmax. The classification loss based on LMS is shown in Equation (10), where C is the total number of categories, l is the true label, x is the input and f denotes the logits value of the output.

$$\mathcal{L}(x,l) = -\log\left(\frac{\exp(f_l)}{\sum_{c=1}^{C}\exp(f_c)}\right) + \frac{\alpha}{2}\sum_{c\neq l}\left(\frac{\exp(f_c)}{\sum_{c*\neq l}\exp(f_{c*})} - \frac{1}{C-1}\right)\log\left(\frac{\exp(f_c)}{\sum_{c*\neq l}\exp(f_{c*})}\right) \tag{10}$$

In addition, a succinct auxiliary output is added to learn associative features across modalities at a coarse-grained level. To achieve this, we introduce a label that signifies modal matching, with samples in the original dataset labeled as 1 if their URLs and HTML belong to the same category. We randomly select URL and HTML data from distinct categories to create new samples and set their corresponding label values to 0. During training, a binary classification task is included to differentiate whether the two types of data belong to the same category, thus achieving coarse-grained modal matching.

Finally, the overall optimization objective of the model is shown in Equation (11). Where y represents the true label, y^* represents the predicted output of the model, and the tasks of the model are denoted by Main and UHM, referring to the main task and the URL-HTML modal matching task, respectively. A balanced hyperparameter λ is used to adjust the importance of the two losses. Consider that the main output of the model is meaningless for the additional data added in the modal matching task. Therefore, the main loss is multiplied by factor y_{UHM}, which prevents the added data from interfering with the main task during training. At this time, since y_{UHM} equals 0, the gradient is not backpropagated.

$$\mathcal{L} = y_{UHM} \times \mathcal{L}_{Main}(y_{Main}, y^*_{Main}) + \lambda \times \mathcal{L}_{UHM}(y_{UHM}, y^*_{UHM}) \tag{11}$$

4. Experiment and Analysis

4.1. Experimental Datasets and Metrics

In order to validate the effectiveness of MM-ConvBERT-LMS in the malicious web page detection task, a synthetic malicious web page dataset based on the public dataset [35,36] is used in this paper, which contains features of both HTML and URL modalities. The length distribution of the two data is shown in Figure 6. In the experiments, 10,000 data are used as the training set, 5000 data are used as the validation set, and 10,000 data are used as the test set, and the distribution of positive and negative samples is balanced. Remarkably, we find that the HTML features extracted directly from the original dataset suffer from severe data duplication problems between the training, validation, and test sets, and one possible reason is that the HTML tag sequences of some of the simpler web pages are likely to be the same. To ensure the fairness of the experiments and to avoid the data pollution problem, we additionally de-duplicated the HTML tag sequence data, even though this duplicated data came from different original web pages. The hardware configuration and software environment of the server used for the experiments are shown in Table 2.

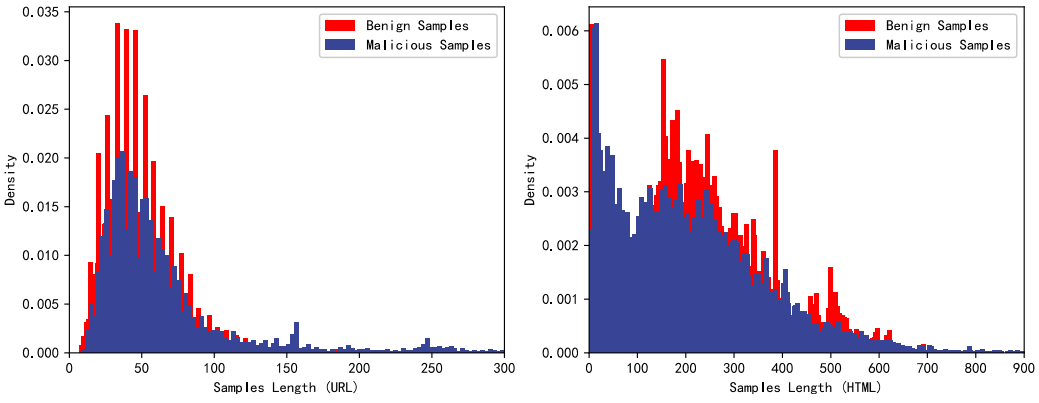

Figure 6. The distribution of the two-modal data.

Table 2. Hardware configuration and experimental environment.

Device & Software	Information
CPU	Intel(R) Xeon(R) CPU E5-2690 v4
RAM	12 GB
External Storage	512 GB SSD
Operating System	Ubuntu 18.04.3 LTS
Python Version	3.8.8 (AMD64)
Machine Learning Library	Pytorch 1.11.0

For the evaluation metrics, Accuracy, Precision, Recall, and F1 Score were chosen to evaluate the performance of the model, and the calculation procedure is shown in Equations (12)–(15). Where *TP* and *TN* represent the number of true positive samples and true negative cases, *FP* and *FN* represent the number of false positive samples and false negative samples. Figure 7 provides a graphical representation of these metrics, with *TP* and *TN* indicating correctly classified samples, and *FP* and *FN* representing misclassified samples.

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{12}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

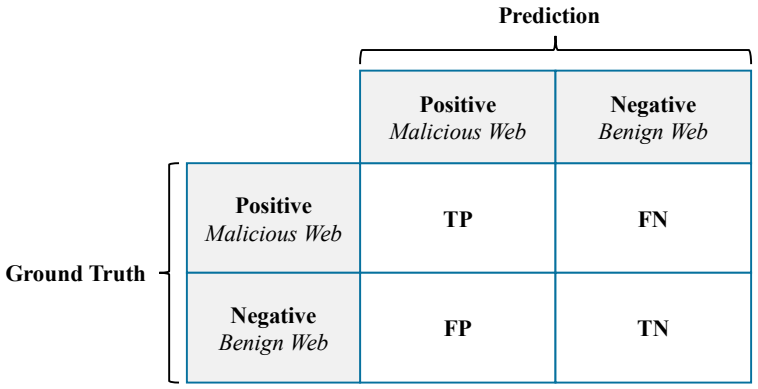

Figure 7. In our experiment, the meanings of TP, TN, FP, and FN.

