electronics

MDPI

Article

PhiShield: An AI-Based Personalized Anti-Spam Solution with Third-Party Integration

Hyunsol Mun [1,†], Jeeeun Park [1,†], Yeonhee Kim [2], Boeun Kim [1] and Jongkil Kim [1,*]

[1] Department of Cybersecurity, Ewha Womans University, Seoul 03760, Republic of Korea; tranquil@ewha.ac.kr (H.M.); lucykorea414@ewha.ac.kr (J.P.); bekim@ewha.ac.kr (B.K.)
[2] Department of Artificial Intelligence Convergence, Ewha Womans University, Seoul 03760, Republic of Korea; heedong@ewha.ac.kr
* Correspondence: jongkil@ewha.ac.kr
† These authors contributed equally to this work.

Abstract: In this paper, we present PhiShield, which is a spam filter system designed to offer real-time email collection and analysis at the end node. Before our work, most existing spam detection systems focused more on detection accuracy rather than usability and privacy. PhiShield is introduced to enhance both of these features by precisely choosing the deployment location where it achieves personalization and proactive defense. The PhiShield system is designed to allow enhanced compatibility and proactive phishing prevention for users. Phishield is implemented as a browser extension and is compatible with third-party email services such as Gmail. As it is implemented as a browser extension, it assesses emails before a user clicks on them. It offers proactive prevention for users by showing a personalized report, not the content of the phishing email, when a phishing email is detected. Therefore, it provides users with transparency surrounding phishing mechanisms and helps them mitigate phishing risks in practice. We test various locally trained Artificial Intelligence (AI)-based detection models and show that a Long Short-Term Memory (LSTM) model is suitable for practical phishing email detection (>98% accuracy rate) with a reasonable training cost. This means that an organization or user can develop their own private detection rules and supplementarily use the private rules in addition to the third-party email service. In this paper, we implement PhiShield to show the scalability and practicality of our solution and provide a performance evaluation of approximately 300,000 emails from various sources.

Keywords: anti-spam; AI-based spam detection; LSTM; phishing prevention

check for updates

Academic Editor: Aryya Gangopadhyay

Received: 12 March 2025
Revised: 9 April 2025
Accepted: 11 April 2025
Published: 13 April 2025

Citation: Mun, H.; Park, J.; Kim, Y.; Kim, B.; Kim, J. PhiShield: An AI-Based Personalized Anti-Spam Solution with Third-Party Integration. *Electronics* **2025**, *14*, 1581. https://doi.org/10.3390/electronics14081581

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Phishing is a serious cybersecurity threat that significantly impacts global business. According to aag-it.com [1], 3.4 billion spam emails are sent daily, and 91% of cyber attacks are initiated with phishing attacks. Phishing emails deceive recipients into believing that they originate from trusted sources, prompting them to provide sensitive information or click malicious links that help more sophisticated attacks, such as downloading malware. These attacks pose significant threats to individuals and corporations, and their frequency has been rapidly increasing globally in recent years [2]. Phishing attacks primarily target financial data, personal information, or confidential corporate data, with attackers impersonating official institutions or trusted sources to elicit sensitive details from unsuspecting users. As phishing techniques become more sophisticated, traditional signature-based detection systems struggle to counter evolving tactics effectively [3].

Most existing phishing detection systems rely on signature-based rules, such as blacklists, to identify URLs or email addresses linked to phishing attacks. However, this signature-based approach has limitations; the attackers can easily bypass the signature-based detection systems using modified URLs or newly registered domains. Phishing emails bypassing blacklists can reach recipients and this increases the risk of users opening malicious emails or clicking dangerous links, which can lead to financial losses or data breaches [4].

There is a growing need for more intelligent and efficient phishing email detection systems. These systems must go beyond fixed signature-based rules to analyze various elements of an email comprehensively. Artificial intelligence (AI)-based detection is one of the promising solutions to meet these demands. AI algorithms can detect unknown patterns and anomalous behavior representative of attackers by learning from datasets, providing better adaptability to real-time phishing techniques. This enhances the accuracy of phishing email detection and reduces the damage caused by phishing.

Moreover, traditional phishing detection systems, often server-based, face challenges in detecting encrypted emails or spear-phishing attacks. In the case of encrypted emails, such as those using OpenPGP or Secure/Multipurpose Internet Mail Extensions (S/MIME), the server cannot analyze the email by content, rendering traditional detection methods ineffective. Cloud-based email services such as Gmail and Outlook also make detection systems totally blind as they are transmitted in an encrypted format using Transport Layer Security (TLS) encryption. Furthermore, server-based spam detection services often operate with generic rules, making them less effective against targeted spear-phishing attacks and unable to offer personalized services.

This paper presents PhiShield, a newly designed spam filter system that is compatible with third-party email services and supports enhanced privacy and proactive spam detection. The detailed contributions of PhiShield are as follows:

- Compatibility: By developing and deploying as a browser extension, PhiShield collects and analyzes emails in a user's browser. Therefore, it is compatible with third-party spam detection services provided by cloud-based email service providers such as Gmail and adds an extra layer of protection to those services.
- Privacy and Personalization: PhiShield allows a user (or a group of users) to build a private detection rule without compromising user privacy. It eliminates the need to share sensitive email data with a third party by locally training and deploying personalized AI models for the detection.
- Proactive Detection: PhiShield provides proactive protection by analyzing emails in real-time before users access them. It delivers detailed analytical reports explaining phishing detection results, significantly enhancing user awareness and security practices.

In this paper, we present the system architecture of PhiShield and evaluate its performance. In particular, we locally train multiple AI-based spam filter models and show that the LSTM model can be deployed within PhiShield and exhibit good performance (>98% accuracy) in spam detection.

Related Work

The field of phishing email detection has been actively researched in recent years, utilizing various AI models [5–7]. Studies using Long Short-Term Memory (LSTM) [8,9] demonstrate that it can effectively detect phishing emails by learning the sequence of words in the email body and remembering recurring patterns in email data. However, such LSTM-based detection systems do not provide detailed explanations for the detection results, making it difficult for users to identify how the phishing attempt occurred. PhiShield provides detailed analytical reports on the detection process and results, helping users

clearly understand the findings. Additionally, it is designed to analyze emails in real-time, allowing users to immediately check for phishing attempts upon receiving an email without sending data to a server.

SpamFilter-ChromeExtension [10] was introduced as a browser extension that works universally with third-party email services. However, SpamFilter-ChromeExtension uses Naive Bayes classifiers to detect spam emails. Naive Bayes classifiers are a well-known method for spam detection. This method computes the probability of spam by measuring how much the words in the mail are correlated with spam and non-spam emails. However, according to [11], its accuracy is inferior to LSTM in classification. Moreover, SpamFilter-ChromeExtension requires a user to access emails to filter spam emails. It does not provide proactive protection for users from spam emails.

