Received 5 December 2023, accepted 13 January 2024, date of publication 22 January 2024, date of current version 29 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3357095

RESEARCH ARTICLE

Efficient Classification of Malicious URLs: M-BERT–A Modified BERT Variant for Enhanced Semantic Understanding

BOYANG YU[1], FEI TANG[1], DAJI ERGU[1], RUI ZENG[2], BO MA[1], AND FANGYAO LIU[1]
[1]College of Electronics and Information, Southwest Minzu University, Chengdu 610041, China
[2]College of Pharmacy, Southwest Minzu University, Chengdu 610041, China

Corresponding author: Fangyao Liu (FLIU028@163.COM)

This work was supported in part by the Southwest Minzu University Research Startup Funds under Grant RQD2021061, in part by the National Natural Science Foundation of China under Grant 72174172, and in part by the Science and Technology Planning Project of Sichuan Province of China under Grant 2022YFG0377.

ABSTRACT Malicious websites present a substantial threat to the security and privacy of individuals using the internet. Traditional approaches for identifying these malicious sites have struggled to keep pace with evolving attack strategies. In recent years, language models have emerged as a potential solution for effectively detecting and categorizing malicious websites. This study introduces a novel Bidirectional Encoder Representations from Transformers (BERT) model, based on the Transformer encoder architecture, designed to capture pertinent characteristics of malicious web addresses. Additionally, large-scale language models are employed for training, dataset assessment, and interpretability analysis. The evaluation results demonstrate the effectiveness of the large language model in accurately classifying malicious websites, achieving an impressive precision rate of 94.42%. This performance surpasses that of existing language models. Furthermore, the interpretability analysis sheds light on the decision-making process of the model, enhancing our understanding of its classification outcomes. In conclusion, the proposed BERT model, built on the Transformer encoder architecture, exhibits robust performance and interpretability in the identification of malicious websites. It holds promise as a solution to bolster the security of network users and mitigate the risks associated with malicious online activities.

INDEX TERMS Natural language processing, deep learning, fraudulent URL classification.

I. INTRODUCTION

In recent years, the rapid development of the Internet has led to a significant increase in the number of Internet users worldwide, accompanied by the emergence of various applications and services. However, this widespread adoption of the internet has also brought about a growing concern: the proliferation of malicious websites. Malicious websites pose a significant threat by stealing user information and causing various forms of harm and losses. Consequently, the need to address the spread of malicious websites has become an urgent problem requiring immediate attention. In recent years, the rapid development of the Internet has led to a significant increase in the number of Internet users worldwide, accompanied by the emergence of various applications and services. However, this widespread adoption of the internet has also brought about a growing concern: the proliferation of malicious websites. Malicious websites pose a significant threat by stealing user information and causing various forms of harm and losses. Consequently, the need to address the spread of malicious websites has become an urgent problem requiring immediate attention.

To tackle the issue of malicious websites, there are three mainstream approaches currently. The first method, based on black and white list databases, involves blacklisting known malicious URLs and whitelisting known benign URLs. When

The associate editor coordinating the review of this manuscript and approving it for publication was Kostas Kolomvatsos.

a URL is detected, it is checked against these lists. If a match is found in the blacklist, it is classified as malicious; otherwise, it is deemed benign [1].

The second method involves the use of machine learning (ML) algorithms. With the rise of ML, researchers have explored the application of machine learning techniques in malicious website detection [1], [2]. By utilizing common algorithms such as logistic regression and decision trees, researchers mainly utilize numerical features such as the URL, domain name, and length of the domain name to input into the model for classification [2].

The third method employs deep learning (DL) techniques. Recognizing the limitations of manual feature design, researchers have proposed the use of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models for malicious website detection [3], [4]. These models have shown improved detection accuracy. However, the ever-increasing variability and complexity of malicious websites pose challenges to CNN and LSTM models, which may struggle to capture the semantic understanding necessary for accurate detection.

There are notable limitations within the prevailing methodologies utilized for the identification of malicious websites. The approach relying on black and white list databases heavily leans on established malicious URLs, thus restricting its efficacy in recognizing emergent threats. Likewise, while machine learning (ML) algorithms bolster precision, they demand meticulous manual feature curation and selection, proving insufficient in accommodating the dynamic nature of malicious online entities. Moreover, the application of deep learning techniques like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models demonstrates promising accuracy in pinpointing malicious websites. However, their challenge lies in grasping the evolving semantic nuances crucial amidst the escalating intricacies and variability of these malevolent sites. The limitations of these methodologies stem from their incapacity to swiftly adapt to the escalating dynamism and permutations of malicious online entities. The black and white list database method lacks the adeptness to identify newly emerging malicious URLs, whereas ML algorithms mandate substantial manual intervention. Despite the strides made by deep learning techniques in bolstering accuracy, they still grapple with capturing the evolving semantic intricacies inherent in malicious websites. This underscores the formidable challenges confronting existing methodologies in navigating the perpetually evolving landscape of malevolent online entities.

With the advent of pre-trained language models, such as BERT (Bidirectional Encoder Representations from Transformers), the choice of the BERT model over others can be justified by its prowess as a pre-trained language model, particularly in semantic comprehension. BERT's pre-training methodology, such as the masking task applied to words during training, allows the model to grasp semantic relationships between words. For instance, by masking specific characters within a URL during pre-training, the model enhances its comprehension of malicious web addresses. This underlines BERT's advantage in detecting malicious URLs and justifies its selection over alternative models. We can leverage the shortcomings of the aforementioned methods and propose the M-BERT model based on the BERT language model. The M-BERT model incorporates more robust semantic understanding capabilities compared to CNN and LSTM models, allowing for more effective detection of malicious web addresses. BERT has gained remarkable performance in classification tasks, demonstrated in prior benchmarks, positioned it as a frontrunner for consideration. BERT's standard-setting performance set a crucial benchmark for achieving robust results, especially given its efficiency in resource usage and training time. The decision to adopt BERT as the benchmark model stemmed from a careful balance between cutting-edge innovation and practical feasibility. BERT's historical significance, proven performance, and efficiency in resource utilization made it an ideal choice for this research. By harnessing the power of language models, the M-BERT model aims to address the limitations of existing approaches and enhance the accuracy of malicious website detection.

The following is a summary of the main points of this work:

1) The paper proposes an improved M-BERT model based on BERT for detecting malicious web addresses.
2) The study employs various language models and finds that the M-BERT model achieves the best performance, with a reported precision of 94.42%.
3) The major contribution lies in redesigning the BERT model specifically for malicious website detection, resulting in higher accuracy.
4) The interpretability of the trained model is analyzed, shedding light on its reliability and addressing the black box nature of deep learning.
5) The study highlights the potential of utilizing powerful language models, like M-BERT, to accurately detect and prevent malicious web addresses.

II. RELATED WORK

A. TRADITIONAL DETECTION METHODS

In recent years, researchers have proposed various techniques to detect malicious URLs and have achieved notable advancements, which can be primarily categorized into the following three methods.

1) DETECTING BY BLACKLISTS AND WHITELISTS

Utilizing Blacklists and Whitelists, where known malicious websites are enlisted in a blacklist for identification [1]. "Blacklist" consists of known malicious or perceived unsafe websites, typically encompassing sites associated with malware distribution, phishing, fraudulent activities, among others. "Whitelist" contains known safe and trusted websites, allowing users unrestricted access to them, while other websites may require additional scrutiny or be subject to restrictions. When accessing a website, if the domain name is already listed should either be blocked immediately or trigger a warning message. However, this method has certain drawbacks such as delayed up-dates to the blacklists and limited resistance against interference. It fails when the same website changes its domain name or when the domain name is automatically generated by an algorithm. What's more, Whitelists can impose constraints on users' freedom to access websites and necessitate continuous updates to ensure the inclusion of all trusted sites.

2) DETECTING BASED ON LEXICAL FEATURES

The approach based on lexical features, which includes characteristics like URL length and specific characters, represents a fundamental technique in the field of malicious URL detection. Nonetheless, there are some limitations and challenges associated with this feature-based approach that need to be elaborated upon:

1) Limited Scope: The feature-based approach primarily relies on a set of predefined lexical features, such as URL length, presence of specific keywords, and special characters. While these features can be effective for detecting known types of malicious URLs, they have a limited scope and might not adequately capture emerging or novel threats.
2) Cultural and Grammatical Disparities: One of the significant limitations is that this approach may not be suitable for detecting new attacks or threats that exploit cultural and grammatical rule disparities. Malicious actors often adapt their tactics to specific regions or languages, and these adaptations might not be captured by the predefined lexical features.
3) Dynamic Nature of Attacks: Malicious URL attacks are dynamic and ever-evolving. Attackers constantly develop new techniques and tactics to evade detection. The feature-based approach, with its fixed set of features, may struggle to keep up with these evolving attack strategies.
4) False Positives and False Negatives: Depending solely on lexical features can lead to both false positives and false negatives. False positives occur when benign URLs exhibit characteristics that trigger alarms, while false negatives occur when malicious URLs do not exhibit typical malicious features.
5) Lack of Contextual Understanding: This approach often lacks the ability to understand the broader context in which a URL is being used. It may flag legitimate URLs that happen to contain certain keywords or characters, leading to user in-convenience and a loss of trust in the detection system.