To better train the model, we use the AdamW optimizer and use a warm-up mechanism during the training of the pre-trained models. The number of training epochs was set to 10 for the pre-trained models and made adjustments to a maximum of 20 epochs for the shallow deep learning models.

4.2. Comparative Experiment

Methods from both statistical machine learning and deep neural network categories were selected as the baselines for the experiments. For statistical machine learning methods, Naive Bayesian (NB), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN) were selected as comparison models. For deep learning methods, in addition to comparing with commonly used Text Convolutional Neural Network (TextCNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Deep Pyramid Convolutional Neural Network (DPCNN), we also selected several pre-trained models as a baseline, including BERT, ALBERT [37], and the original ConvBERT. In our experiments, we evaluated the performance of each model on three datasets: pure HTML data, pure URL data, and multi-modal dataset (HTML + URL). The experimental results are presented in Tables 3–5, respectively. The optimal results are marked in black bold.

According to the results, deep learning methods clearly outperform statistical machine learning model approaches, with several deep neural networks achieving accuracy rates of more than 90%. Meanwhile, among all the neural network models, the pre-trained methods outperform classical neural network models trained from scratch, demonstrating the benefits of using dynamic word vectors as prior knowledge. The MM-ConvBERT-LMS model even achieved the best results across all three experiments, achieving 98.72% accuracy on the task of detecting malicious web pages based on multi-modal features, and also outperforming the baselines in terms of precision, recall, and F1 score.

Table 3. Experimental results based on URL data.

Methods	Models	Accuracy	Precision	Recall	F1 Score
Machine Learning	NB	88.72	97.52	79.64	87.68
	SVM	89.63	95.27	83.57	89.04
	KNN	84.11	92.28	74.72	82.58
	DT	89.41	93.51	84.88	88.99
	RF	91.37	94.62	87.88	91.12
Deep Learning	Bi-LSTM	92.49	94.97	89.86	92.34
	Bi-GRU	92.18	93.50	90.79	92.13
	TextCNN	92.70	94.93	90.34	92.58
	DPCNN	92.38	94.10	90.56	92.30
	ALBERT	95.70	95.84	95.62	95.73
	BERT	96.95	97.08	96.87	96.97
	ConvBERT	97.63	97.66	**97.64**	97.65
	ConvBERT-LMS(ours)	**97.76**	**97.93**	97.62	**97.77**

Table 4. Experimental results based on HTML data.

Methods	Models	Accuracy	Precision	Recall	F1 Score
Machine Learning	SVM	76.39	77.56	74.80	76.15
	KNN	74.35	77.49	69.21	73.12
	DT	77.83	79.10	76.13	77.59
	RF	84.24	82.61	87.06	84.78
Deep Learning	Bi-LSTM	75.67	75.94	75.71	75.83
	Bi-GRU	76.68	79.67	72.14	75.72
	TextCNN	85.68	83.78	88.77	86.20
	DPCNN	83.89	85.64	81.75	83.65
	ALBERT	85.65	86.25	85.10	85.67
	BERT	86.45	86.81	86.21	86.51
	ConvBERT	87.13	**87.02**	87.52	87.27
	ConvBERT-LMS(ours)	**87.45**	86.71	**88.69**	**87.69**

Table 5. Experimental results based on multi-modal data.

Methods	Models	Accuracy	Precision	Recall	F1 Score
Machine Learning	NB	89.91	97.21	82.34	89.16
	SVM	91.15	95.55	86.47	90.78
	KNN	83.99	91.03	75.69	82.66
	DT	83.25	84.40	81.90	83.13
	RF	91.03	92.23	89.76	90.98
Deep Learning	Bi-LSTM	93.60	96.73	90.36	93.43
	Bi-GRU	93.75	96.37	91.03	93.62
	TextCNN	95.02	96.38	93.63	94.99
	DPCNN	93.89	94.73	93.06	93.88
	ALBERT *	96.93	97.06	96.85	96.95
	BERT	97.47	97.65	97.32	97.49
	ConvBERT	98.06	98.29	97.86	98.07
	MM-ConvBERT-LMS(ours)	**98.72**	**98.90**	**98.55**	**98.73**

* means no modal-type encoding is used, and the accuracy of the ALBERT model is only 86% after using modal-type encoding.

Upon further comparison of the three sets of experiments, it can be seen that the models perform better on the URL dataset than on the HTML tag sequence. For example, the Naive Bayesian method trained on pure HTML data fails to converge even when its F1 score is below 50%, thus we do not record this meaningless result in the Table 4. The reason for this phenomenon may be that the difference between positive and negative samples in URL data is more obvious: the body of positive

samples tends to contain meaningful, regular words, while negative samples tend to contain numbers more frequently, and strings in the body part rarely represent complete semantic information.

To precisely observe the improvement in the effectiveness of the multi-modal model compared to the single-modal model, we plotted the performance gain of each model on different types of features, as shown in Figures 8 and 9. The results demonstrate a clear improvement in accuracy when using multi-modal features, with an average increase of 0.44% and 11.29% compared to models based on URL and HTML data, respectively. With the exception of random forest and decision tree models, which exhibit a performance decrease after incorporating multi-modal features, all other models can aggregate and analyze information from different perspectives, leading to meaningful representations and reliable decision-making.