Furthermore, recent studies that use large language models (LLMs) for phishing detection [12] have also gained momentum. For example, ChatSpamDetector uses LLMs like GPT, LLaMA, and Gemini to detect phishing emails. When provided with the email's content and role as input, the LLM assesses whether the email is phishing-related and returns a confidence score on phishing risk, relevant brand information, and an explanation of the results. Although ChatSpamDetector can explain the detection results to users, it lacks discussion on the real-time collection and processing of email data. Moreover, it does not guarantee users' privacy, as the content of the email is shared with the LLM service provider.

The key differences among these systems, including compatibility, privacy and personalization, proactive detection, and AI model types, are summarized in Table 1. Compatibility implies that the proposed system is compatible with third-party email solutions like Gmail. Privacy and personalization relates whether the proposed detection rule can be trained locally without leaking the email data to the external organization. Proactive detection implies that users can obtain a result of spam detection before they see the content of emails. As shown in the table, our system achieves all those important properties at the same time to the best of our knowledge.

Table 1. Comparison of phishing detection systems.

System	Compatibility	Privacy and Personalization	Proactive Detection	AI Model
LSTM based phishing detection [8,9]	×	×	×	LSTM
SpamFilter-ChromeExtension [10]	◯	△	×	Naive Bayes
ChatSpamDetector [12]	◯	×	×	LLM
PhiShield (Ours)	◯	◯	◯	LSTM

(◯ indicates full support, △ indicates partial support, and × indicates either no support or lack of sufficient information in the paper).

2. Materials and Methods

2.1. System Overview

In a typical email system, a sender generates an email and sends it through the sender's mail server. Then, the sender's mail server delivers the email to a receiver via the receiver's mail server. However, in our proposed protocol, we locate the private spam filter system PhiShield between the receiver's mail server and the receiver in order to protect the receiver from spam emails.

In particular, we observed that the location of PhiShield is important because email is usually delivered to a receiver via an end-to-end encryption channel. The typical spam filter system or intrusion detection system (even host-based intrusion detection system) does not work as they cannot see the content of such mail. However, in our proposed

system, we successfully deploy the spam filter system by implementing it as an extension of a browser.

Figure 1 depicts PhiShield's location by showing the flow of email. In the figure, an email sent by the sender is delivered via mail servers. When the mail is delivered, it is not delivered directly to a receiver, but processed and analyzed by PhiShield in real time. The email is examined by PhiShield before a receiver clicks on the email in a browser. Then, the detection result will be delivered to the receiver. If the email sent was benevolent, this result will be the original mail item. Otherwise, it will be a report notifying a spam delivery. Therefore, it allows proactive protection from a spam email. This means that PhiShield affords the receiver a chance to review the email safely before opening it.

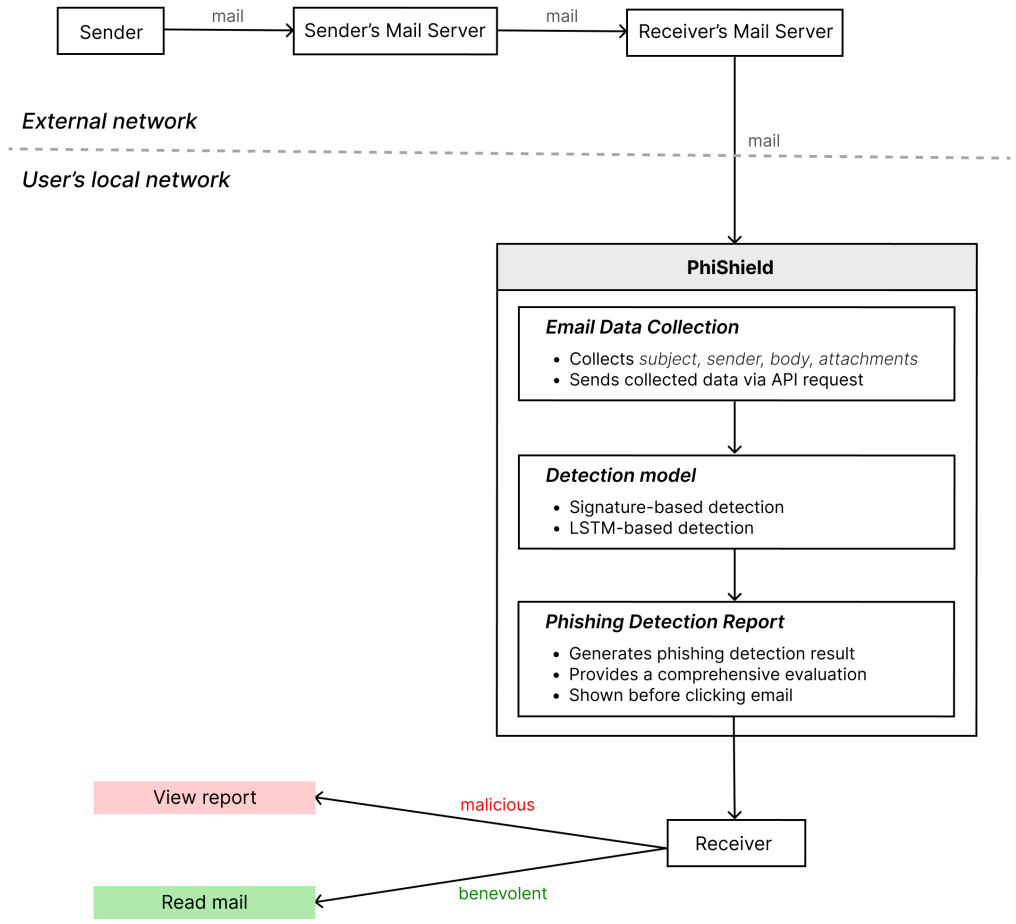


Figure 1. Flow of email processing and analysis in the PhiShield system.

System Architecture

We illustrate the detailed architecture of the proposed PhiShield system in Figure 2. PhiShield is designed with scalability and efficiency in mind. PhiShield consists of four key components: email data collection, detection report, Django Representational State Transfer (REST) application programming interfaces (API), and the AI-based phishing detection model. Among those components, the email data collection and detection report components are implemented in the browser extension of a user. The Django REST API and AI-based phishing detection model components are located on an external server called the API server. Each component operates independently while being interconnected to perform phishing detection tasks. This offers a flexible structure compatible with various email services.