3) DETECTING BASED ON PAGERANK [5]

The third prevalent method involves incorporating PageRank values to aid in the detection of malicious URLs. PageRank is an algorithm introduced by Google for webpage ranking, assessing the significance of webpages [5]. It measures the degree to which a webpage is linked by other important webpages. In the context of malicious URL detection, researchers utilize PageRank values to assess the credibility of webpages. If a URL directs to a webpage with a higher PageRank value, it is more likely to be a trustworthy webpage. Conversely, if a URL leads to a webpage with a lower PageRank value, it may indicate a malicious webpage. By combining PageRank values with other features such as the count of friend links and sensitive words, the accuracy of malicious URL detection can be enhanced. However, this approach still exhibits certain vulnerabilities, as attackers can manipulate sensitive words and purchase friend links to inflate the PageRank value of malicious webpages, thereby evading detection. For instance, attackers can create fake backlinks on other websites that direct to their own site, and search engines typically consider backlinks as indicators of a website's authority, so an increase in fake backlinks can potentially boost the PageRank value of a webpage. Additionally, attackers may misuse sensitive vocabulary on webpages, such as using offensive or inappropriate language in titles, descriptions, and content. While this practice may attract undesirable audiences, it can also lead to search engines penalizing the webpage due to its content being deemed unsafe or inappropriate.

B. MACHINE LEARNING

Machine learning-based approaches for malicious URL detection primarily focus on feature extraction, followed by classification using classical machine learning algorithms. Features extracted from malicious URLs are crucial for training models to accurately differentiate between malicious and benign websites. The process of feature extraction involves converting raw input data into a numerical representation that can be processed by machine learning algorithms [6]. This part also provides additional details on the types of features extracted and their conversion into numerical formats, along with classical machine learning algorithms commonly used for this task.

1) FEATURE TYPES

1) Lexical Features: Lexical features capture textual characteristics of the URL. These features include:
    a) URL Length: The length of the URL, which can be an indicator of suspicious or excessively long URLs.
    b) Specific Keywords: The presence of specific keywords in the URL, which may be indicative of malicious intent.
    c) Special Characters: The use of special characters in the URL, which might signal obfuscation or attempts to deceive.
    d) Digits or Symbols: The presence of digits or symbols within the URL, which could be associated with certain types of threats.

2) Website Features: Website features examine attributes related to the website hosting the URL. These attributes include:
    a) Domain Age: The age of the domain associated with the URL. New or recently created domains may be more suspicious.
    b) Hosting Server's IP Address: The IP address of the server hosting the website. Unusual or suspicious IP addresses can raise red flags.
    c) SSL Certificate Status: The status of the SSL certificate used by the website, where an expired or invalid certificate could indicate security issues.
    d) Presence of Redirects or Suspicious Domains: Detection of redirects in the URL or references to known suspicious domains.

2) CONVERSION TO NUMERICAL FORMATS

1) Categorical Features:
Categorical features, such as the presence of specific keywords or SSL certificate status, can be converted into numerical format using one-hot encoding. Each category is represented as a binary feature. For example, if a specific keyword is present, its corresponding binary feature is set to 1; otherwise, it is set to 0. This transformation allows machine learning algorithms to work with categorical data effectively.
2) Numeric Features:
Numeric features, such as URL length or domain age, are already in numerical format and do not require additional conversion. These features can be directly used as numerical values in machine learning models.

3) CLASSICAL MACHINE LEARNING ALGORITHMS

1) Logistic Regression:
Logistic Regression is a linear model used for binary classification. It maps the linear combination of input features through a logistic function (also known as the Sigmoid function), which transforms continuous real-valued outputs into values between 0 and 1, representing the probability of belonging to two classes. In the context of malicious URL detection, logistic regression is employed to model the relationship between various features, such as URL characteristics, keyword presence, and the use of special characters, and the likelihood of a URL being malicious. Notably, Darling. M et al. achieved a good effect on the accuracy of URL detection realized by logistic regression algorithm [7].
2) Decision Trees:
Decision Trees are tree-like structures that recursively partition data based on different features, resulting in a set of decision rules. Each internal node represents a feature test, while each leaf node corresponds to a class. Decision trees can be used to create decision rules for classification tasks, such as categorizing a URL as malicious if it contains specific keywords or characters. Decision tree models are known for their interpretability but may be prone to overfitting. In the study conducted by Patil et al., the utilization of the decision tree algorithm yielded favorable predictive outcomes, based on a dataset comprising 52082 labeled URLs [8].
3) Random Forest:
Random Forest is an ensemble learning method that combines multiple decision tree models. Each tree is trained on different data subsets, and a random subset of features is used for splitting in each tree to reduce overfitting. Random Forest is employed to enhance classification accuracy, handle high-dimensional feature spaces, and decrease model variance. It can capture more information and improve the detection performance of malicious URLs. M. Weedon et al. utilized the random forest algorithm for implementation, and initial results from experiments indicate that the Random Forest algorithm performs the best yielding an 86.9% accuracy [9].
4) Support Vector Machines (SVM):
SVM is a binary classification algorithm that identifies an optimal hyper-plane in the feature space to maximize the margin between different classes. SVM is suitable for nonlinear classification problems, and it can be trans-formed using kernel functions. In the context of malicious URL detection, SVM can be utilized to separate malicious and non-malicious URLs by finding a hyperplane in the feature space. It effectively handles high-dimensional feature spaces [10].
5) Gradient Boosting:
Gradient Boosting is an ensemble technique that iteratively focuses on mis-classified instances by combining multiple weak classifiers to construct a robust classifier. It progressively reduces classification errors. In the context of malicious URL detection, Gradient Boosting is applied to improve detection accuracy by iteratively refining the performance of the classifier. It excels in handling noisy and imbalanced datasets. The work by T. Manyumwa et al. successfully employed XGBoost (eXtreme Gradient Boosting) in the detection of malicious URLs. In summary, both XGBoost and AdaBoost (Adaptive Boosting) performed exceptionally well in detecting URL types [11].

These classical machine learning algorithms leverage the extracted features and their numerical representations to train models that can effectively classify malicious URLs. By learning the distribution of malicious websites in the dataset, machine learning models can predict new categories [12]. However, due to the variability and time-sensitivity of malicious websites, machine learning often fails to extract deeper features, leading to reduced accuracy.

In the field of malicious URL detection, "deeper features" typically refer to more complex, abstract, or implicit information that needs to be considered in URL analysis. For traditional machine learning algorithms, features are often manually designed or selected rather than learned by the machine itself. In this scenario, hidden feature in-formation cannot be recognized, and machine learning methods may overlook these details. These characteristics include:

4) BEHAVIORAL FEATURES

Considering how users interact with URLs, such as user click history, dwell time, and navigation path within the URL. These features can provide deeper insights into the relationship between URLs and user behavior.

5) CONTENT FEATURES

Analyzing the content of the web page to which the URL points and detecting the presence of malicious code, malicious links, or false information. This may require the use of natural language processing (NLP) techniques and content analysis.

6) CONTEXTUAL FEATURES

Considering the context in which the URL appears, such as whether it is included in social media posts, emails, or linked from specific sources. Contextual information can reveal the intent and credibility of the URL.

7) USER FEATURES

Taking into account the characteristics of the users accessing the URL, such as their location, device, and browsing history. This information can be used for personalized malicious URL detection.

Unlike traditional machine learning methods, deep learning and pre-trained models like BERT typically have the ability to automatically learn advanced feature representations without the need for manual feature engineering. These models are trained on large-scale data and can automatically extract relevant features from the data, including deep semantic features. This is an advantage of deep learning because it reduces the need for manual intervention, although it also requires more computational resources and data to train these complex models.

C. DEEP LEARNING

The advancement of deep learning technology has led to an increasing number of studies applying it to the detection of malicious websites. This research can be categorized into two main approaches: one based on convolutional neural networks (CNN) for feature extraction [3], [4], and the other based on long short-term memory (LSTM) networks for feature extraction. For the CNN-based algorithm, Wang Z et al. proposed to use dynamic convolution and vector embedding to extract URL features for classification [13]. The reliance on website features alone for feature extraction in the con-text of malicious website detection poses a notable limitation. This approach fails to capture the entirety of relevant information due to its narrow focus. Malicious web-sites exhibit diverse characteristics that extend beyond web-page content, including domain attributes, network traffic patterns, and user behavior.

Thus, relying solely on website features results in an incomplete feature set that may not adequately support effective detection. Jiang J et al. used CNN network to combine the URL features extracted by CNN with DNS in the network for detection, improving the detection accuracy [14]. Typically, convolutional networks are used for feature extraction, along with other website features as input. For the method used in combination of LSTM and CNN, Le et al. artificially detected malicious URLs and combine the characteristics of LSTM with CNN to propose URLNet, which can process various types of URL features and also take the learned vector as input [15].