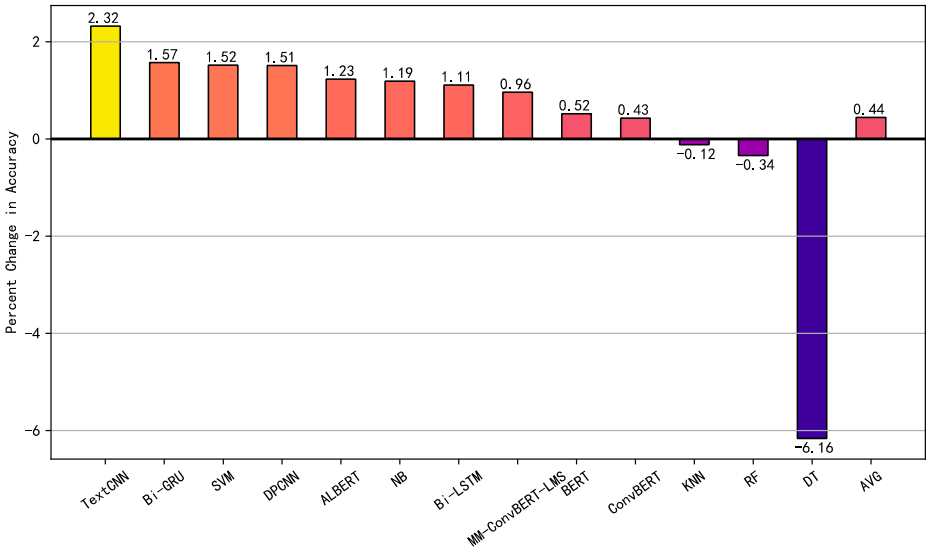

Figure 8. Accuracy improvement of multi-modal models compared to pure URL-based trained models.

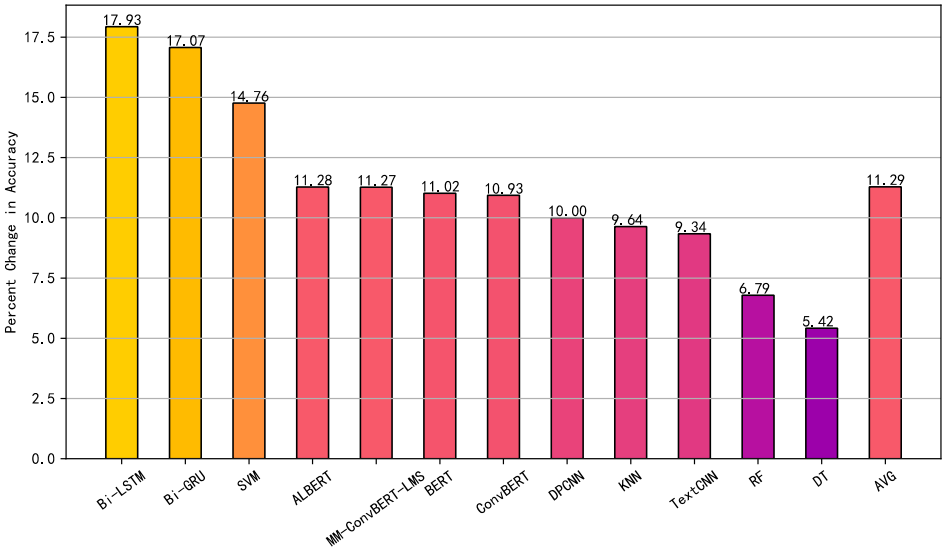

Figure 9. Accuracy improvement of multi-modal models compared to pure HTML-based trained models.

To observe the reliability of classification for each model, we select the SVM, CNN, original ConvBERT, and MM-ConvBERT-LMS models, which exhibit better performance and greater variability in principle, as representatives and visualize their ROC and PR curves, as shown in Figure 10. The AUC scores of the MM-ConvBERT-LMS model outperform all three baselines, which proves that the model has better classification ability and robustness. Moreover, we plotted the ROC and PR curves of the models trained based on different data, as shown in Figure 11. The multi-modal model displayed superior performance compared to the single-modal model in both ROC and PR curves, reaffirming the effectiveness of our proposed approach.

While the above experiments were conducted with a sufficient number of training samples, labeled data in real-world scenarios is often extremely limited. As a result, we also explored the impact of the size of the training set on the model's effectiveness. The results, presented in Figure 12, show a decreasing trend in accuracy as the size of the training set is reduced. However, even when the training set is only 5% of the test set size, the MM-ConvBERT-LMS model outperforms the compared models and the single-modal model, with an accuracy exceeding 93%. In contrast, the TextCNN model tends to overfit more easily than other models and its accuracy is even lower than that of the SVM model. These results suggest that the use of multi-modal models can be beneficial for scenarios with limited labeled data.

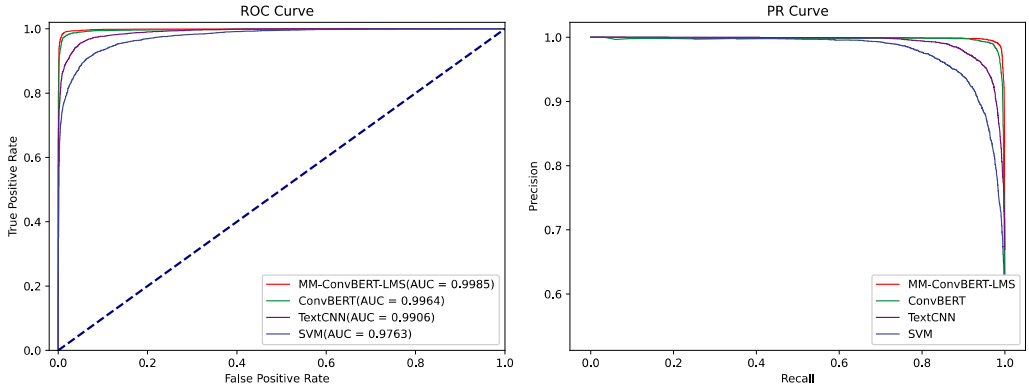

Figure 10. ROC and PR curves to compare the performance of different multi-modal models.