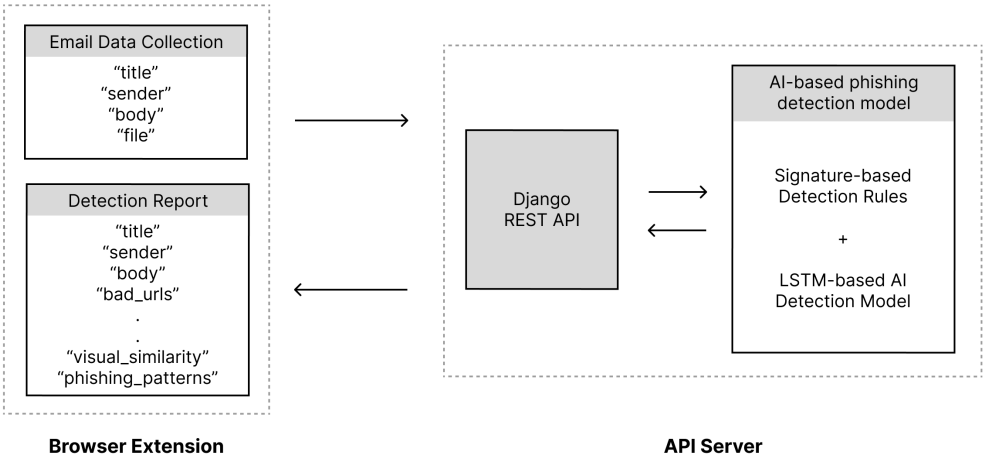


Figure 2. System architecture of PhiShield.

Browser extension: The browser extension serves as the front-end and is responsible for real-time email collection and detection result reporting when users access web-based email services such as Gmail.

- The email data collection uses APIs that a third-party email service provider provides and collects email data such as the subject, sender, body, and attachments before a user clicks on the email. After collecting all the information, it sends the data to the API server.
- The detection report component provides a review of the analysis results for users before opening the email. This report includes detailed explanations to help users easily identify if the email is spam or ham using the result that the API server sends.

API server: This server plays a central role in back-end data processing. It is deployed on a local server or a cloud server such as Amazon Web Services (AWS) Elastic Compute Cloud (EC2).

- The Django REST API is a management module receiving email data from the browser extension to perform a comprehensive analysis of the email body, URLs, sender information, and attachments. The server connects to an SQLite database for efficient detection rule management, and the API cooperates with the AI-based detection model to identify phishing threats in real time and generates reports for users based on the analysis.
- The AI-based phishing detection model: The core analytical engine of PhiShield consists of signature-based detection rules and an LSTM-based AI model. By combining both detection methods, PhiShield can identify various phishing attacks successfully. In PhiShield, signature-based detection that identifies suspicious terms or phrases in harmful URLs or email addresses evaluates the trustworthiness of URLs and senders and checks if attachments contain malicious content. The model also analyzes hidden links or abnormal patterns in emails to detect potential phishing attacks more accurately. It also uses an LSTM model. LSTM [12] is an AI algorithm that can effectively detect phishing emails by learning the sequence of words in the email body and remembering recurring patterns in the email data. This will help PhiShield detect the spam emails that the signature-based models miss. This hybrid approach significantly enhances the accuracy of phishing detection compared to the detection system that only uses a signature-based detection method.

The back-end architecture is designed to provide the system with flexibility for future updates of detection rules. Notably, it can easily expand by adding new detection items

or rules to the AI model without requiring redesign. Furthermore, the front-end's real-time email collection capability is designed with compatibility for various email services, ensuring the system's adaptability to new services or features. This enables PhiShield to operate reliably across diverse email platforms and service environments, maximizing the system's performance and efficiency.

2.2. Implementing the AI-Based Detection Model

We implement signature-based and AI-based detection models to identify various phishing attacks. In this study, signature-based detection identifies suspicious terms or phrases in harmful URLs or email addresses, evaluates the trustworthiness of URLs and senders, and checks attachments for malicious content. The model also includes a signature-based analysis of hidden links or abnormal patterns in emails to detect potential phishing attacks more accurately. In addition to those signature-based analyses, the AI-based phishing detection model is implemented as the core analytical engine of PhiShield. We use an LSTM model to check for spam mail. This improves the accuracy of phishing detection in the system.

2.2.1. Dataset Description

To evaluate the performance of our detection model, four distinct datasets were utilized as follows:

1. Dataset 1: Our first dataset, sourced from Kaggle-Phishing Email Detection https://www.kaggle.com/datasets/subhajournal/phishingemails (accessed on 10 April 2025) [13], contains approximately 18,600 emails, with 61% classified as legitimate emails and 31% as spam emails. The dataset captures the text body of emails, allowing for extensive analysis and classification through machine learning techniques.
2. Dataset 2: The second dataset [14] was compiled by Al-Subaiey et al. for the purpose of studying phishing email tactics. They collected emails from various sources to provide a comprehensive resource for phishing detection analysis. The dataset includes a total of approximately 82,500 emails, of which 42,891 are classified as spam and 39,595 as legitimate. As the emails were sourced from multiple origins, the dataset offers a diverse range of phishing email content and the contextual environment in which these emails were sent, making it an invaluable tool for enhancing the effectiveness of phishing detection models.
3. Dataset 3: The third dataset was obtained from the publicly available Large-scale Phishing Email Dataset https://zenodo.org/records/8339691 (accessed on 10 April 2025) [15] on Zenodo. It includes approximately 200,000 emails sourced from internationally recognized datasets such as CEAS-2008, TREC 2005/2007, Enron, Nazario, and Nigerian Fraud. Each of these datasets contains a variety of spam and phishing emails as well as legitimate emails. The emails are categorized by source and topic, enabling model training that reflects a wide range of phishing scenarios.
4. Dataset 4: The fourth dataset, spam.csv https://github.com/Apaulgithub/oibsip_taskno4/blob/main/spam.csv (accessed on 10 April 2025) [16] is provided by an open-source project and consists of approximately 5500 email messages. Each message is labeled as either 'ham' or 'spam'. Due to its simple structure, this dataset is well-suited for initial model testing and benchmarking purposes.

It should be noted that the AI-based detection model in PhiShield can be more personalized by using the emails within the personal mailbox. Specifically, emails exchanged within the user's organization can provide a valuable set of phishing examples, while phishing emails identified by the organization's filter serve as an additional source. Including

those emails in the dataset will enable PhiShield to develop a more tailored and effective phishing email detection filter.

2.2.2. Data Preprocessing and Tokenization

To extract model input from email content, several preprocessing steps were implemented. First, email text data were loaded using the Pandas library, with each email's content converted to string format and designated as the feature input x, while the email labels were assigned as target values y. Under this framework, all emails underwent two main preprocessing steps: tokenization and padding. For tokenization, the Tokenizer class was employed to convert the words in each email into unique integer indices. Subsequently, to standardize input sequences to a fixed length, padding was applied to accommodate varying email lengths. A maximum sequence length of 1000 words was established, whereby sequences shorter than 1000 words were padded with zeros at the end, while sequences exceeding this length were truncated to preserve only the first 1000 words. Following these preprocessing steps, all emails became compatible with the LSTM model as they were represented as integer-indexed sequences of uniform length.