Subsequent work has proposed different network structures by combining CNN with networks related to natural language processing. For example, Yang et al. proposed a combination algorithm of CNN and CGRU [16], which first transforms URL into a sequence of keywords, and then uses CGRU and CNN to extract and classify the sequence features. What's more, a convolutional neural network (CNN) associated with Genetic Algorithm(GA) model used to classify malicious URLs is authored by Wu et al.in 2022 [17]. They pointed out that previous models for malicious URL classification have not achieved good results. The GA is used to reduce the data dimension of the grammatical features, structural features, and probabilistic features in the extracted malicious URL text. The convolutional neural network is then used to establish the model and classify the malicious URL. Through experimental verification, the model has achieved good results and improves the accuracy of malicious URL recognition compared to traditional machine learning models. Overall, deep learning-based malicious URL detection methods no longer rely on manual feature design, exhibiting higher feature extraction and generalization capabilities. However, these methods face challenges when confronted with rapidly evolving and diverse modern malicious websites. Additionally, the black-box nature of deep learning models makes it difficult to understand their internal workings. Furthermore, false positives and false negatives remain challenges in deep learning-based malicious URL detection systems due to the diverse attack modes and variations of malicious URLs. In order to address the issue of false positives and false negatives in malicious URL detection, some researchers have employed Transformer models. By passing the URLs through the Trm module, they are better represented, which leads to a reduced rate of false positives and false negatives compared to using conventional CNN and LSTM models [18]. As of late, with the advent of language models, their potential is being recognized across various domains. It is worth noting that language models are primarily based on Transformer encoder or decoder mechanisms. Capitalizing on this development, we propose a novel Trm model for malicious URL detection.

When addressing the limitations of previous research, several key aspects warrant consideration: Firstly, previous methods have exhibited limited adaptability to emerging threats. For instance, blacklist-based and lexically derived detection methods often struggle to promptly adapt to the emergence of novel malicious URLs. This constraint leads to decreased accuracy when confronting emerging threats.

Secondly, traditional methods tend to confine themselves to static feature detection. Approaches based on lexical features and blacklists typically focus solely on static attributes such as URL length and specific characters, neglecting semantic and dynamic attributes. This fixed feature set inadequately captures the dynamic alterations and contextual variations of malicious websites. Furthermore, prior studies have underutilized deep features. Although traditional machine learning approaches extract a series of features, they often fail to delve deeply into user behavior, content semantics, and contextual information, leading to limitations in addressing the diversity and complexity of malicious URLs.

Taking cues from the limitations and shortcomings identified in previous research efforts, the M-BERT model based on the Transformer [19] architecture leverages pre-trained language models to automatically learn semantic nuances and associations within text, enabling a more comprehensive capture of deeper features underlying URLs. Compared to conventional feature extraction methods, the M-BERT model demonstrates greater adaptability and accuracy in identifying novel threats. By utilizing pre-trained language models, the M-BERT model can comprehensively explore the characteristics of malicious URLs without the need for manual feature engineering. It not only considers conventional URL features but also identifies malicious behavior from semantic and contextual perspectives, thereby enhancing detection accuracy and robustness. Leveraging the automatic learning capability of the Transformer model, the M-BERT model mitigates issues related to false positives and false negatives. This model excels in comprehending the semantic correlations within URLs, enabling a more precise differentiation between malicious and benign URLs, consequently enhancing detection precision and reliability.

Lastly, previous research might have fallen short in comprehensive feature selection and model optimization. This shortfall hampers a thorough understanding of the complexity and variations of malicious URLs.

D. TRANSFORMER

The Transformer framework has gained significant attention with the introduction of large language models across different languages. It serves as the foundation for various language models and has revolutionized natural language processing tasks. The key principle of the Transformer is to encode and decode input sequences using a self-attention mechanism [19].

In the encoder part of the Transformer, the self-attention mechanism plays a crucial role in capturing the relationship between different positions within the input sequence. It computes a weight for each position based on its relationship with other positions in the sequence. By weighting and averaging the encoding vectors of these other positions, the encoder generates a comprehensive encoding vector for each position. This allows each position's encoding vector to encapsulate information from the entire input sequence, resulting in a more robust representation.

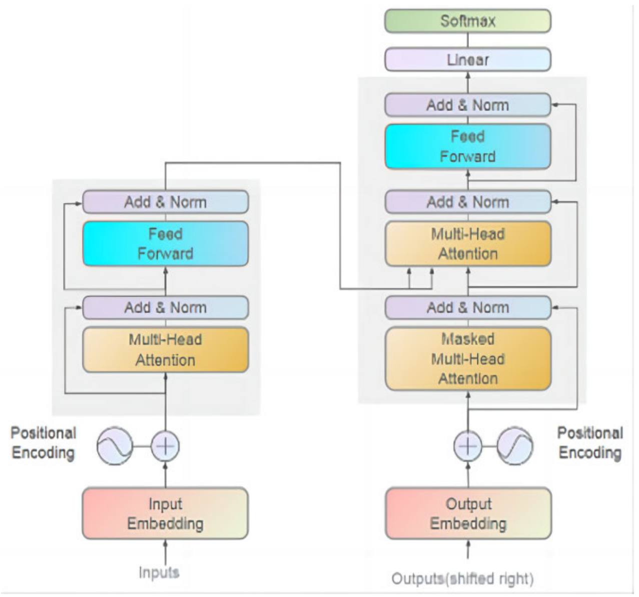

FIGURE 1. The transformer - model architecture [19].

In the decoder part of the Transformer, the self-attention mechanism facilitates the relationship between the input sequence and the output sequence. For each position in the output sequence, the self-attention mechanism calculates a weight based on its relationship with all positions in the input sequence. Similar to the encoder, the decoder then weights and averages the encoding vectors of all positions in the input sequence. This generates a decoding vector for each position in the output sequence that incorporates information from both the input sequence and the generated output sequence. This comprehensive representation aids in generating accurate and contextually relevant output sequences.

By utilizing the self-attention mechanism in both the encoder and decoder, the Transformer model can effectively capture dependencies and relationships within in-put sequences. This allows it to better represent input and output sequences, thereby improving the performance of various natural language processing tasks. Currently, the Transformer architecture has achieved significant success in various tasks within the field of NLP, such as machine translation [20], text generation [21], [22], text classification [22], [23], question-answering systems [21], [24], language modeling [25], etc. Particularly in text classification tasks, BERT models and their variants have become standard models. These remarkable achievements highlight the powerful potential of the

Transformer architecture in the NLP field, providing a strong direction for the work presented in this paper. On the other hand, considering that the nature of malicious URL classification is essentially text classification, this paper chooses to improve it based on the BERT model.

III. PROPOSED ARCHITECTURE

The exploration of Transformer architecture, as highlighted in the prior section, significantly influenced subsequent advancements such as BERT in the realm of natural language processing. The decision-making process surrounding the adoption of BERT as the benchmark model echoes this influence. It's important to note that when this research was initiated, BERT was one of the pioneering pre-trained language models in the field of natural language processing (NLP). Its groundbreaking release marked a significant milestone in NLP research. This context provides a historical perspective on why BERT was initially considered. Prior research and benchmarking had shown BERT's remarkable performance in NLP tasks of classifications. This track record of success made it a strong candidate for this re-search. BERT had set a high standard for model performance, which was crucial for achieving robust results.

The mention of RoBERTa [26] and XLNet [27] highlights the evolving landscape of NLP models. These models, while promising, introduced certain complexities. RoBERTa and XLNet may excel in specific tasks and provide advanced features, but they often require larger computational resources and longer training times. This introduces practical challenges, such as increased costs and time constraints, especially for re-search projects with limited resources.

The decision-making process involved balancing the desire for cutting-edge models with the practical constraints of computing resources and project timelines. These considerations are crucial when choosing a model, as they impact the feasibility of conducting experiments and obtaining timely results.

One of the key advantages of BERT models is their efficiency in terms of resource usage and training time. Their relatively lightweight nature allows researchers to iterate experiments faster, which is especially valuable in dynamic research environments. This efficiency empowers researchers to explore a wide range of experiments and quickly adapt to new developments in the field.

The decision to choose BERT as the benchmark model was made after careful deliberation of these factors. It reflects a pragmatic approach, considering the model's historical significance, proven performance, and practical advantages. BERT was deemed well-suited for achieving the research objectives while respecting the constraints of resources and time.