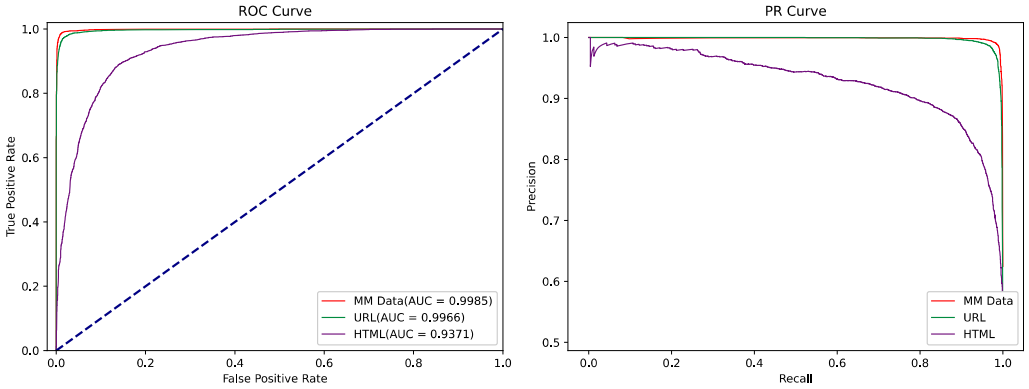

Figure 11. ROC and PR curves to compare the performance of single-modal and multi-modal ConvBERT models.

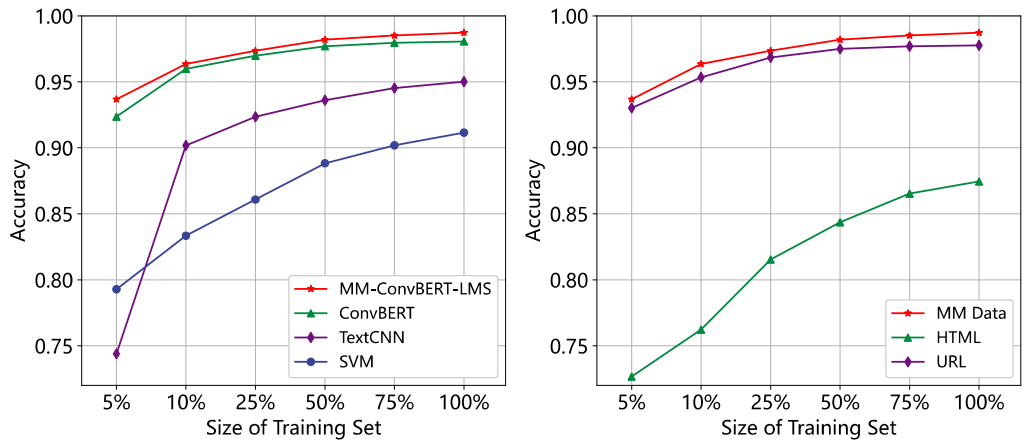

Figure 12. Effect of dataset size on detection effectiveness. **Left**: Comparison between multimodal models. **Right**: Comparison between single-modal and multimodal models.

Phung [18] et al. have shown that unbalanced data distribution is another challenge for real-world malicious web page detection tasks. In our study, we also examined the effectiveness of the MM-ConvBERT-LMS model under different positive and negative sample ratios. We conducted experiments by adjusting the fraction of malicious samples while keeping the total number of training sets fixed at 5000 samples. To better observe the changes in all four-evaluation metrics, we visualized the results, which are presented in Figure 13. Our findings show that the baseline models are susceptible to sample fractions, resulting in significantly lower accuracy and recall when there are fewer malicious samples. On the other hand, the stability performance of the MM-ConvBERT-LMS model is higher, and it can maintain an accuracy above 94% even when the malicious and normal samples are 1:9, which is 3.34% higher than the original ConvBERT model. Hence, the MM-ConvBERT-LMS model has a higher practical potential.

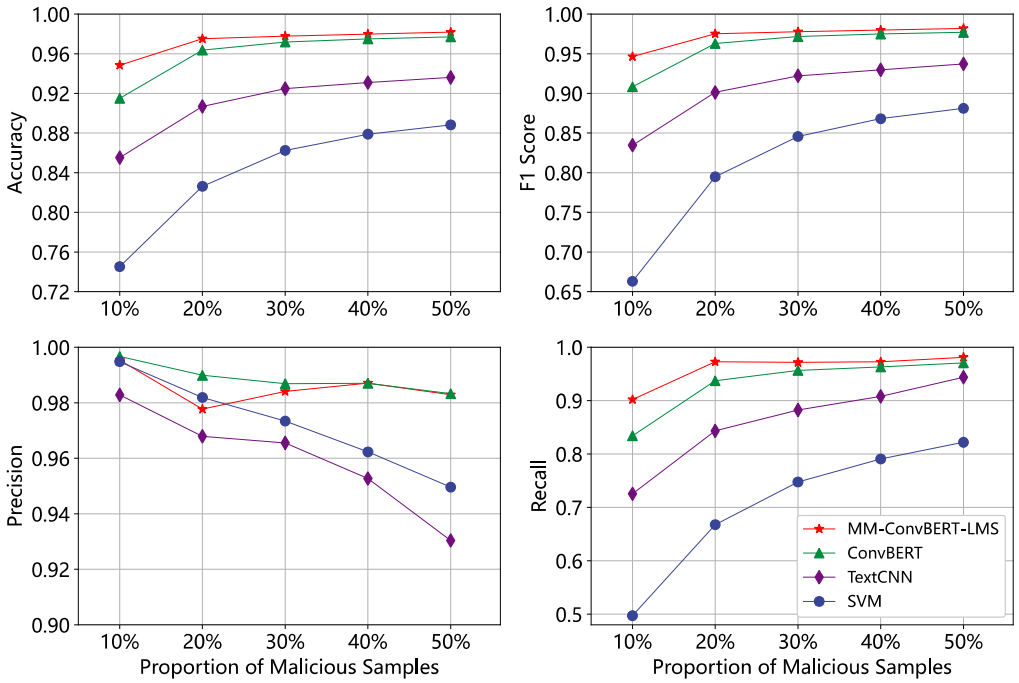

Figure 13. Impact of positive and negative sample ratios on multi-modal models.