2.2.3. LSTM Model Architecture

An LSTM network is a type of recurrent neural network designed to overcome the vanishing gradient problem in traditional Recurrent Neural Networks (RNN). Using memory cells with three key gating mechanisms (Forget Gate, Input Gate, and Output Gate), LSTM selectively retains or discards information over time [17], allowing it to model long-term dependencies in sequential data. The gating mechanisms work as follows:

- Forget Gate: Determines which information to discard:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{1}$$

- Input Gate: Decides which new information to store:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \quad \tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{2}$$

- Output Gate: Determines the information passed to the next hidden state:

$$h_t = o_t \odot \tanh(C_t), \quad o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3}$$

By leveraging these mechanisms, LSTM excels at capturing complex temporal patterns in applications like network traffic analysis and time-series anomaly detection.

The proposed LSTM model architecture in Figure 3 is designed to take vectorized text data as input and sequentially process them by utilizing multiple layers having specific roles as described below.

1. Embedding layer: The first layer in the model is the embedding layer, which transforms the input data's sequence length and unique word indices into dense vector representations. The 'input_dim' parameter is set to the total vocabulary size ('vocab_size'), and the 'output_dim' parameter defines the embedding dimension as 128, enabling the model to capture semantic similarities between words. The 'input_length' parameter denoting the maximum sequence length of the sentences ensures that the model handles input data in a fixed sequence length. This layer outputs data in the form '(batch size, sequence length, vector dimension)'.
2. Bidirectional LSTM layer: The bidirectional LSTM layer applies a bidirectional RNN to read the text data in both forward and backward directions. This layer uses 64 units and is set with 'return_sequences=True' to ensure that each time step's output

is passed to the following layer. As a result, the model can learn the contextual relationships of the data more comprehensively. The output of this layer is in the shape '(batch size, sequence length, 2×units)'.

3. Dropout layer: The dropout layer is used to prevent overfitting. A dropout rate of 0.5 is applied, randomly disabling some neurons during training to improve the model's generalization capabilities. This layer maintains the output dimension from the 'Bidirectional LSTM' layer.
4. Single-directional LSTM layer: A single-directional LSTM layer with 32 units processes the sequence. It is set with 'return_sequences=False', so only the last time step's output is passed to the next layer. This layer extracts the most significant time step information from the input sequence, summarizing it into features that contribute to phishing email predictions.
5. Dense layer: The final layer is the dense layer using a sigmoid activation function to output binary classification results, where phishing emails are predicted as 1 and legitimate emails as 0. This layer contains a single output node, and its output can be interpreted as a probability between 0 and 1.

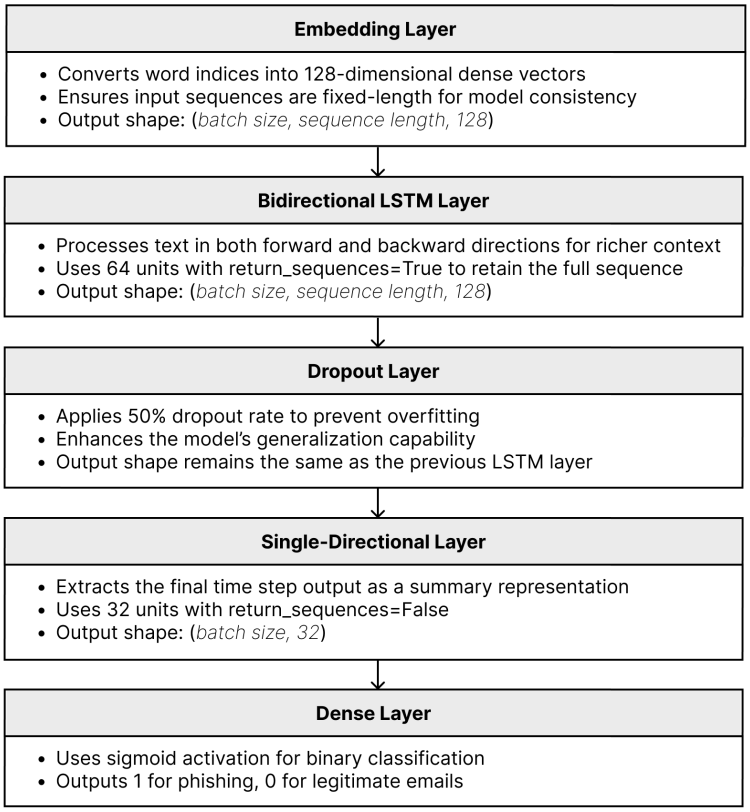


Figure 3. LSTM model architecture of PhiShield.

2.3. Signature-Based Spam Detection Model

The PhiShield system proposed in this study is implemented with scalability and flexibility in mind. The system is designed to collect emails in real-time from the front-end and analyze the data through the back-end, ensuring compatibility with various email services. When adding new detection rules or AI models, updates can be applied solely on the back-end, which increases scalability. This scalability lays the foundation for PhiShield to operate effectively across different email platforms, contributing to strengthened email security and safeguarding personal and corporate emails when commercialized.

In this subsection, we briefly introduce conventional signature-based detection rules that can be combined with an AI-based phishing detection model in the previous section. Table 2 presents 11 potential signature-based detection rules effectively identify various phishing attacks. These rules include URL and email address similarity checks, suspicious word and phrase detection, and verification of the safety of attachments. These 11 rules can be used as signature-based rules [18] to improve the efficiency of real-time detection. They can pre-screen spam emails by being applied before our AI-based detection method is used for the detection.

Table 2. Signature-based spam mail detection models and their implementation methods.

B	Name	Description	Implementation Method
1	Harmful URL Check [19,20]	Check whether the email's URL is on a blacklist	Compare with URL blacklist
2	Harmful Email Address Check [20]	Check whether the email address is on a blacklist	Compare with email blacklist
3	Suspicious Pattern Detection	Detect suspicious phishing words or phrases	Keyword detection using regular expressions
4	Attachment File Extension Check	Check the file extension for potential malicious content	Compare with a list of suspicious file extensions
5	URL Similarity Check [21]	Compare the URL with legitimate websites for similarity	Use SequenceMatcher to compare similarity
6	Email Address Similarity Check [21]	Compare the email address with legitimate addresses for similarity	Use SequenceMatcher to compare similarity
7	N-gram Based URL Similarity Check [22]	Compare the URL with legitimate websites using N-gram technique	Use TfidfVectorizer and cosine similarity for N-gram comparison
8	N-gram Based Email Address Similarity Check [22]	Compare the email address with legitimate addresses using N-gram technique	Use TfidfVectorizer and cosine similarity for N-gram comparison
9	Homoglyph Detection [23]	Check for the use of homoglyphs	Detection through homoglyph mapping
10	Behavior Pattern Analysis	Analyze suspicious behavior patterns such as clickbait or urgent requests	Analysis using regular expressions and HTML parsing
11	Hidden Text and Link Detection [24,25]	Analyze for the presence of hidden text or links	HTML/CSS analysis using BeautifulSoup

3. Results

3.1. Model Performance

In this study, a comparative analysis was performed in various experimental environments to evaluate the performance of AI-based phishing detection models. The experiments were conducted in an Ubuntu 22.04.5 LTS environment using an NVIDIA RTX 6000 GPU. To optimize training speed, the number of epochs was set to 10 and the batch size to 64. To prevent overfitting, Early Stopping was applied during the training of all models, excluding Naive Bayes.