A. M-BERT

The BERT model, proposed by the Google team in 2018 [28], is based on pre-training with large-scale unlabeled text data and subsequent fine-tuning for specific natural language processing tasks. It leverages a cascading Transformer encoder,

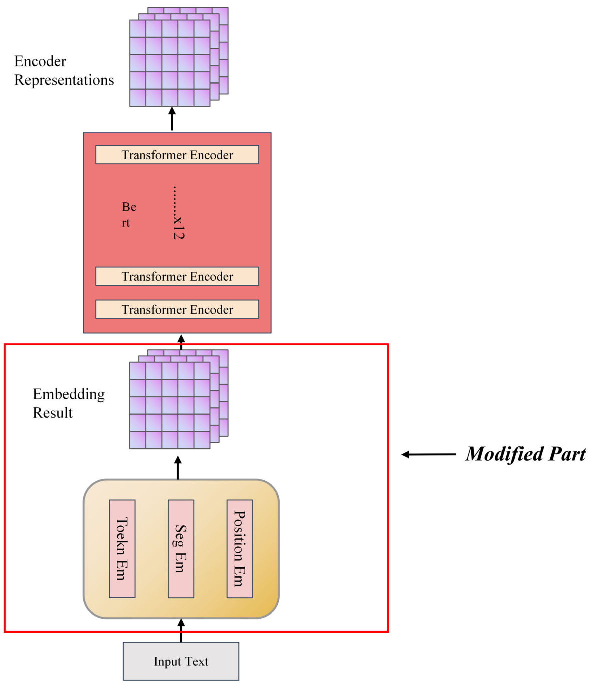

FIGURE 2. The architecture overview of BERT model. ***a. Token Em, Seg Em and Position Em represent the three Embedding layers of the original BERT model on the input text, which are word embedding, word segmentation record and position record respectively. b. Stack with encoder of Transformer to get a word vector with stronger characterization ability. c. The part marked in the red box in the figure is the part we modified for the BERT model, and the specific modification details will be shown in Figure 3.***

which excels in capturing bidirectional contextual information, making it well-suited for various language processing tasks [28], [29]. However, when applying the original BERT model to detect malicious URLs, several shortcomings are observed:

1) The original BERT model can only handle the original URL information and generate a word vector representation for all URLs.
2) It cannot process information from other types of models. For instance, if we want to input the homepage image of a webpage as a feature, the original BERT model cannot incorporate this feature information.
3) The BERT model cannot embed additional features of the website since it solely relies on the Token Embedding layer, which hampers the accuracy of website detection. However, other website features, such as IP address length, top-level do-main name, domain name length, and domain name characteristics, hold practical significance in detecting malicious websites.

To address these limitations, this paper proposes the M-BERT model. Its main structure, as illustrated in Figure 3, resembles the original BERT model. The innovation lies in the inclusion of a New Embedding layer after the Position Embedding layer that can embed all other relevant features. These features are transformed into a vector after passing

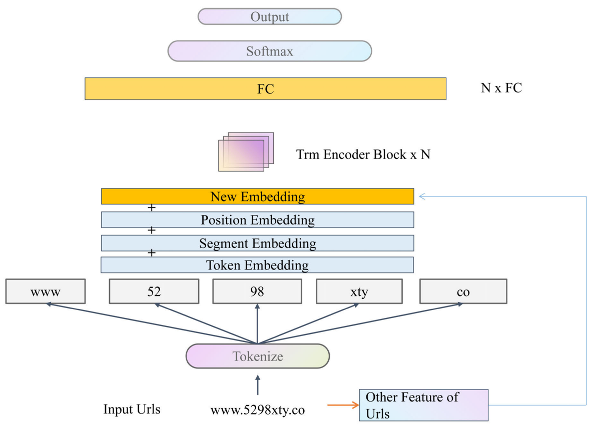

FIGURE 3. **The M-BERT model structure.** ***a. More features will be input to the New Embedding layer to better represent the site. b. After word segmentation, it is input to the Embedding layer, and other features of URL will be sent into the New Embedding Layer. Then, 12-layer Transformer Encoder is stacked to get a better network representation by stacking the embedded website features with other features through the Full-connected (FC) linear layer. c. After obtaining the final network representation, using Softmax to detect malicious web addresses. d. Other Feature of URLs: Other characteristics refer to the domain name of the URL, the length of the URL, the number of digits in the URL, etc.***

through the new embedding. The size of the new embedding layer remains consistent with that of the other three embedding layers in the original BERT model. During initialization, the New Embedding layer maps the website's other relevant features into high-dimensional vector representations, effectively representing each malicious website. Hence, the M-BERT model not only accepts website inputs but also incorporates other website features, enabling the consideration of various aspects such as IP address length, do-main name characteristics, etc. After embedding the website features and representations of other websites, 12-layer Transformer encoder blocks are stacked to obtain a more comprehensive vector representation.

At the terminal classification head of the model, we amalgamated these diverse embedding layers along the channel dimension, thereby effecting their integration. This strategic design enables the model to comprehensively synthesize information from various sources, ensuring more precise classification outcomes. The following simplifies the technical description of the above M-BERT:

1) **New Embedding Layer in M-BERT:** The New Embedding Layer in M-BERT enhances the original BERT model by incorporating URL-specific features like URL length, numerical content, and domain details. It enriches the model's understanding by merging linguistic context with diverse URL attributes.
2) **Role of the New Layer:** It integrates URL-specific details into the BERT model, providing a comprehensive view of URLs. This comprehensive representation strengthens the model's ability to identify malicious URLs accurately.
3) **M-BERT's Strength:** By fusing linguistic context with URL-specific features, M-BERT excels in comprehending various aspects of URLs. The added URL attributes empower the model to better detect malicious URLs.

IV. ALGORITHMIC FRAMEWORK

In harnessing the power of linguistic context and URL-specific features, M-BERT stands out in its holistic understanding of diverse URL aspects. This comprehensive representation lays the groundwork for a deeper exploration. Algorithm (1) delineates the step-by-step process of the M-BERT algorithm for malicious URL detection. Our experiments showcased M-BERT's superior performance compared to the original BERT model. Furthermore, leveraging different interpretability analysis algorithms shed light on our model's decision-making process, providing invaluable insights into the classification outcomes.

Algorithm 1 URL Classification Algorithm Based on M-BERT

```
1:  Input: URLs and features of URLs
2:  Output: Malicious URL or Non-Malicious URL
3:  Initialize The M-BERT Model
4:  Map URLs to Dataloader
5:  Input batch URLs to M-BERT
6:  for $k = 1 \ldots K$ do
7:      Input URLs to Token Embedding
8:      Input other features of URLs to New Embedding
9:  end for
10: Train All Embeddings to get new representation of URLs
```

The M-BERT model follows a URL classification algorithm, which can be described as follows:

1) First, input the URL and other relevant features of the URL, and output whether the URL is malicious.
2) The M-BERT model is initialized, and the URL is mapped as a dataloader. The input URL, along with other relevant features, is fed into the M-BERT model for correlation embedding. The embedding process captures the relationships between different components of the URL and its associated features.
3) The Transformer encoder block is then employed for cyclic stacking. This process involves repeatedly applying the encoder block to the embedded representations, allowing the model to learn hierarchical representations that capture both local and global contextual information.
4) Training is conducted by adjusting all the embedding parameters. This process involves fine-tuning the model structure of M-BERT to optimize its performance for URL classification. The training phase aims to minimize the classification loss and improve the model's ability to accurately distinguish between malicious and benign URLs.

5) Finally, the trained M-BERT model is used to classify new URLs by inputting them into the model and obtaining the classification result based on the learned representations and parameters.

Through this process, the M-BERT model leverages the URL and its relevant features to make predictions regarding the maliciousness of the URL, offering an effective solution for URL classification tasks.

In the M-BERT model, when considering four layers of embedding while keeping the encoder parameters unchanged, the computational complexity of the algorithm can be outlined as follows:

1) Embedding Stage:
During the embedding stage, each token in the input sequence undergoes transformation into an embedding vector. Assuming an input sequence length of N and an embedding dimension of d, the computational complexity of the embedding stage can be represented as $O(N \times d)$. With four layers of embedding parameters, the cumulative computational complexity of the embedding stage would be $O(4 \times N \times d)$.
2) Encoder Stage:
The M-BERT model employs a Transformer encoder, which consists of self-attention mechanisms and feed-forward neural networks. Considering an input sequence length of N, an encoder with L layers, a hidden dimension of d, and h attention heads, the complexity of each self-attention mechanism is $O(N^2 \times d)$, while the complexity of each feed-forward neural network is $O(N \times d)$. Since the Transformer encoder comprises L layers, the overall computational complexity of the encoder stage can be expressed as $O(L \times (N^2 \times d + N \times d))$.

In summary, for the M-BERT model with four layers of embedding parameters (without altering the encoder parameters), the computational complexity of the algorithm can be approximated as $O(4 \times N \times d + L \times (N^2 \times d + N \times d))$.

In addressing the algorithm's scalability and performance across diverse operational environments, our integration strategy extends to embedding the model within a browser plugin. When a user initiates a web page visit through this plugin, our system leverages the trained model for detecting malicious URLs, presenting a visual output that aids in determining the site's safety status.

Furthermore, this implementation provides real-time feedback to users, enhancing their understanding of the URL's potential threat. By employing this approach, our algorithm seamlessly integrates into daily browsing activities, demonstrating its adaptability in practical scenarios. Additionally, the plugin's functionality showcases the adaptiveness of our algorithm in different operational contexts, highlighting its efficiency in assessing web safety across varied browsing environments. This real-time, user-centric approach serves as a tangible demonstration of the algorithm's robustness and adaptability.

