algorithms

MDPI

Article

Evaluating the Effectiveness of Large Language Models (LLMs) Versus Machine Learning (ML) in Identifying and Detecting Phishing Email Attempts

Saed Tarapiah [1,*], Linda Abbas [2], Oula Mardawi [3], Shadi Atalla [4], Yassine Himeur [4] and Wathiq Mansoor [4]

[1] Department of Telecommunication Engineering, An-Najah National University, Nablus P.O. Box 7, Palestine
[2] Department of Information Technology, An-Najah National University, Nablus P.O. Box 7, Palestine; s12255086@stu.najah.edu
[3] Department of Computer Engineering, An-Najah National University, Nablus P.O. Box 7, Palestine; oula.mardawi@najah.edu
[4] College of Engineering and Information Technology, University of Dubai, Dubai P.O. Box 14143, United Arab Emirates; satalla@ud.ac.ae (S.A.); yhimeur@ud.ac.ae (Y.H.); wmansoor@ud.ac.ae (W.M.)
* Correspondence: s.tarapiah@najah.edu

Abstract

Phishing emails remain a significant concern and a growing cybersecurity threat in online communication. They often bypass traditional filters due to their increasing sophistication. This study presents a comparative evaluation of machine learning (ML) models and transformer-based large language models (LLMs) for phishing email detection, with embedded URL analysis. This study assessed ML training and LLM fine-tuning on both balanced and imbalanced datasets. We evaluated multiple ML models, including Random Forest, Logistic Regression, Support Vector Machine, Naïve Bayes, Gradient Boosting, Decision Tree, and K-Nearest Neighbors, alongside transformer-based LLMs DistilBERT, ALBERT, BERT-Tiny, ELECTRA, MiniLM, and RoBERTa. To further enhance realism, phishing emails generated by LLMs were included in the evaluation. Across all configurations, both the ML models and the fine-tuned LLMs demonstrated robust performance. Random Forest achieved over 98% accuracy in both email detection and URL classification. DistilBERT obtained almost as high scores on emails and URLs. Balancing the dataset led to slight accuracy gains in ML models but minor decreases in LLMs, likely due to their sensitivity to majority class reductions during training. Overall, LLMs are highly effective at capturing complex language patterns, while traditional ML models remain efficient and require low computational resources. Combining both approaches through a hybrid or ensemble method could enhance phishing detection effectiveness.

Keywords: phishing; emails; cybersecurity; threat; URL; machine learning; large language models; balanced datasets; imbalanced datasets; ensemble

Academic Editor: Mostafa Abbaszadeh

Received: 25 July 2025
Revised: 16 September 2025
Accepted: 22 September 2025
Published: 25 September 2025

Citation: Tarapiah, S.; Abbas, L.; Mardawi, O.; Atalla, S.; Himeur, Y.; Mansoor, W. Evaluating the Effectiveness of Large Language Models (LLMs) Versus Machine Learning (ML) in Identifying and Detecting Phishing Email Attempts. *Algorithms* **2025**, *18*, 599. https://doi.org/10.3390/a18100599

1. Introduction

With the rapid evolution of internet technologies and increasing reliance on online platforms, concerns about cybersecurity issues have amplified. These emerging threats pose an increasing danger to users across digital environments, which can also lead to financial and identity loss. Phishing attacks take several forms, including URLs, emails, and websites, which are among the most frequently used methods [1]. Email, in particular, has emerged as the most widely used, efficient, and cost-effective form of communication, resulting in a significant increase in the volume of emails exchanged; as a result, there

is a need for more accurate spam filtering solutions that can detect phishing emails in real time [2].

Major email service providers, like Gmail, Yahoo Mail, and Outlook, have deployed advanced spam filtering systems that incorporate methods like blocklists alongside a variety of machine learning techniques, including neural networks, to address the threats posed by email spam effectively [3]. Since phishing attacks are constantly evolving, there is still a real need for new and more effective ways to detect them. However, emails that do not contain any links, also known as link-less emails, require an effort for spam filters [4]. Although the capabilities of large language models (LLMs) have been widely studied for multiple tasks such as code generation, text synthesis, and summarization, their potential in analyzing and detecting malicious web content, particularly phishing emails, has received comparatively less attention [5].