These datasets were merged to form a unified data resource. For training and testing purposes, the combined dataset was split into a training set and a testing set in a ratio of 85:15 in our performance evaluation. The performance of the model was evaluated using standard classification metrics. Additionally, the training and inference times of each model were compared to verify their practicality. The results are summarized in Table 3 and Figure 4.

Table 3. Phishing detection model performance metrics.

Metric	LSTM100	LSTM500	LSTM1000	BERT [26,27]	FCNN [28]	Naive Bayes [29,30]
True Positives (TP)	27,238	27,234	27,488	27,394	23,054	21,905
False Positives (FP)	446	314	574	205	2,825	10,967
False Negatives (FN)	502	506	252	346	4686	5835
True Negatives (TN)	28,759	28,891	28,631	29,000	26,380	18,238
Precision	0.9839	0.9886	0.9795	0.9926	0.8908	0.6664
Recall	0.9819	0.9818	0.9909	0.9875	0.8311	0.7897
F1 Score	0.9829	0.9852	0.9852	0.9900	0.8599	0.7228
Test Accuracy	0.9834	0.9856	0.9855	0.9903	0.8681	0.7049
Training Time (s)	743.24	2005.22	2936.55	4639.72	227.11	1.11
Inference Time (s)	24.30	61.20	108.07	113.99	3.96	0.05

For the LSTM-based model, experiments were conducted with three different settings to analyze performance differences based on input text length. LSTM 100 used the first 100 words from the email body as input, LSTM 500 used 500 words, and LSTM 1000 used 1000 words.

To compare performance with the LSTM-based model, we also implement two well-known algorithms for spam detection: the Naive Bayesian model [29,30] and the model based on Fully Connected Neural Network (FCNN) [28], which we call the FCNN-based model in this paper.

The Naive Bayesian model is often used as a baseline for comparison. The Naive Bayesian model is a statistical classification method based on prior probabilities and word distribution, which determines phishing status by reflecting only the frequency of individual words without considering the order of data. In this study, the multinomialNB model, commonly used in text data-based classification, was employed.

We also tested the FCNN-based model as another baseline. The FCNN-based model does not learn sequential dependencies but instead processes input data as a fixed-length vector, making it computationally efficient. In this study, email content was converted into histograms and statistical features, which were then used as input. The model consisted of a dense layer with 512 neurons, followed by hidden layers with 256 and 128 neurons, and a final output layer utilizing a sigmoid activation function to classify phishing emails.

To further compare phishing detection performance, we also implemented the Bidirectional Encoder Representations from Transformers (BERT) model [26,27]. BERT is a pre-trained language model based on the Transformer architecture that captures bidirectional contextual information from input text. In this study, we used the pre-trained bert-base-uncased model. The full email content was tokenized using the BERT tokenizer, and the first 128 tokens were extracted and used as input to the model.

These results indicate that the LSTM-based models recorded a high performance of over 98% in Precision, Recall, F1 Score, and Accuracy, with the LSTM1000 model showing the highest performance. However, as the input length increased, there was a tendency for training and inference times to increase, suggesting a balance between speed and performance should be considered when applying to actual systems.

The FCNN-based model exhibited a performance of 75%, which was lower than the LSTM-based model but achieved higher accuracy than the Naive Bayesian approach. This can be attributed to FCNN's classification mechanism, which relies solely on fixed-length vector inputs without considering contextual flow. Nevertheless, the FCNN-based model outperformed LSTM in terms of training and inference speed, demonstrating its potential applicability in real-time detection systems where computational efficiency is critical.

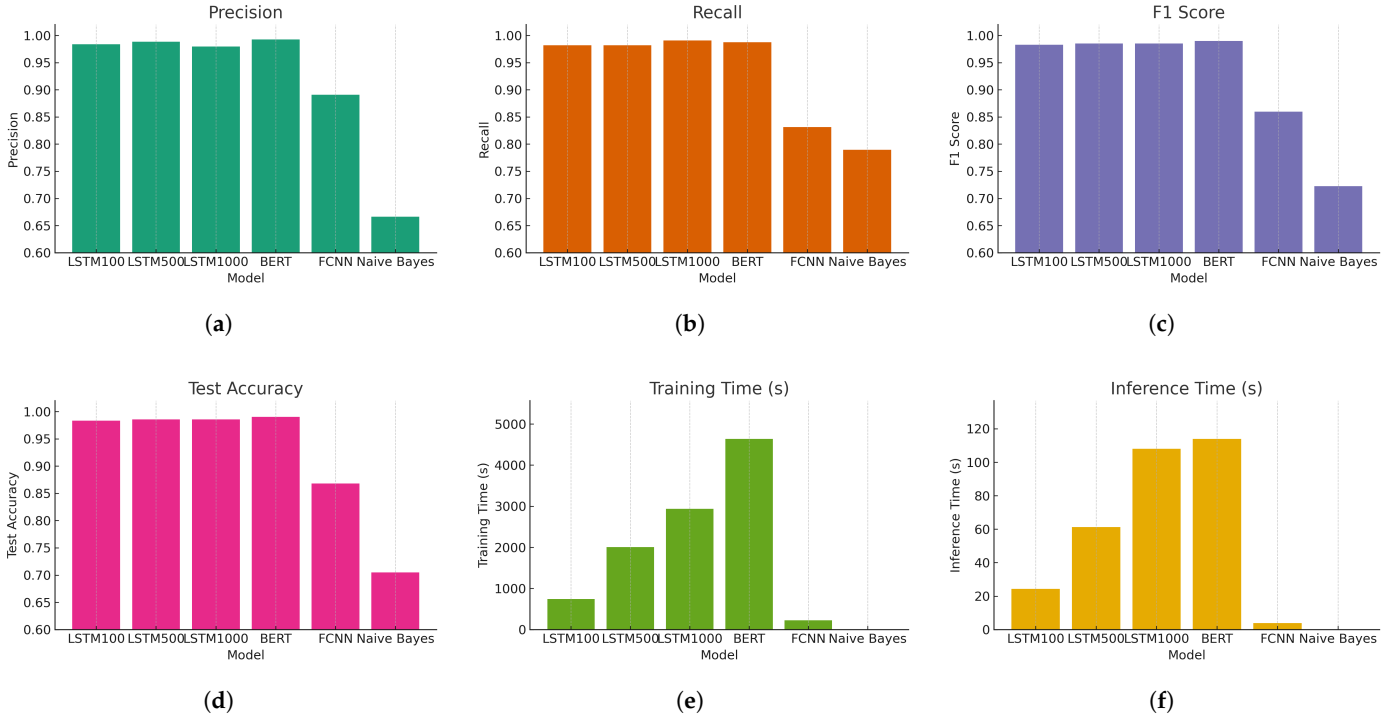


Figure 4. Performance comparison of six phishing detection models across six evaluation metrics: (**a**) Precision, (**b**) Recall, (**c**) F1 Score, (**d**) Test accuracy, (**e**) Training time (seconds), and (**f**) Inference time (seconds). Each graph illustrates the results of LSTM100, LSTM500, LSTM1000, BERT [26,27], FCNN [28], and Naive Bayes [29,30] models.