V. EXPERIMENT

A. DATASET DESCRIPTION

In extending our algorithm's capabilities to diverse operational settings, such as embedding it within a browser plugin for real-time malicious URL detection, we've laid a foundation for enhanced user safety during web browsing. This seamless integration not only provides users with instant feedback on a site's safety status but also exemplifies the adaptability of our approach in practical scenarios. Now, transitioning to the dataset that fuels our algorithm's intelligence, the research acquired its dataset from the Zhejiang Mobile Innovation Research Institute, consisting of a total of 600,000 labeled URL samples. These URLs represent real, accessible websites. Notably, when accessing these URLs via web crawlers, a status code of 200 was returned, indicating successful data retrieval and ensuring its temporal relevance. Prior to conducting the analysis, a meticulous data cleaning and feature engineering procedure was executed. Subsequently, the dataset was partitioned into training and validation sets, following an 8:2 ratio. To model this dataset, a customized M-BERT model was developed. During network initialization, the M-BERT model was employed with input from website information, along with the embedding of supplementary website features. The model then underwent a classification process. Post model training, its performance underwent evaluation employing the validation set, which constituted 20% of the dataset. Through experimentation on the validation set, the study assessed the effectiveness of the proposed model. The evaluation primarily aimed to measure the model's performance in terms of classification accuracy and other pertinent metrics. This validation procedure provided valuable insights into the efficacy of the model and its potential applicability in real-world scenarios.

B. DATA PROCESSING

To address the issue of imbalanced category samples in the dataset, we implemented data augmentation and processing techniques. Firstly, unlabeled samples were excluded from the dataset. Since there was an abundance of non-malicious website samples, down-sampling was performed on the non-malicious websites. However, instead of using random sub-sampling, a hierarchical sub-sampling approach was adopted.

Hierarchical sub-sampling ensures that the feature distribution of the non-malicious URL samples in each subset remains consistent with that of the original dataset. This approach helps preserve the essential features of the dataset while balancing the representation of the non-malicious category. Figure 4 illustrates the flow chart depicting the steps involved in this process. By employing hierarchical sub-sampling, we aim to mitigate the potential issues related to over-fitting that may arise during model training. This data processing technique ensures a more balanced and representative dataset, allowing the model to learn effectively and generalize well across different categories.

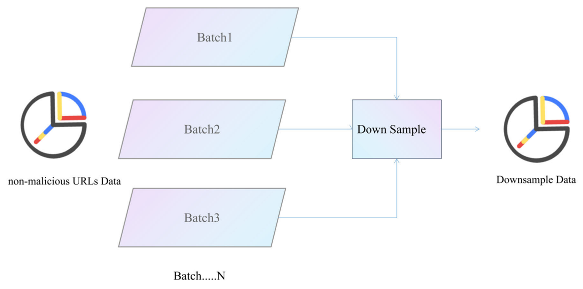

FIGURE 4. The flow chart of hierarchical sub-sampling algorithm.

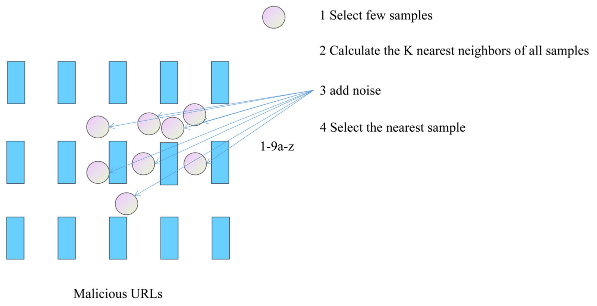
1 Select few samples
2 Calculate the K nearest neighbors of all samples
3 add noise
4 Select the nearest sample
1-9a-z
Malicious URLs

FIGURE 5. The general process of SMOTE.

After the down-sampling, considering that the malicious URL samples are very few, the number of malicious URLs is increased by the method of up-sampling. SMOTE is an up-sampling method based on generating synthetic samples, by interpolating between a few class samples to increase the number of malicious URL samples in the dataset [30]. However, when using SMOTE, you need to make sure that the resultant sample is not too close to the original sample, or you may cause problems with overfit-ting. In order to avoid this situation, noise information is added to the sample generation, and SMOTE flow chart for generating the malicious URL sample is shown in Figure 5.

In SMOTE algorithm, the process of generating synthetic samples for the minority class involves the following steps:

1) For each sample x belonging to the minority class, the Euclidean distance is calculated between x and all other samples in the minority class set. This distance calculation helps identify the K-nearest neighbors of x.
2) A sampling proportion is determined based on the class imbalance ratio in the dataset. This proportion determines the sampling ratio N. For each minority sample x, N samples are randomly selected from its K-nearest neighbors. Let's assume that the nearest neighbor selected is denoted as o.
3) For each randomly selected nearest neighbor o, a new synthetic sample is constructed by combining x and o according to the following Equation (1):

$$o(new) = o + rand(0, 1) \times (x - o). \tag{1}$$

In Equation (1), *rand* (0, 1) is a random value between 0 and 1, and $(x - o)$ represents the difference vector between o and x. By multiplying the difference vector with the random number and adding it to x, a new sample is generated that lies on the line connecting x and o.

This process is repeated for each minority sample x and its selected nearest neighbors, resulting in the generation of multiple synthetic samples to augment the minority class. By introducing variations through the random number and the difference vector, the synthetic samples exhibit diversity while preserving the characteristics of the minority class.

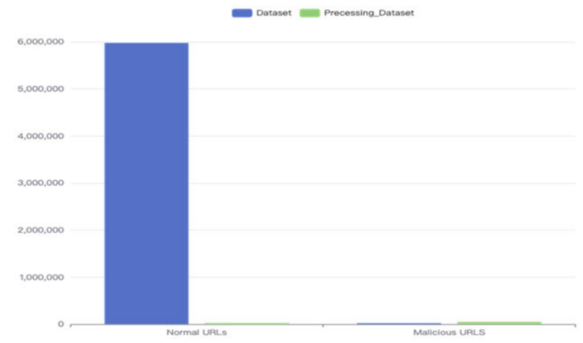

FIGURE 6. The visualization of dataset quantity statistics.

C. DATA VISUALIZATION

After performing the up-sampling and down-sampling operations, the entire dataset is visualized to gain a comprehensive understanding of its composition and characteristics. The Figure 6 provides a graphical representation of the dataset after these operations.

Since the dataset primarily consists of website addresses and textual information, additional features are generated to enhance the available information. These generated features include the length of the website, domain name, and other relevant information. By incorporating these features, the dataset becomes more informative and enables the model to capture important characteristics and patterns associated with the websites.

VI. RESULTS

After performing critical operations like up-sampling and down-sampling, the dataset's visualization in Figure 6 sheds light on its composition and characteristics. This visual representation serves as a foundation for further insights. Given that the dataset primarily comprises website addresses and textual data, the addition of supplementary features, such as website length and domain information, significantly enriches its depth. These augmented attributes not only enhance the dataset's informativeness but also empower the model to discern crucial website characteristics and underlying patterns.

A. PERFORMANCE ASSESSMENT

Moving from data preparation to performance evaluation, assessing these models' efficacy on our dataset becomes paramount. The datasets were processed using the M-BERT

TABLE 1. **Comparisons of quantitative results of existed methods.**

Model	Data Source	Macro F1	Macro Recall	Macro Precision
M-BERT	**Our Dataset**	**94.00**	**94.20**	**94.42**
XGBoost	Our Dataset	76.52	76.68	77.12
LightBoost	Our Dataset	74.23	74.54	74.95
LSTM	Our Dataset	85.27	86.12	86.67
NB [9]	From [9]	-	39.7	64.6
LR [9]	From [9]	-	70.5	81.5
RF [9]	From [9]	-	80.5	86.9
SVM [10]	From [10]	90.18	90.23	92.29
KNN [10]	From [10]	86.64	85.74	87.56

[a] The dataset from [9] contains 4000 samples and the dataset from [10] contains over 110000 samples, our dataset containing 600000 samples.
[b] F1 Score was not utilized in models' evaluation from [9].

TABLE 2. **Comparisons of experimental results of different large language models and M-BERT.**

Model	Batch	Macro F1	Macro Recall	Macro Precision
M-BERT	**512**	**94.00**	**94.20**	**94.42**
BERT-Base	512	91.35	91.51	91.28
ERNIE-2.0-base-en	512	81.76	79.41	81.67
ERNIE-3.0-tiny-base-v2-zh	512	50.39	50.64	52.63
ERNIE-2.0-large-en	512	91.15	90.05	93.13
GPT2-en-small	512	91.72	91.82	91.92
convBERT-small	512	90.60	90.3	91.22
ERNIE-m-base	512	83.77	82.15	87.58
convBERT-m-small	512	88.32	88.82	90.30
convBERT-base	512	88.23	88.62	89.12
GPT2-medium-en	512	92.24	92.54	92.85
mpnet-base	512	87.65	87.83	88.35
fnet-base	512	88.45	88.83	89.24
xlnet-base-cased	512	89.21	89.56	90.23

model as well as other large language models. Since the data set categories are not well-balanced, traditional metrics such as Recall, F1, and Precision may not be suitable. Instead, we utilize metrics specifically designed for unbalanced data sets, such as Macro-F1, Macro-Recall, and Macro-Precision [31]. Taking macro-recall as an example, we calculate the recall rate (*Recall_i*) for each category *i*, and then average the recall rates of all categories to obtain the Macro Recall.