According to Tables 3 and 4, it can be observed that when the training data is sufficiently abundant, the addition of LMS has a limited effect on the performance improvement of the single-modal models. To further verify the effectiveness of LMS, we also evaluated the performance of several single-modal models under the same settings, and the results are presented in Figure 14. As expected, we observed that the difference in accuracy between ConvBERT-LMS and the original ConvBERT becomes more pronounced as the percentage of malicious samples decreases. One possible explanation for the observed results is that, under the condition of an evenly distributed dataset, both the standard softmax and LMS-based models can learn good classification boundaries by utilizing sufficient data and confidently predict the test samples, leading to no significant performance differences between the two. However, in cases where the dataset is extremely imbalanced, ConvBERT, with its large number of parameters, is more susceptible to overfitting. In such situations, when standard softmax is employed, the model is only able to fit a boundary that distinguishes between positive and negative samples, without taking into account the inter-class distance between them. Consequently, the model's generalization ability may be weakened. In contrast, the LMS-based optimization objective not only helps the model fit the classification boundary, but also requires the model to reduce the intra-class distance and increase the inter-class distance during the learning process. This encourages the model to better distinguish between positive and negative samples, thus improving the model's performance. Therefore, these findings provide partial evidence for the effectiveness of LMS.

Furthermore, it is evident that the size of the model has a significant impact on the detection performance. Intuitively, larger models tend to achieve better results. However, more parameters significantly increase the training cost and lead to slow inference, making it impractical for real-time malicious web detection scenarios. To explore the effect of model parameter size on the

detection effect, we trained several pre-trained models of different sizes, as shown in Figure 15, and the parameter details of these models are shown in Table 6. Our experiments show that, for both ConvBERT and MM-ConvBERT-LMS models, increasing the model size within a certain limit does improve the detection effectiveness of the model against malicious web pages. However, as demonstrated by comparing the BERT-base and BERT-large models, using a model that is too large not only incurs high computational costs but also negatively affects the results due to overfitting and other issues. In addition, the experimental results also show that MM-ConvBERT-LMS maintains an advantage even when its number of parameters is only 1/9 of BERT, making it a viable option for some malicious web page detection tasks with limited computing power or high real-time requirements.

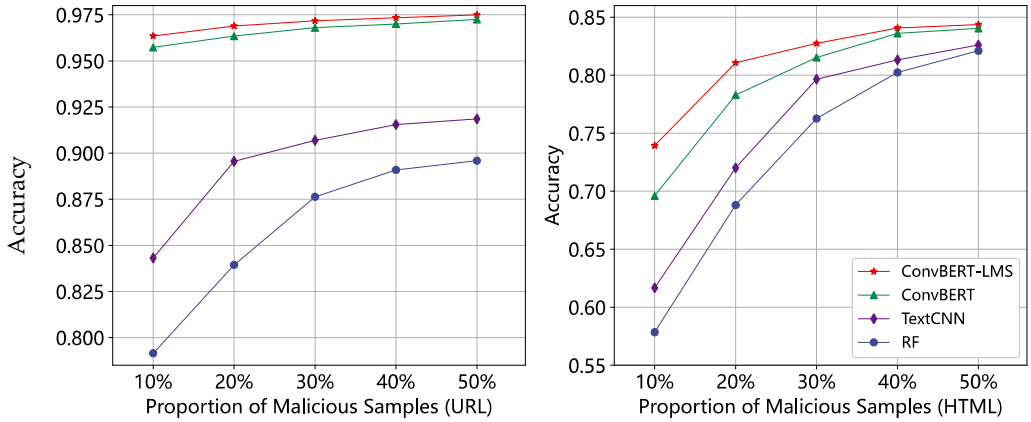

Figure 14. Impact of positive and negative sample ratios on single-modal models.

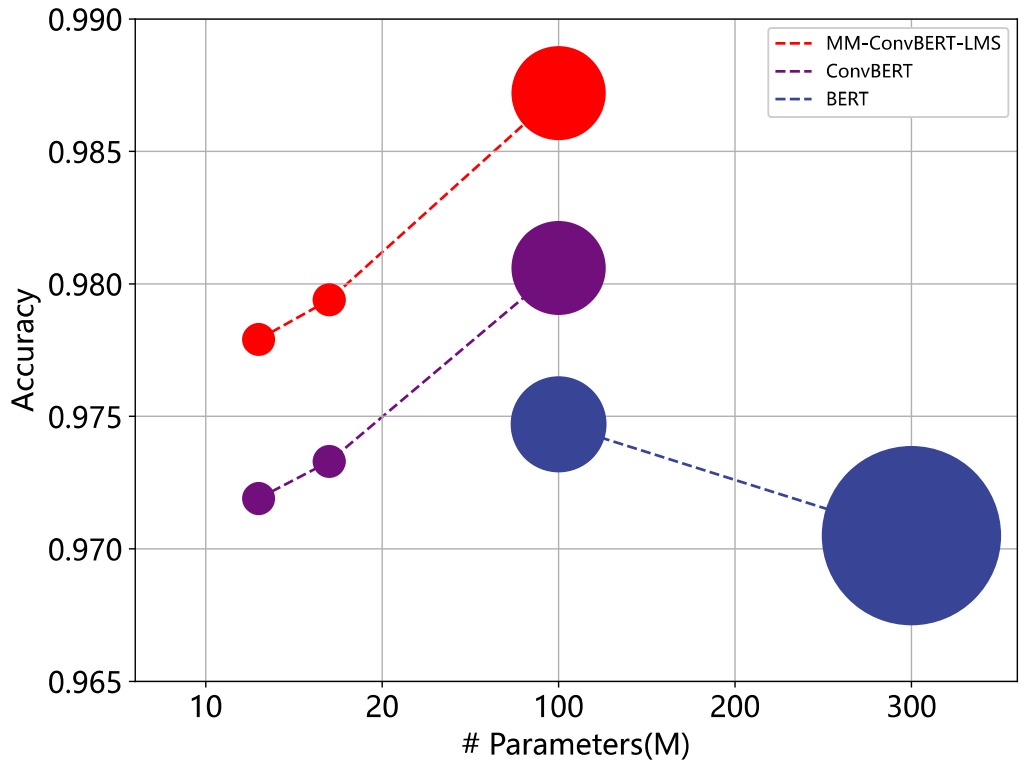

Figure 15. Effect of model size on detection performance. The area of the circle represents the inference FLOPs of the models.