The Naive Bayesian model showed low classification performance in the 60% range, with relatively high rates of False Positives and False Negatives. This is analyzed to be due to the model utilizing only simple statistical features.

The BERT-based model achieved the highest classification performance among all tested models, with an accuracy of 99%. However, it demonstrated significantly lower efficiency in terms of training and inference time. Compared to the LSTM100 model, BERT required more than eight times longer for training and over four times longer for inference, and improved the accuracy by less than 0.1%. This is mainly due to BERT's computationally intensive architecture, which performs repeated attention-based operations over the entire input sequence.

These results demonstrate that the LSTM-based model proposed in this study achieves superior phishing detection performance compared to other approaches. Detection accuracy further improves when the input word count increases to 500 or 1000 words. While the FCNN-based model showed lower accuracy than LSTM, it achieved faster training and inference speeds, making it a viable option for real-time detection systems. However, the LSTM-based model consistently outperformed FCNN in accuracy, with the gap widening as more input data were utilized. The BERT model achieved the highest accuracy among all models and delivered the best performance in terms of classification. Nonetheless, its considerably long training and inference times posed limitations in terms of efficiency for real-world system deployment. In particular, our system aims to provide a solution in that detection rules are trained locally using the data collected inside an organization. The costs for the training and detection should be considered important if the difference in accuracy is subtle. Therefore, considering both detection speed and accuracy, the LSTM-based model maintains a practical processing time while delivering high detection performance. For the same reasons, we did not consider the use of LLM (Large Language Model)-based detection

methods as they are usually prohibitively costly and require providing the content of emails to LLM service providers. Therefore, the LSTM-based model offers a compelling balance between detection accuracy and processing speed, making it a suitable and practical option for real-world phishing detection systems.

3.2. Phishing Detection Report

PhiShield is a browser extension that analyzes emails in real-time and provides users with reports on whether an email is a phishing attempt. The system is designed to allow users to check for phishing threats in web-based email services, such as Gmail, before clicking on any emails. Figure 5 illustrates the operational flow of the PhiShield system, visually explaining the entire process from receiving an email in the browser to completing the analysis and generating a report.

When a user receives an email, PhiShield parses it in real-time. In this process, PhiShield gathers the subject, sender information, email content, and attachments, then it transmits the parsed and structured data to the server. Therefore, PhiShield automatically analyzes the email before any interaction with the user.

The parsed email data is sent to the PhiShield server, where our combined detection model is applied for the analysis. PhiShield uses both signature-based and AI-based detection algorithms to assess whether the email is phishing. Several analyses are performed simultaneously, including malicious URL analysis, sender trust evaluation, determining the safety of attachments, and so on. The AI model also effectively detects new phishing techniques by leveraging previously trained data.

Once the analysis is complete, PhiShield immediately sends the results back to the user's browser extension. Users can review the analysis results before opening the email. The analysis report provided by PhiShield includes detailed explanations to help users intuitively grasp the email's safety. Alongside the results for each element, the report provides a comprehensive evaluation of the email's URLs and the safety of any attachments.

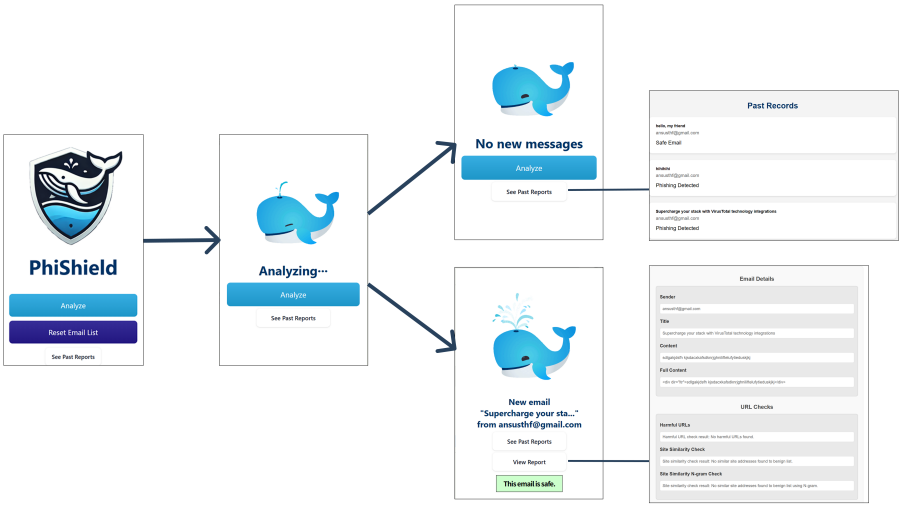


Figure 5. PhiShield system operation example.

Figure 6 shows an example of a phishing detection report provided by PhiShield. The report displays detailed information about the phishing risk, allowing users to quickly judge the email's safety based on the provided information.

Through this, PhiShield supports users in determining phishing threats before clicking on emails. They make a personalized decision about spam, as PhiShield phishing detection transparently processes offer reliability in email management. This helps users immediately

judge the safety of emails before opening them, effectively preventing damage caused by phishing emails.