To ensure a fair comparison with proposed M-BERT model, we also include a large language model with a similar number of parameters for comparison, such as ERNIE model [32]. ERNIE model follows a similar principle to BERT, learning general vector representations through a substantial amount of unsupervised training. Table 2 and Figure 7 presents the results of different models, including M-BERT and the comparable large language models. Moreover, Figure 8 illustrates the visualized results of the training set loss based on M-BERT model.

Figure 8 demonstrates that the M-BERT model maintains relative stability throughout the training process. The loss curve indicates that M-BERT model has converged.

- Methodology

Macro-F1, Macro-Recall, and Macro-Precision are metrics used for evaluating classification models in a multi-class setting. Here are the definitions of these metrics:

1) Macro-F1:
F1 score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance. Macro-F1 calculates the F1 score for each class individually and then takes the average across all classes. It gives equal importance to each class, regardless of class imbalance.
Macro-F1 for class i:

$$\text{Macro\_F1}(i) = \frac{2 \times P(i) \times R(i)}{P(i) + R(i)} \tag{2}$$

Overall Macro-F1:

$$\text{Macro F1} = \frac{1}{N}\sum_{i=1}^{N} \text{Macro\_F1}(i) \tag{3}$$

2) Macro-Recall:
Recall measures the model's ability to correctly identify positive instances (true positives) out of all actual positive instances (true positives + false negatives). Macro-Recall calculates the recall for each class individually and then takes the average across all classes. It provides an overview of how well the model performs in capturing all positive instances across different classes. Macro-Recall for class i:

$$\text{Macro\_Recall}(i) = \frac{TP(i)}{TP(i) + FN(i)} \tag{4}$$

Overall Macro-Recall:

$$\text{Macro Recall} = \frac{1}{N}\sum_{i=1}^{N} \text{Macro\_Recall}(i) \tag{5}$$

3) Macro-Precision:
Precision measures the model's ability to correctly identify positive instances (true positives) out of all predicted positive instances (true positives + false positives). Macro-Precision calculates the precision for each class individually and then takes the average across all classes. It gives an understanding of the model's accuracy in predicting positive instances across different classes.
Macro-Precision for class i:

$$\text{Macro\_ Precision}(i) = \frac{TP(i)}{TP(i) + FP(i)} \tag{6}$$

Overall Macro-Precision:

$$\text{Macro Precision} = \frac{1}{N}\sum_{i=1}^{N} \text{Macro\_Precision}(i) \tag{7}$$

In these Equations mentioned above:

1) True Positives (TP): TP represents the number of instances that are correctly classified as positive by the model.

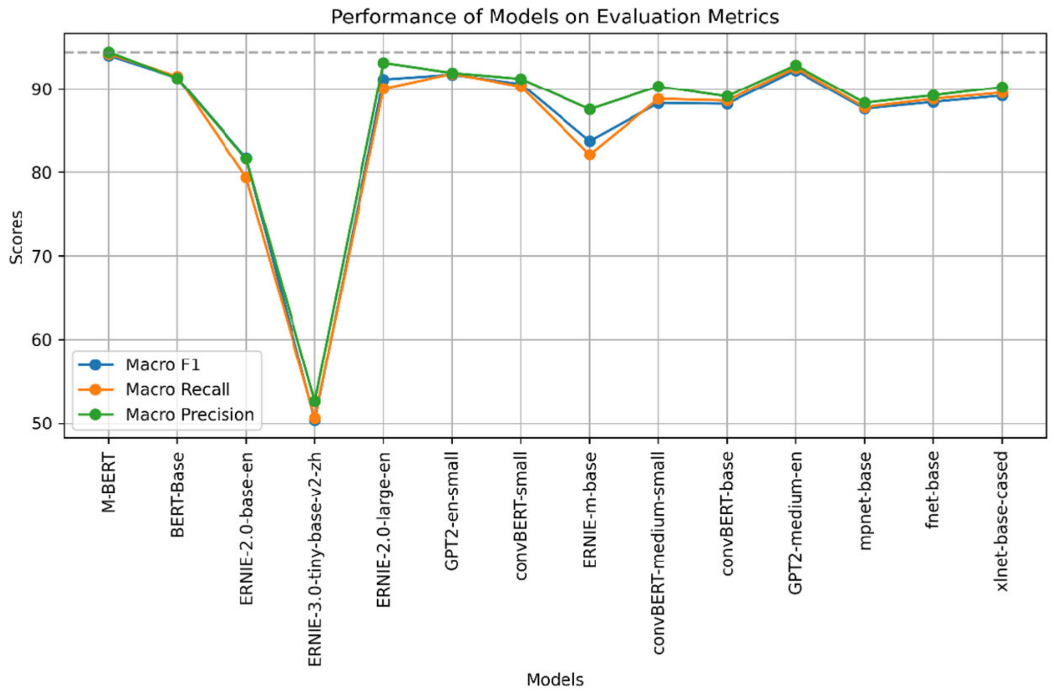

FIGURE 7. **Visualization of other LLMs models and m-bert on evaluation metrics.** ***a. Batchsize for training is 520.***

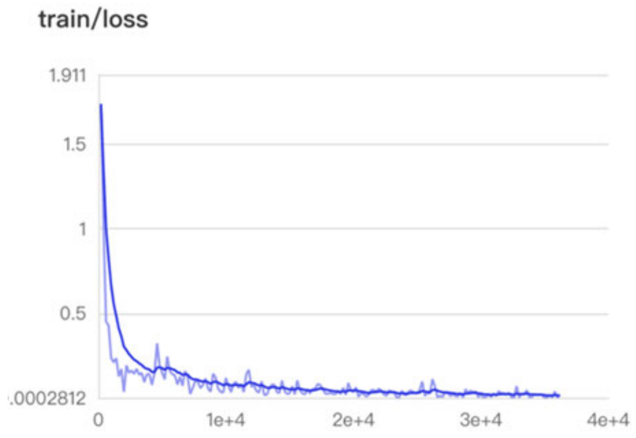

FIGURE 8. **The M-BERT model training loss.**

2) True Negatives (TN): TN represents the number of instances that are correctly classified as negative by the model.
3) False Positives (FP): FP represents the number of instances that are incorrectly classified as positive by the model.
4) False Negatives (FN): FN represents the number of instances that are incorrectly classified as negative by the model.

B. INTERPRETABILITY ANALYSIS

When it comes to deep learning models, researchers often face a significant challenge known as the "black box" problem. This implies that the internal workings of deep learning models are difficult for humans to comprehend, and the decision-making process of the model appears to lack interpretability. This issue becomes particularly prominent in many critical application domains, especially those closely related to security and decision-making. Fortunately, researchers have developed some methods to explain and understand the decision-making process of these black box models. Com-pared with the previous studies and other open-source large language models, the M-BERT model presented in this paper is of high accuracy, but it does not probe into the interior of the model and does not carry out interpretative analysis of the model. In order to explore the correctness and interpretability of this model, LIME and GrapShap algorithms are used to analyze M-BERT results.

LIME(Local Interpretable Model-agnostic Explanations), as a locally interpretable algorithm, is employed in this study to interpret predictions made by black box models. It operates by generating a set of new instances centered around a given instance and then explaining the model's predicted outcomes by elucidating the dissimilarities be-tween these new instances and the original instance [33].

TABLE 3. **Confusion matrix of classification.**

		Predicted	
		Positive (P)	**Negative (N)**
Actual	**Positive (P)**	TP	FP
	Negative (N)	FN	TN

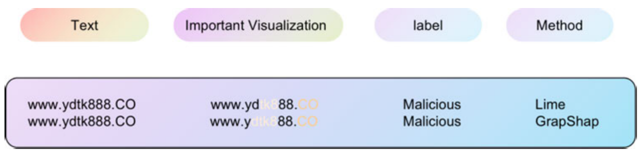

FIGURE 9. The interpretability results based on LIME & GrapShap. *a.lime: Constructs a locally linear model around a given predicted sample that can explain the model's decisions. From the lime analysis results in the above figure, it can be seen that the CO feature is highlighted, and it is precisely because of these features that the model judges that this URL is a malicious URL. b:GrapShap: Build a graph structure around a given prediction sample, and then calculate the contribution of each node in the prediction result. Similar to lime, its character features of interest are also CO.The results of the interpretability analysis, employing LIME and GrapShap. The highlighted portions represent the specific aspects that the M-BERT model emphasizes and focuses on during the classification process. These highlighted areas correspond to the features or patterns that have the most significant influence on the model's predictions.*

TABLE 4. Results of the ablation validation experiment.