The term "phishing" first appeared in the mid-1990s, with early incidents involving fake credit card details to access America Online (AOL) services. By 1995, attackers began impersonating AOL staff to trick users into revealing personal information. Although phishing started during that period, it did not become widely recognized by the public until many years later [6,7]. Phishing remains one of the most widespread and destructive types of cyberattacks, designed to trick users into revealing sensitive information or installing malware. CyBOK classifies it as both a technical and social engineering threat, exploiting users' lack of awareness to deceive them [8].

Detecting phishing emails is becoming more difficult due to various modern challenges. Many phishing websites now use secure protocols like HTTPS, which gives users a false sense of safety, misled by SSL certificates. Studies show that over 45% of phishing URLs use HTTPS, making it hard to distinguish between fake and real sites [9]. Another issue is advanced persistent threats (APTs) and zero-day attacks, which are new and evolving phishing methods that traditional machine learning models struggle to catch; they have never encountered similar patterns before [10]. Additionally, traditional machine learning models (ML) approaches often depend on manual feature engineering, which is time-consuming, error-prone, and dependent on expert knowledge. This manual process can limit the scalability and effectiveness of detection systems [11,12].

The Anti-Phishing Working Group (APWG), a global organization that monitors phishing attacks, reported that nearly 5 million phishing incidents occurred in 2023, making it the worst year on record [13]. The 2023 SlashNext Phishing Intelligence Report indicated a 45% increase in malicious threats and noted phishing and Business Email Compromise (BEC) were reported as the primary source of financial loss. According to the Federal Bureau of Investigation's (FBI) Internet Crime Complaint Center (IC3) report, total losses are estimated at $6.9 billion, led by phishing and BEC. SlashNext also confirmed that malicious phishing email attacks increased by 1265% since Q4 of 2022 and that 68% of phishing attacks were text-based, and credential theft increased by 967%. The generative AI tools currently accessible to the public, like ChatGPT 4, have also enabled much more sophisticated attack methods [14].

Phishing attacks rely on social engineering to deceive victims into providing sensitive info or funds by impersonating trusted online sources (e.g., banks, organizations using online platforms, online shops, and government services). To identify phishing threats, ML algorithms can be applied in various forms, including kNN, Random Forest, Decision Tree, SVM, and Naïve Bayes. The final choice of algorithm often depends heavily on the problem and data type, as some ML models are better suited to structured data as opposed to unstructured data [15–18]. In the context of phishing email detection, fine-tuning improves the classification accuracy of a pre-trained LLMs by matching it with the patterns of malicious messages. Fine-tuning involves training a model, such as an LLM,

on a particular dataset to enhance its performance on specialized tasks. However, it has challenges, mainly the high computation process, as it needs a lot of memory and GPU power, making it both expensive and time-consuming [19–21].

2. Research Objectives

The main goals of this research are outlined below:

1. To compare the classification performance of large language models (LLMs) and traditional machine learning (ML) models in phishing email detection, using metrics such as precision, recall, F1 score, accuracy, and balanced accuracy.
2. To assess the computational efficiency of both LLMs and ML models during training and fine-tuning, including processor usage, memory consumption, and overall resource requirements.
3. To evaluate the capability of LLMs to handle complex and nuanced phishing content, highlighting their advancements in text classification.
4. To analyze misclassification cases from both LLMs and ML models to identify potential areas for improvement in future phishing detection systems.