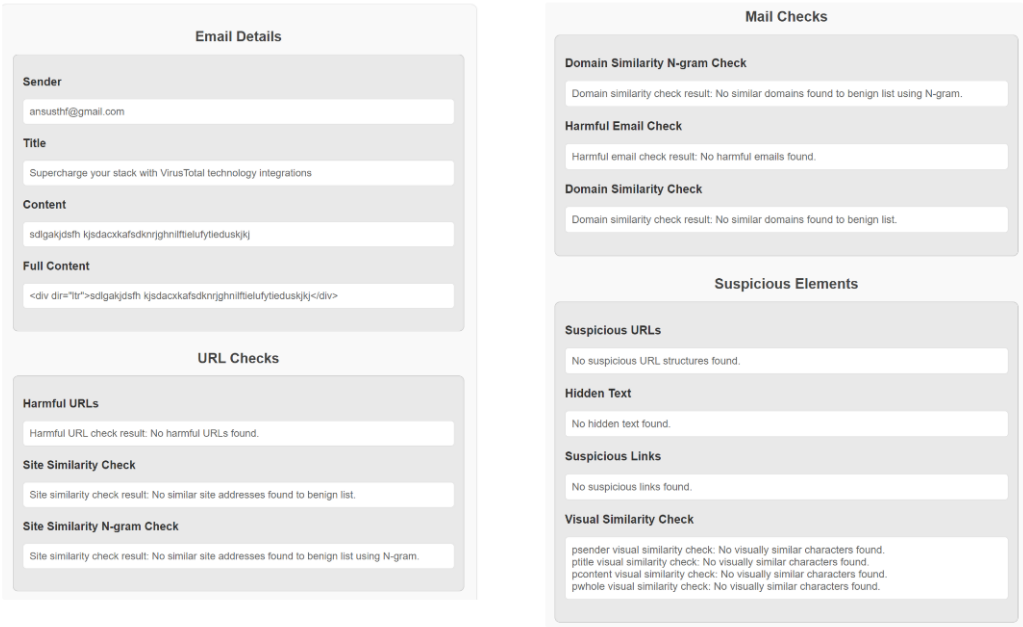


Figure 6. Phishing detection report.

4. Discussion

The results of this study demonstrate that the AI-based phishing detection system, PhiShield, achieves high detection accuracy while ensuring compatibility and proactive protection through its browser-based architecture. In particular, the LSTM-based detection model exhibited a strong balance between performance and efficiency, indicating its practical applicability for real-world deployment.

Compared to traditional LSTM-based phishing detection studies [8,9], PhiShield not only achieves comparable or better accuracy but also offers critical improvements in usability by providing proactive, real-time detection. Previous studies employing LSTM have primarily demonstrated model effectiveness without addressing real-time or privacy-related concerns. In contrast, PhiShield leverages a browser-based architecture to analyze and detect phishing emails proactively, thus preventing potential harm before users interact with malicious content.

Similarly, when compared with the SpamFilter-ChromeExtension [10], PhiShield exhibits clear advantages. While SpamFilter-ChromeExtension relies on Naive Bayes classifiers known for lower accuracy relative to LSTM [11], it also necessitates users to access potentially harmful emails before performing spam detection. PhiShield overcomes this limitation by enabling preemptive spam detection and safeguarding user privacy through localized training and personalized detection rules.

Furthermore, recent developments such as ChatSpamDetector [12] employing large language models (LLMs) have demonstrated advanced capabilities in explaining phishing detection outcomes. However, these solutions compromise privacy by transmitting sensitive email content to external LLM service providers and lack real-time detection capabilities. PhiShield addresses these shortcomings by locally analyzing emails directly within the user's browser, ensuring no sensitive data leaves the user's control and providing immediate, actionable insights without latency.

Based on these findings, future research should focus on incorporating personalized detection features and further enhancing the performance of AI-based models. By learning

the email usage patterns of individual users or organizations, PhiShield could be adapted to deliver customized detection rules, thereby evolving into a system capable of effectively mitigating advanced threats such as spear phishing. Although the current implementation is limited to web-based email services, future iterations may consider extending the deployment to include personal and enterprise mailboxes. Such an expansion would enhance email security for both individual and organizational users and facilitate consistent performance across diverse platforms.

Despite the use of four publicly available datasets for model evaluation, this study is limited by the absence of validation in real-world enterprise environments. To overcome this limitation, subsequent efforts should involve the construction and testing of the system using datasets that reflect realistic email environments. This approach would allow for more accurate simulation of actual operational conditions and provide a comprehensive assessment of PhiShield's performance against a broader spectrum of phishing threats.

Another area requiring further investigation is the system's resilience to evasion techniques. Given that attackers may deliberately manipulate email content to circumvent detection—particularly in the case of signature-based methods—future work should consider integrating adversarial training. Introducing manipulated phishing samples during training may improve the system's robustness and enable consistent detection of deceptive or intentionally tampered messages.

To enhance user trust and promote transparency, the integration of explainable AI (XAI) techniques should also be pursued. Techniques such as Shapley Additive explanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) offer intuitive explanations for classification outcomes and could provide valuable insights into the model's decision-making process. Although PhiShield currently categorizes detection results by type, it does not yet incorporate a dedicated interpretability layer. Future versions of the system should aim to implement such functionality to support user understanding and interaction.

5. Conclusions

In this study, we introduce an AI-based phishing detection system, PhiShield. Unlike conventional phishing detection systems that mainly focus on detection accuracy, PhiShield also considers compatibility, privacy, and proactive security at the same time.

PhiShield offers compatibility with third-party email services. PhiShield is implemented as a browser extension. This makes PhiShield highly scalable from a deployment perspective and enables it to be compatible with multiple email service providers. Moreover, PhiShield also preserves the user's privacy and personalization by allowing the user to use the locally trained AI-based spam detection models. Because the generation of and updates to the models can be controlled from the user's side, PhiShield provides personalized detection rules that can reflect the specific security needs of organizations or individuals without giving private information to a third party. It also enables users to determine phishing threats proactively by reporting the detection results before a user opens it. This proactive approach greatly reduces the risks of phishing by reducing the user's risk of clicking spam emails. Moreover, PhiShield provides users with a detailed report explaining the results of phishing detection clearly and transparently. By generating comprehensive reports, PhiShield helps users understand why an email was classified as phishing, enhancing user trust and awareness regarding email security.

Under these design principles, PhiShield shows outstanding detection performance. According to our performance evaluation, it boasts above 98% detection accuracy with a reasonable training and inference time. Therefore, it is highly expected to strengthen the email security environment and effectively prevent the damage caused by phishing emails.

Author Contributions: Conceptualization, H.M. and J.P.; methodology, Y.K.; validation, B.K., Y.K. and J.K.; formal analysis, H.M.; investigation, J.P.; data curation, Y.K.; writing—original draft preparation, H.M. and J.P.; writing—review and editing, J.K.; visualization, H.M.; supervision, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

LSTM	Long short-term memory
AI	Artificial intelligence
S/MIME	Secure/Multipurpose Internet Mail Extensions
TLS	Transport Layer Security
LLM	Large language model
REST	Representational state transfer
API	Application programming interface
AWS	Amazon Web Services
EC2	Elastic Compute Cloud
RNN	Recurrent neural network
FCNN	Fully connected neural network
BERT	Bidirectional Encoder Representations from Transformers
XAI	Explainable artificial intelligence
SHAP	SHapley additive exPlanations
LIME	Local interpretable model-agnostic explanations