Model	Dataset	Macro F1	Macro Recall	Macro Precision
M-BERT	Dataset-A	**94.00**	**94.20**	**94.42**
M-BERT	Dataset-B	**93.43**	**93.56**	**93.58**
BERT	Dataset-A	91.30	91.50	91.2
BERT	Dataset-B	89.71	90.33	90.23
GPT2-en	Dataset-A	91.72	91.84	91.96
GPT2-en	Dataset-B	90.20	90.60	91.12

On the other hand, GrapShap is a globally interpretable algorithm based on a graphical model, specifically designed to interpret the prediction results of deep neural networks [34]. It achieves this by transforming the neural network into a graph model and calculating the contribution of each node's activation value to the overall prediction result, thus providing an explanation of the model's prediction.

By combining the two methods mentioned above, a more comprehensive under-standing of how deep learning models operate in the context of malicious URL detection can be achieved. This approach provides valuable insights from an interpretability perspective, facilitating an evaluation of the model's decisions within this paper. Not only does this enhance our understanding of the model's behavior within the paper, but it also contributes to the improvement and optimization of the model, thereby increasing its reliability and trustworthiness in practical applications. In the following sections of this paper, we will provide a detailed explanation of how to employ these methods for interpretability analysis, aiming to enhance the transparency and interpretability of the model. These methods not only aid in meeting the requirements for interpretability but also enhance the credibility and utility of deep learning models across various application domains.

Notably, this interpretability analysis provides valuable insights into the inner workings of the M-BERT model, allowing researchers and practitioners to gain a deep-er understanding of its decision-making process and justify its effectiveness in classifying malicious URLs.

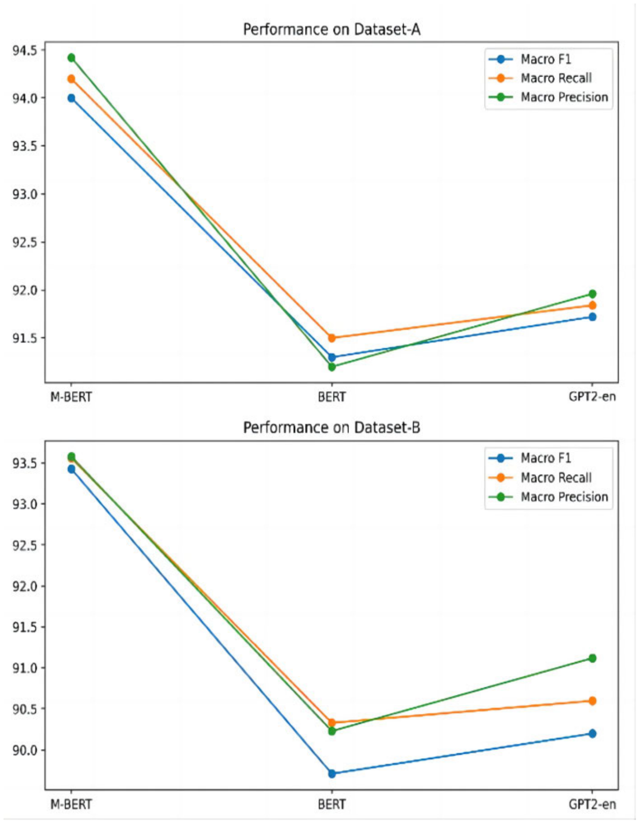

FIGURE 10. Visualization of the ablation validation.

C. ABLATION VALIDATION

To address the potential issue of domain name separation affecting the performance of the M-BERT model, an experiment was conducted to mitigate this effect. In this experiment, a copy of the original dataset was created, and the domain name in-formation of the websites was extracted using ''urlextract'' and then shuffled. The M-BERT model was then retrained and evaluated on this modified dataset, referred to as Dataset-B. The original dataset is denoted as Dataset-A.

Table 4 presents the results of the ablation validation experiment. It can be observed that the impact of domain name shuffling on the three selected models is not as significant as that of Dataset-A, although there is still a noticeable difference. These findings suggest that the performance of the experiment is not significantly affected by domain name word segmentation. Therefore, even if word segmentation is applied to the domain name, the experiment results are expected to remain reliable and valid.

The above contents have presented a comprehensive exploration of data processing, performance evaluation, methodology, and interpretability analysis in the context of utilizing the M-BERT model for malicious URL detection. These segments converge to illustrate the intricate procedure involved in employing the M-BERT model for such detection. The data processing segment delineates pivotal dataset operations aimed at enhancing model information. Meanwhile, the performance evaluation section provides a holistic comparison of multiple models' performance. Additionally, both the methodology and interpretability analysis sections underscore the significance of elucidating model decisions and enhancing model transparency. In essence, these findings validate the M-BERT model's high accuracy and robustness in malicious URL detection. In-depth evaluation and explication of the model afford a better understanding of its operational principles, bolstering trust and reliability in its decision-making process. Despite potential impacts on domain name segmentation, the model exhibits consistent and robust performance.

VII. CONCLUSION AND FUTURE

In this paper, an improved M-BERT model based on BERT is proposed for the detection of malicious web addresses, yielding promising results. Existing methods, such as black and white list detection, rely on extensive databases to identify malicious URLs and have certain limitations. Malicious URL detection based on CNN and LSTM struggles to extract meaningful representations of malicious URLs and fails to keep up with the rapidly evolving nature of these addresses. With the release of GPT-3 [35], the impressive generalization capabilities and semantic understanding of large language models have become evident [36], [37]. Thus, the current and future utilization of large language models to address tasks previously handled by traditional machine learning and deep learning approaches is inevitable.

Building upon previous research, we employ various language models to detect malicious websites, with the M-BERT model demonstrating the best performance among them. Our major contribution lies in the redesign of the BERT model specifically for malicious website detection, resulting in higher accuracy compared to the original BERT model. In addition, unlike other researchers, we analyze the interpretability of our proposed M-BERT model, shedding light on its reliability and unveiling the black box of deep learning. For potential limitations or scenarios affecting model performance, several considerations exist. Firstly, the ability of the model to correctly identify a URL when provided solely with URL information, lacking additional data, is a critical concern. This scenario might lead to suboptimal model performance due to the absence of comprehensive information. Therefore, further exploration is necessary to assess the model's robustness and performance when dealing exclusively with URL information. Secondly, the impact of model parameterization is another significant limiting factor. In situations where resources are constrained or rapid responses are required, extensive model parameterization may lead to decreased performance or latency. Hence, attention must be directed towards understanding the practical implications of model parameterization on performance and application, seeking solutions to optimize the model's behavior in resource-limited environments.

For the future work, we envision that by utilizing the M-BERT model and other large language models, we can more accurately detect and prevent malicious web addresses, thereby safeguarding users' network security. This study serves as a testament to the potential of harnessing cutting-edge language models within the domain of network security. However, it is crucial to acknowledge that future research should address multifaceted challenges and limitations linked to the practical implementation of these models, including issues related to scalability and privacy.

In the context of the M-BERT model's future, addressing scalability challenges stands as a paramount priority. This entails pivotal strategies such as parameter simplification through techniques like parameter pruning and quantization, effectively reducing model complexity. In practical applications, we acknowledged that the parameter volume of the M-BERT model is substantial, potentially impacting browser responsiveness under resource constraints. To mitigate this issue, we contemplate adopting methods to optimize the model's parameter volume, such as parameter pruning and quantized inference, to reduce the model's complexity. This strategy aims to enhance the model's response speed under resource-limited conditions while ensuring its effectiveness. We will explore additional solutions to further refine the model's behavior and deliver an enhanced user experience in real-world applications. Simultaneously, we acknowledge the constraints posed by the excessive parameterization of the M-Bert model, particularly in scenarios requiring rapid browser responsiveness. Additionally, adopting an incremental training strategy facilitates efficient model updates, ultimately minimizing training time and resource consumption. The incorporation of model parallelism not only boosts inference speed but also enables parallel processing across multiple computing nodes. For scenarios involving edge computing, optimizing the model to adapt to resource constraints while preserving detection accuracy remains crucial.

For the future investigation into model scalability, further exploration will focus on the enhanced capabilities of the M-BERT model in handling multimodal data. In addition to textual data, the integration of image information with textual data is being considered to enhance the detection efficacy of malicious URLs. However, this data fusion process raises privacy concerns. The anticipated research direction inclines towards the utilization of federated learning, a method allowing users to upload data to servers for analysis without actually transmitting sensitive user information. This data processing approach aims to safeguard user privacy while effectively addressing the challenges posed by multimodal data. This direction stands as one of the primary focal points for our future research endeavors.

The multifaceted ethical considerations intertwined within the realm of malicious website detection encompass various critical facets. This encompasses the imperative use of data desensitization and anonymization techniques to safeguard user identities and sensitive in-formation. Robust access control and permission management mechanisms, encompassing access restrictions for authorized personnel and meticulous access log maintenance, are indispensable.