Table 6. Information on ConvBERT and BERT models of different sizes.

Models	Version	Layers	Heads	Hidden Size	Parameters (M)
ConvBERT	Small	12	4	128	13
	Medium-Small	12	8	384	17
	Base	12	12	768	106
BERT	Base	12	12	768	109
	Large	24	16	1024	335

4.3. Ablation Experiment

To verify the effectiveness of each component of the MM-ConvBERT-LMS model, ablation experiments were performed while maintaining the same experimental dataset and hyperparameters. The full MM-ConvBERT-LMS model was used as the baseline and the specific effects of various components on the experimental results were analyzed.

(1) In the model comparison experiments, we observed a significant impact of mode-type encoding on the ALBERT model and hypothesized that it might also affect the performance of the MM-ConvBERT-LMS model. To test this hypothesis, we constructed a comparison model without modal-type encoding and evaluated its performance. Moreover, we investigated the ability of the model to handle modal-missing data (single-modal data) before and after removing the modal-type encoding. It is worth noting that, in contrast to the experiments reported in Tables 3 and 4, the model used in this evaluation was trained on a multimodal dataset, rather than on single-modal datasets.

(2) We conducted an ablation test by replacing the LMS of the classification layer with a standard softmax in the ablation test to further validate the effectiveness of the design.

(3) In this paper, we have selected a single-stream ConvBERT as the backbone model, as we believe that it can effectively learn representations and modal fusion. In the field of multi-modal, two-stream neural networks are also a popular architecture that involves using multiple backbones to process data from different modalities. To further investigate the impact of architectures on detection tasks. To verify the impact of the model structure, we constructed and evaluated the performance of a two-stream model, which consists of two parallel ConvBERTs. One ConvBERT is responsible for learning the features of HTML tag sequences, while the other focuses on learning the features of URL strings.

(4) The coarse-grained modal matching task is the core of our design to help the model learn cross-modal relationships. We analyzed the contribution of the auxiliary module to the model's performance in detecting malicious web pages by removing the coarse-grained modal matching task.

The experimental results are shown in Table 7. According to the experimental results, it can be seen that the removal of modal-type encoding, LMS, and coarse-grained modal matching auxiliary tasks all lead to a decrease in the effectiveness of the model. These findings provide evidence for the effectiveness of these components in detecting malicious web pages.

Table 7. Results of the ablation experiments. MTE is the abbreviation for modal-type encoding. After the change, the result of performance reduction is marked in red, the improvement is marked in blue. ↑ means the metric is improved, while ↓ means the metric is decreased.

Modules	Model	Accuracy	Precision	Recall	F1 Score
	MM-ConvBERT-LMS	98.72	98.90	98.55	98.73
The input modules	MM-ConvBERT-LMS w/o HTML	96.42 (2.30 ↓)	94.19 (4.71 ↓)	99.01 (0.46 ↑)	96.54 (2.19 ↓)
	MM-ConvBERT-LMS w/o URL	86.20 (12.52 ↓)	86.81 (12.09 ↓)	85.63 (12.92 ↓)	86.22 (12.51 ↓)
	MM-ConvBERT-LMS w/o MTE	98.41 (0.31 ↓)	98.68 (0.22 ↓)	98.15 (0.40 ↓)	98.42 (0.31 ↓)
	MM-ConvBERT-LMS w/o MTE & HTML	97.07 (1.65 ↓)	96.06 (2.84 ↓)	98.21 (0.34 ↓)	97.13 (1.60 ↓)
	MM-ConvBERT-LMS w/o MTE & URL	82.50 (16.22 ↓)	82.46 (16.44 ↓)	82.92 (15.63 ↓)	82.69 (16.04 ↓)
The output modules	MM-ConvBERT-LMS w/o modal-matching	98.26 (0.46 ↓)	98.43 (0.47 ↓)	98.12 (0.43 ↓)	98.27 (0.46 ↓)
	MM-ConvBERT-LMS w/o LMS	98.61 (0.11 ↓)	98.42 (0.48 ↓)	98.83 (0.28 ↑)	98.62 (0.11 ↓)
Main structure	Two-stream MM-ConvBERT-LMS	98.63 (0.09 ↓)	98.90 (0)	98.37 (0.18 ↓)	98.64 (0.09 ↓)

After removing the modal-type encoding, it is noteworthy that the model's performance on pure HTML input deteriorates significantly, and it tends to rely more on easily learnable URL features. This suggests that the model treats URLs and HTML as the same type of information flow during training, and under the influence of the attention mechanism, the model considers URLs with more salient features as meaningful information while regarding the HTML part with difficult-to-explore features as noisy data, ultimately leading to the neglect of HTML features. However, when using modal-type encoding, ConvBERT can effectively distinguish between HTML and URLs as two different types of data, and the modal matching task can further ensure that the model does not overlook any features. Therefore, the MM-ConvBERT-LMS model not only yields satisfactory results on multimodal data but also proves to be effective in handling modal-missing samples.

The detection accuracy of the two-stream network is slightly lower than that of the single-stream structure. One main reason is the insufficient attention-based modal interaction between the two-stream network's two inputs, where feature fusion occurs only near the output. This limitation results in the model's inability to fully explore the correlation between the two modalities. Meanwhile, the single-stream network has a significant advantage in terms of reducing both the training and inference cost, with 50% fewer model parameters and a training time that is approximately 39% shorter. Therefore, a single-stream network is more cost-effective.

The above experiments indicate that pre-trained models may outperform those trained from scratch. However, the findings are inconclusive, possibly due to the inconsistent parameter sizes between the two types of models, and the possibility that better results achieved with pre-training may also be due to its network architecture or the number of parameters. Therefore, we establish baselines with a model without pre-trained weights (No PTW) and models with frozen layers and observe the impact of freezing different layers on the model's detection performance. The experimental results are shown in Figure 16. Our results suggest that pre-training endows the models with prior knowledge and improves their ability to identify malicious web pages. Specifically, the accuracy of the model without pre-trained weights decreases by around 3% compared to the model with pre-trained weights. At the same time, the model is also negatively impacted when too many layers are frozen. This may be due to the fact that the word vectors obtained by pre-training on the original corpus are not fully applicable to HTML tag sequences or URL strings, and fine-tuning is necessary to improve the representation of these tokens. In addition, since the original ConvBERT model was designed primarily for single-modal tasks, freezing too many layers can also limit the model's adaptability to multi-modal tasks.