References

1. Griffiths, C. The Latest 2024 Phishing Statistics (Updated June 2024). 2024. Available online: https://aag-it.com/the-latest-phishing-statistics/ (accessed on 20 September 2024).
2. Gajjar, V.R.; Taherdoost, H. Cybercrime on a Global Scale: Trends, Policies, and Cybersecurity Strategies. In Proceedings of the 2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI), Lalitpur, Nepal, 18–19 January 2024; IEEE: New York, NY, USA, 2024; pp. 668–676.
3. Agboola, O.T. Development of a Novel Approach to Phishing Detection Using Machine Learning. *J. Sci. Technol. Educ.* **2024**, *12*, 337–338.
4. Aljabri, M.; Altamimi, H.S.; Albelali, S.A.; Al-Harbi, M.; Alhuraib, H.T.; Alotaibi, N.K.; Salah, K. Detecting malicious URLs using machine learning techniques: Review and research directions. *IEEE Access* **2022**, *10*, 121395–121417. [CrossRef]
5. Do, N.Q.; Selamat, A.; Krejcar, O.; Herrera-Viedma, E.; Fujita, H. Deep learning for phishing detection: Taxonomy, current challenges and future directions. *IEEE Access* **2022**, *10*, 36429–36448. [CrossRef]
6. Almalkawi, I.T.; Al-Hammouri, M.F.; Mallouh, M.A.; Barakat, T. Improving Email Security Through Machine Learning-Based Phishing Attack Detection. In Proceedings of the 2024 International Jordanian Cybersecurity Conference (IJCC), Amman, Jordan, 17–18 December 2024; pp. 180–185. [CrossRef]
7. Sarkar, S.; Yadav, A.; Balachander, T. Email Phishing Detection Using AI and ML. In Proceedings of the Deep Sciences for Computing and Communications, Chennai, India, 20–22 April 2023; R., A.U., Kottursamy, K., Raja, G., Bashir, A.K., Kose, U., Appavoo, R., Madhivanan, V., Eds.; Springer: Cham, Switzerland, 2024; pp. 357–377.
8. Li, Q.; Cheng, M.; Wang, J.; Sun, B. LSTM based phishing detection for big email data. *IEEE Trans. Big Data* **2022**, *8*, 278–288. [CrossRef]
9. Alshingiti, Z.; Alaqel, R.; Al-Muhtadi, J.; Haq, Q.E.U.; Saleem, K.; Faheem, M.H. A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN. *Electronics* **2023**, *12*, 232. [CrossRef]

10. SpamFilter-ChromeExtension. Available online: https://github.com/surya-veer/SpamFilter-ChromeExtension (accessed on 20 September 2024).
11. Naive Bayes and LSTM Based Classifier Models. Available online: https://towardsdatascience.com/naive-bayes-and-lstm-based-classifier-models-63d521a48c20 (accessed on 20 September 2024).
12. Koide, T.; Fukushi, N.; Nakano, H.; Chiba, D. Chatspamdetector: Leveraging large language models for effective phishing email detection. *arXiv* **2024**, arXiv:2402.18093.
13. Phishing Email Detection. Available online: https://www.kaggle.com/datasets/subhajournal/phishingemails (accessed on 20 September 2024).
14. Al-Subaiey, A.; Al-Thani, M.; Alam, N.A.; Antora, K.F.; Khandakar, A.; Zaman, S.A.U. Novel Interpretable and Robust Web-based AI Platform for Phishing Email Detection. *arXiv* **2024**, arXiv:2405.11619. [CrossRef]
15. Amrutkar, P. Large-Scale Phishing Email Dataset. 2023. Available online: https://zenodo.org/record/8339691 (accessed on 20 September 2024).
16. Apaulgithub. Spam Email Dataset (spam.csv). 2022. Available online: https://github.com/Apaulgithub/oibsip_taskno4/blob/main/spam.csv (accessed on 20 September 2024).
17. Said Elsayed, M.; Le-Khac, N.A.; Dev, S.; Jurcut, A.D. Network anomaly detection using LSTM-based autoencoder. In Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, Alicante, Spain, 16–20 November 2020; pp. 37–45. [CrossRef]
18. Khonji, M.; Iraqi, Y.; Jones, A. Phishing Detection: A Literature Survey. *IEEE Commun. Surv. Tutorials* **2013**, *15*, 2091–2121. [CrossRef]
19. Sheng, S.; Wardman, B.; Warner, G.; Cranor, L.F.; Hong, J.; Zhang, C. An Empirical Analysis of Phishing Blacklists. In Proceedings of the 6th Conference on Email and Anti-Spam (CEAS 2009), Mountain View, CA, USA, 16–17 July 2009; pp. 60–69.
20. Sharifi, M.; Siadati, S.H. A phishing sites blacklist generator. In Proceedings of the 2008 IEEE/ACS International Conference on Computer Systems and Applications (AICCSA '08), Washington, DC, USA, 31 March–4 April 2008; IEEE Computer Society: Washington, DC, USA, 2018; pp. 840–843.
21. Pascariu, C.; Bacivarov, I.C. Detecting Phishing Websites Through Domain and Content Analysis. In Proceedings of the 2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Pitesti, Romania, 1–3 July 2021; pp. 1–4. [CrossRef]
22. Islavath, S.; Bhat, C.R. Uniform Resource Locator Phishing in Real Time Scenario Predicted Using Novel Term Frequency-Inverse Document Frequency +N Gram in Comparison with Support Vector Machine Algorithm. In Proceedings of the 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kamand, India, 24–28 June 2024; pp. 1–5. [CrossRef]
23. Sokolov, M.; Olufowobi, K.; Herndon, N. Visual spoofing in content-based spam detection. In Proceedings of the 13th International Conference on Security of Information and Networks, Istanbul, Turkey, 4–6 November 2020; pp. 1–5.
24. Cook, D.L.; Gurbani, V.K.; Daniluk, M. Phishwish: A stateless phishing filter using minimal rules. In Proceedings of the Financial Cryptography and Data Security, Cozumel, Mexico, 28–31 January 2008; Tsudik, G., Ed.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 182–186.
25. Zhang, Y.; Hong, J.I.; Cranor, L.F. Cantina: A content-based approach to detecting phishing web sites. In Proceedings of the 16th International Conference on World Wide Web (WWW '07), New York, NY, USA, 8–12 May 2007; pp. 639–648.
26. Tida, V.S.; Hsu, S.H.Y. Universal Spam Detection using Transfer Learning of BERT Model. *arXiv* **2022**, arXiv:2202.03480.
27. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
28. Subba, B. A heterogeneous stacking ensemble-based security framework for detecting phishing attacks. In Proceedings of the 2023 National Conference on Communications (NCC), Guwahati, India, 23–26 February 2023; pp. 1–6. [CrossRef]
29. Chien, A.; Khethavath, P. Email Feature Classification and Analysis of Phishing Email Detection Using Machine Learning Techniques. In Proceedings of the 2023 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Nadi, Fiji, 4–6 December 2023; pp. 1–8. [CrossRef]
30. Chinnasamy, P.; Krishnamoorthy, P.; Alankruthi, K.; Mohanraj, T.; Kumar, B.S.; Chandran, L. AI Enhanced Phishing Detection System. In Proceedings of the 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), Krishnankoil, India, 14–16 March 2024; pp. 1–5. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.