Moreover, data encryption during both transmission and storage serves as a pivotal bulwark against unauthorized access and data leakage. A well-defined data retention policy specifying retention duration and purposes minimizes potential privacy risks. Lastly, securing explicit informed consent from users remains an essential ethical prerequisite, necessitating transparent communication of information and privacy policies. Collectively, these measures not only ensure robust data privacy and compliance with regulations but also uphold the model's legality and credibility. Furthermore, transparency and accountability emerge as imperatives, necessitating clear elucidation of the operational procedures of the detection model. This step ensures accuracy and fairness while mitigating the risk of biased or unjust outcomes. Moreover, preventing the potential misuse of detection technologies is essential. This involves averting undue constraints on content review or impingements on freedom of speech, thereby safeguarding information freedom and upholding individual rights. These ethical concerns will be central to future research endeavors in the domain of malicious website detection.

Although the challenges mentioned above for the future's deployment are still in the research and development stage, the remarkable results achieved by the M-BERT model underscore the immense potential of large-scale language models in addressing various cybersecurity challenges.

REFERENCES

[1] D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL detection using machine learning: A survey," 2017, *arXiv:1701.07179*.

[2] P. Shi, X. Yao, S. He, and B. Cui, "Malicious URL detection with feature extraction based on machine learning," *Int. J. High Perform. Comput. Netw.*, vol. 12, no. 2, p. 166, 2018.

[3] C. Johnson, B. Khadka, R. B. Basnet, and T. Doleck, "Towards detecting and classifying malicious URLs using deep learning," *J. Wireless Mobile Netw. Ubiquitous Comput. Dependable Appl.*, vol. 11, no. 4, pp. 31–48, 2020.

[4] J. Saxe and K. Berlin, "EXpose: A character-level convolutional neural network with embeddings for detecting malicious URLs, file paths and registry keys," 2017, *arXiv:1702.08568*.

[5] S. Brin, "The PageRank citation ranking: Bringing order to the web," in *Proc. ASIS*, 1998, pp. 161–172.

[6] M. Aljabri, H. S. Altamimi, S. A. Albelali, M. Al-Harbi, H. T. Alhuraib, N. K. Alotaibi, A. A. Alahmadi, F. Alhaidari, R. M. A. Mohammad, and K. Salah, "Detecting malicious URLs using machine learning techniques: Review and research directions," *IEEE Access*, vol. 10, pp. 121395–121417, 2022.

[7] M. Darling, G. Heileman, G. Gressel, A. Ashok, and P. Poornachandran, "A lexical approach for classifying malicious URLs," in *Proc. Int. Conf. High Perform. Comput. Simulation (HPCS)*, Jul. 2015, pp. 195–202.

[8] D. R. Patil and J. B. Patil, "Malicious URLs detection using decision tree classifiers and majority voting technique," *Cybern. Inf. Technol.*, vol. 18, no. 1, pp. 11–29, Mar. 2018.

[9] M. Weedon, D. Tsaptsinos, and J. Denholm-Price, "Random forest explorations for URL classification," in *Proc. Int. Conf. Cyber Situational Awareness, Data Anal. Assessment (Cyber SA)*, Jun. 2017, pp. 1–4.

[10] S. Abad, H. Gholamy, and M. Aslani, "Classification of malicious URLs using machine learning," *Sensors*, vol. 23, no. 18, p. 7760, Sep. 2023.

[11] T. Manyumwa, P. F. Chapita, H. Wu, and S. Ji, "Towards fighting cybercrime: Malicious URL attack type detection using multiclass classification," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 1813–1822.

[12] J. Cao, Q. Li, Y. Ji, Y. He, and D. Guo, "Detection of forwarding-based malicious URLs in online social networks," *Int. J. Parallel Program.*, vol. 44, no. 1, pp. 163–180, Feb. 2016.

[13] Z. Wang, X. Ren, S. Li, B. Wang, J. Zhang, and T. Yang, "A malicious URL detection model based on convolutional neural network," *Secur. Commun. Netw.*, vol. 2021, pp. 1–12, May 2021.

[14] J. Jiang, J. Chen, K. K. R. Choo, C. Liu, K. Liu, M. Yu, and Y. Wang, "A deep learning based online malicious URL and DNS detection scheme," in *Proc. 13th Int. Conf. Secur. Privacy Commun. Network SecureComm*, Niagara Falls, ON, Canada. Cham, Switzerland: Springer, Oct. 2017, pp. 438–448.

[15] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "URLNet: Learning a URL representation with deep learning for malicious URL detection," 2018, *arXiv:1802.03162*.

[16] W. Yang, W. Zuo, and B. Cui, "Detecting malicious URLs via a keyword-based convolutional gated-recurrent-unit neural network," *IEEE Access*, vol. 7, pp. 29891–29900, 2019.

[17] T. Wu, Y. Xi, M. Wang, and Z. Zhao, "Classification of malicious URLs by CNN model based on genetic algorithm," *Appl. Sci.*, vol. 12, no. 23, p. 12030, Nov. 2022.

[18] P. Maneriker, J. W. Stokes, E. G. Lazo, D. Carutasu, F. Tajaddodianfar, and A. Gururajan, "URLTran: Improving phishing URL detection using transformers," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Nov. 2021, pp. 197–204.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[20] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao, "Learning deep transformer models for machine translation," 2019, *arXiv:1906.01787*.

[21] H. Zhang, H. Song, S. Li, M. Zhou, and D. Song, "A survey of controllable text generation using transformer-based pre-trained language models," *ACM Comput. Surv.*, vol. 56, no. 3, pp. 1–37, Mar. 2024.

[22] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019, *arXiv:1910.13461*.

[23] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. S. Dhillon, "Taming pretrained transformers for extreme multi-label text classification," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 3163–3171.

[24] T. Shao, Y. Guo, H. Chen, and Z. Hao, "Transformer-based neural network for answer selection in question answering," *IEEE Access*, vol. 7, pp. 26146–26156, 2019.

[25] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pretraining text encoders as discriminators rather than generators," 2020, *arXiv:2003.10555*.

[26] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[27] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[29] A. Yates, R. Nogueira, and J. Lin, "Pretrained transformers for text ranking: BERT and beyond," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 1154–1156.

[30] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[31] S. Wang and X. Yao, "Multiclass imbalance problems: Analysis and potential solutions," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 4, pp. 1119–1130, Aug. 2012.

[32] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: Enhanced language representation with informative entities," 2019, *arXiv:1905.07129*.

[33] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?' Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1135–1144.

[34] A. Perotti, P. Bajardi, F. Bonchi, and A. Panisson, "Explaining identity-aware graph classifiers through the language of motifs," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2023, pp. 1–8.

[35] T. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1877–1901.

[36] C. Wu, S. Yin, W. Qi, X. Wang, Z. Tang, and N. Duan, "Visual ChatGPT: Talking, drawing and editing with visual foundation models," 2023, *arXiv:2303.04671*.

[37] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

BOYANG YU is currently pursuing the bachelor's degree in engineering. He is a Senior Student with the College of Electronics and Information, Southwest Minzu University, China. His research interests include computer vision, artificial intelligence, and machine learning. He has already made some contributions to the field, with three papers published at IEEE conferences.

FEI TANG is currently pursuing the degree with the College of Electronic and Information, Southwest Minzu University, China. He has published some articles at the CVIDL Conference and CNKI. His research interests include the areas of large language models, artificial intelligence, and recommended systems.

DAJI ERGU received the Ph.D. degree from the School of Management and Economics, University of Electronic Science and Technology of China, Chengdu, China, in 2014. He is currently a Professor with Southwest Minzu University, where he also serves as the Vice Chancellor. He has published over 40 journal articles and conference papers, such as *European Journal of Operational Research*, *Computers and Operations Research*, *Annals of Operations Research*, *Journal of the Operational Research Society*, and *Applied Soft Computing*. His current research interests include multicriteria decision-making, data mining, machine learning, and emergency management.

RUI ZENG received the Ph.D. degree from the Guangzhou University of Traditional Chinese Medicine, in 2007. He is currently the Deputy Dean of the School of Pharmacy, Southwest Minzu University. He has published multiple high-level articles as the first author or corresponding author. His research interests include drug quality evaluation, quantitative analysis, and drug literature analysis with artificial intelligence.

BO MA received the Ph.D. degree in atomic and molecular physics from Sichuan University, China. He is currently an Assistant Professor with the College of Electronic and Information, Southwest Minzu University, China. He is participating in the EDA Software Localization Project of the 13th Research Institute of China Electronics Technology Group Corporation. He has participated in several research projects, such as the National Natural Science Foundation Project, the Key Research and Development Project of Sichuan Province Science and Technology Project, the Chengdu Science and Technology Plan Project, and the Central University Special Fund. He has published nine academic articles, which are included by SCI or EI. His research interests include computer software, machine learning, and molecular dynamics simulation.

FANGYAO LIU received the Ph.D. degree in information technology from the University of Nebraska at Omaha, USA. He is currently an Assistant Professor with the College of Electronic and Information, Southwest Minzu University, China. He has published more than 20 articles in *International Journal of Computers Communications and Control*, *Journal of Urban Planning and Development*, *Journal of Software*, *Journal of Asian Development*, *Journal of Contemporary Management*, *Procedia Computer Science*, and several IEEE conferences. His research interests include data mining, artificial intelligence, and statistics.