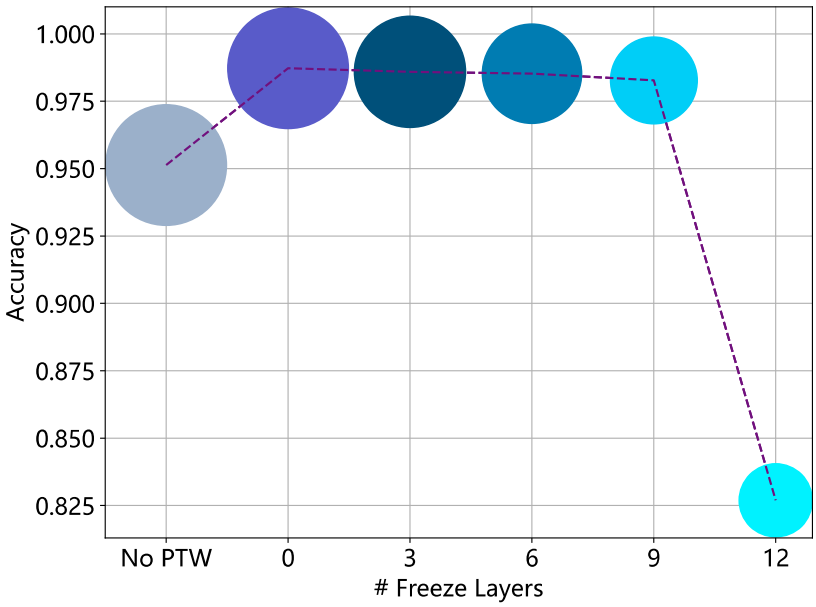

Figure 16. Effect of pre-trained weights on the detection effect. The area of the circle represents the training time.

5. Conclusions

In this paper, we propose a novel approach to address the limitations of current methods in detecting malicious web pages, such as lack of flexibility, insufficient feature extraction, and high

rates of false positives and false negatives. Our proposed approach is an efficient model based on multi-modal learning and a pre-trained model. In terms of model structure, the ConvBERT-based backbone network can represent the local and global information of the input features through the mixed attention module. By adding the modal-type encoding, this model can avoid feature confusion and other problems, and enhance its capacity to analyze multi-modal features. On the other hand, the coarse-grained modal matching task is a simple yet efficient auxiliary training method that helps the model learn the relationship between two modalities. The optimization objective based on LMS also enables the model to learn larger classification boundary distances and improve robustness. Experimental results show that our proposed model achieves a detection accuracy of 98.72%, outperforming the baseline methods. Thus, our approach shows promise in the field of malicious web page detection.

However, there are still certain aspects that require improvement:

(1) **More adapted vocabulary and tokenizer**. In this paper, we use the vocabulary and tokenizer used in the ConvBERT model pre-training process to pre-process the input data. However, as these pre-training tasks are built on a regular corpus of text, the vocabulary may lack feature elements in HTML or URLs. Therefore, it is critical to improving the vocabulary and tokenizer to further enhance the model's effectiveness.

(2) **A more complex yet realistic dataset**. In this paper, the multi-modal samples detected are those where both URL and HTML are malicious. Nevertheless, in a real-world scenario, only one of the HTML or URLs may be malicious, while the other features could be normal. Therefore, collecting such data and designing fine-grained modal matching tasks for these scenarios can broaden the scope of the proposed approach's application. It is essential to consider this complexity to evaluate the model's effectiveness in a more realistic setting.

In the next step of our research, we will not only focus on enhancing the detection performance of the model but also improve its robustness against adversarial sample attacks [38] without affecting the detection accuracy. Furthermore, techniques such as model distillation [39] and pruning will be explored to improve the operational efficiency and practical performance of the model in real-world scenarios. Additionally, by integrating the model into browser plug-ins, we hope to provide real-time protection to users.

Author Contributions: Methodology, X.T., B.J. and J.W.; Validation, Y.Y.; Writing—review & editing, Q.S. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Social Science Foundation Key Project (20AZD114) and the National Key Research and Development Program of China (2022YFB3103200).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All datasets are publicly available.

Acknowledgments: The authors wish to express their appreciation to the reviewers for their helpful suggestions which greatly improved the presentation of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mohammad, R.M.; Thabtah, F.; McCluskey, L. Tutorial and critical analysis of phishing websites methods. *Comput. Sci. Rev.* **2015**, *17*, 1–24. [CrossRef]
2. 2021 China Cybersecurity Report. Available online: http://it.rising.com.cn/dongtai/19858.html (accessed on 23 December 2022).
3. Prakash, P.; Kumar, M.; Kompella, R.R.; Gupta, M. Phishnet: Predictive Blacklisting to Detect Phishing Attacks. In Proceedings of the 2010 IEEE INFOCOM, San Diego, CA, USA, 14–19 March 2010. [CrossRef]
4. Chou, N. Client-side defense against web-based identity theft. In Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS'04), San Diego, CA, USA, 24–27 February 2004.
5. Nicomette, V.; Kaâniche, M.; Alata, E.; Herrb, M. Set-up and deployment of a high-interaction honeypot: Experiment and lessons learned. *J. Comput. Virol.* **2010**, *7*, 143–157. [CrossRef]
6. Atrees, M.; Ahmad, A.; Alghanim, F. Enhancing Detection of Malicious URLs Using Boosting and Lexical Features. *Intell. Autom. Soft Comput.* **2022**, *31*, 1405–1422. [CrossRef]
7. Wang, Z.; Ren, X.; Li, S.; Wang, B.; Zhang, J.; Yang, T. A Malicious URL Detection Model Based on Convolutional Neural Network. *Secur. Commun. Netw.* **2021**, *2021*, 5518528. [CrossRef]

8. Yuan, J.; Liu, Y.; Yu, L. A Novel Approach for Malicious URL Detection Based on the Joint Model. *Secur. Commun. Netw.* **2021**, *2021*, 4917016. [CrossRef]
9. Luo, C.; Su, S.; Sun, Y.; Tan, Q.; Han, M.; Tian, Z. A Convolution-Based System for Malicious URLs Detection. *Comput. Mater. Contin.* **2020**, *62*, 399–411. [CrossRef]
10. Yan, X.; Xu, Y.; Cui, B.; Zhang, S.; Guo, T.; Li, C. Learning URL Embedding for Malicious Website Detection. *IEEE Trans. Ind. Informatics* **2020**, *16*, 6673–6681. [CrossRef]
11. Khan, N.; Abdullah, J.; Khan, A.S. Defending Malicious Script Attacks Using Machine Learning Classifiers. *Wirel. Commun. Mob. Comput.* **2017**, *2017*, 5360472. [CrossRef]
12. Wang, Y.; Cai, W.-D.; Wei, P.-C. A deep learning approach for detecting malicious JavaScript code. *Secur. Commun. Netw.* **2016**, *9*, 1520–1534. [CrossRef]
13. Huang, Y.; Li, T.; Zhang, L.; Li, B.; Liu, X. JSContana: Malicious JavaScript detection using adaptable context analysis and key feature extraction. *Comput. Secur.* **2021**, *104*, 102218. [CrossRef]
14. Alex, S.; Rajkumar, T.D. Spider bird swarm algorithm with deep belief network for malicious JavaScript detection. *Comput. Secur.* **2021**, *107*, 102301. [CrossRef]
15. Fang, Y.; Huang, C.; Liu, L.; Xue, M. Research on Malicious JavaScript Detection Technology Based on LSTM. *IEEE Access* **2018**, *6*, 59118–59125. [CrossRef]
16. Fang, Y.; Huang, C.; Su, Y.; Qiu, Y. Detecting malicious JavaScript code based on semantic analysis. *Comput. Secur.* **2020**, *93*, 101764. [CrossRef]
17. Fang, Y.; Huang, C.; Zeng, M.; Zhao, Z.; Huang, C. JStrong: Malicious JavaScript detection based on code semantic representation and graph neural network. *Comput. Secur.* **2022**, *118*, 102715. [CrossRef]
18. Phung, N.M.; Mimura, M. Detection of malicious javascript on an imbalanced dataset. *Internet Things* **2021**, *13*, 100357. [CrossRef]
19. Hou, Y.-T.; Chang, Y.; Chen, T.; Laih, C.-S.; Chen, C.-M. Malicious web content detection by machine learning. *Expert Syst. Appl.* **2010**, *37*, 55–60. [CrossRef]
20. Altay, B.; Dokeroglu, T.; Cosar, A. Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection. *Soft Comput.* **2018**, *23*, 4177–4191. [CrossRef]
21. Kazemian, H.; Ahmed, S. Comparisons of machine learning techniques for detecting malicious webpages. *Expert Syst. Appl.* **2014**, *42*, 1166–1177. [CrossRef]
22. Wang, R.; Zhu, Y.; Tan, J.; Zhou, B. Detection of malicious web pages based on hybrid analysis. *J. Inf. Secur. Appl.* **2017**, *35*, 68–74. [CrossRef]
23. Deng, W.; Peng, Y.; Yang, F.; Song, J. Feature optimization and hybrid classification for malicious web page detection. *Concurr. Comput. Pr. Exp.* **2020**, *34*, e5859. [CrossRef]
24. Amrutkar, C.; Kim, Y.S.; Traynor, P. Detecting Mobile Malicious Webpages in Real Time. *IEEE Trans. Mob. Comput.* **2016**, *16*, 2184–2197. [CrossRef]
25. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. *arXiv* **2018**, arXiv:1802.05365.
26. Lu, J.; Batra, D.; Parikh, D.; Lee, S. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 13–23.
27. Kim, W.; Son, B.; Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In Proceedings of the International Conference on Machine Learning (PMLR), Virtual, 18–24 July 2021; pp. 5583–5594.
28. Wang, Y.; Shen, Y.; Liu, Z.; Liang, P.P.; Zadeh, A.; Morency, L.-P. Words Can Shift: Dynamically Adjusting Word Representations Using Nonverbal Behaviors. *Proc. Conf. AAAI Artif. Intell.* **2019**, *33*, 7216–7223. [CrossRef] [PubMed]
29. Zhang, H.; Qian, S.; Fang, Q.; Xu, C. Multi-Modal Meta Multi-Task Learning for Social Media Rumor Detection. *IEEE Trans. Multimed.* **2021**, *24*, 1449–1459. [CrossRef]
30. Jiang, Z.H.; Yu, W.; Zhou, D.; Chen, Y.; Feng, J.; Yan, S. ConvBERT: Improving BERT with span-based dynamic convolution. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Virtual, 6–12 December 2020; pp. 12837–12848.
31. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, 5999–6009.
32. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805, 2018.
33. Clark, K.; Luong, M.T.; Le, Q.V.; Manning, C.D. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv* **2020**, arXiv:2003.10555, 2020.
34. Kobayashi, T. Large Margin In Softmax Cross-Entropy Loss. In Proceedings of the British Machine Vision Conference, Cardiff, UK, 9–12 September 2019; p. 139.
35. Saxe, J.; Sanders, H. *Malware Data Science: Attack Detection and Attribution*; No Starch Press: San Francisco, CA, USA, 2018; ISBN 978-159-327-859-5.
36. Faizan, A. Using Machine Learning to Detect Malicious URLs. Available online: https://github.com/faizann24/Using-machine-learning-to-detect-malicious-URLs (accessed on 6 September 2022).

37. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **2019**, arXiv:1909.11942, 2019.
38. Yuan, X.; He, P.; Zhu, Q.; Li, X. Adversarial Examples: Attacks and Defenses for Deep Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 2805–2824. [CrossRef]
39. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge distillation: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]