3. Materials and Methods

3.1. Overview of Dataset Collection

The dataset used in this study was manually collected from various online source platforms. The raw datasets consisted of email texts and URLs from different research projects and repositories. This dataset contains 146,426 email records, which were reduced to about 26,365 samples (approximately 18%). Of these, 7623 were already labeled as phishing, while 18,742 were classified as legitimate. The URL dataset initially had 449,271 entries, but it was filtered down based on the proposed cleaning method to 67,391 (around 15%). Among these, 15,540 were phishing URLs and 51,851 were legitimate. Datasets were randomly combined from different sources to enhance the diversity of the dataset.

The labels were coded as binary values, with "1" representing phishing and "0" representing legitimate URLs and emails. To add variety to the dataset, additional emails generated by LLMs were included to help balance it. The final dataset consisted of samples from Enron, Fraud, LingSpam, Nazario, SpamAssassin, TREC-06, and content generated by LLMs, along with Huggingface_dataset and URL_dataset. The LLM-generated emails used in this study were not created manually but obtained from an existing research project published on Kaggle, titled *Human-LLM Generated Phishing–Legitimate Emails*. This dataset was specially designed to provide realistic synthetic phishing and legitimate email examples, ensuring reliability and reusability. Details about the dataset and its sources are provided in Appendix A.

This study trained traditional ML and fine-tuned the pre-trained LLMs separately on email and URL datasets. Training occurred in two stages: first, with imbalanced data to reflect real-world scenarios, then with balanced datasets. The datasets were balanced through under-sampling of the majority class, which maintains data quality. Figure 1 illustrates the dataset distributions before balancing both the email and URL datasets. The sources of the emails and their distribution are shown in Figure 2. https://www.kaggle.com/datasets/francescogreco97/human-llm-generated-phishing-legitimate-emails/data (accessed on 21 September 2025).

3.2. Data Processing

3.2.1. Text Cleaning and Normalization

Due to the strong dependence of ML model and LLM performance on the quality of the training data, data quality has received increasing attention in recent research, with

numerous approaches developed to detect, correct, and prevent data errors [22]. High-quality, consistent data is essential for ensuring that models can learn meaningful patterns and generalize effectively to unseen examples. To this end, we applied a comprehensive text cleaning pipeline to prepare the email data for training. The cleaning process involved removing HTML tags, special symbols, and stop words, which are typically considered noise and can negatively impact model learning. In addition, all text was converted to lowercase to ensure uniformity across entries and avoid discrepancies due to case sensitivity. Following these steps, the text data was further normalized to maintain consistency across datasets from multiple sources to ensure that variations in formatting did not introduce unintended bias or inconsistencies into the model training process.

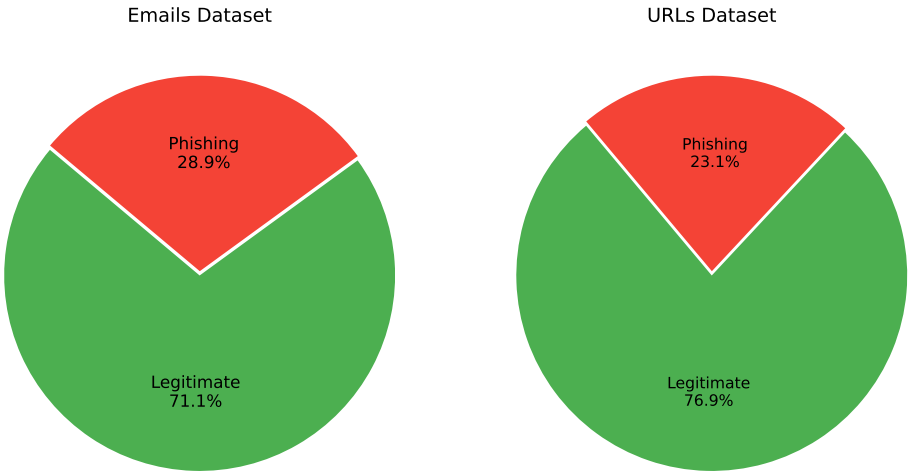


Figure 1. Dataset Distribution: 28.9% of emails and 23.1% of URLs labeled as phishing.

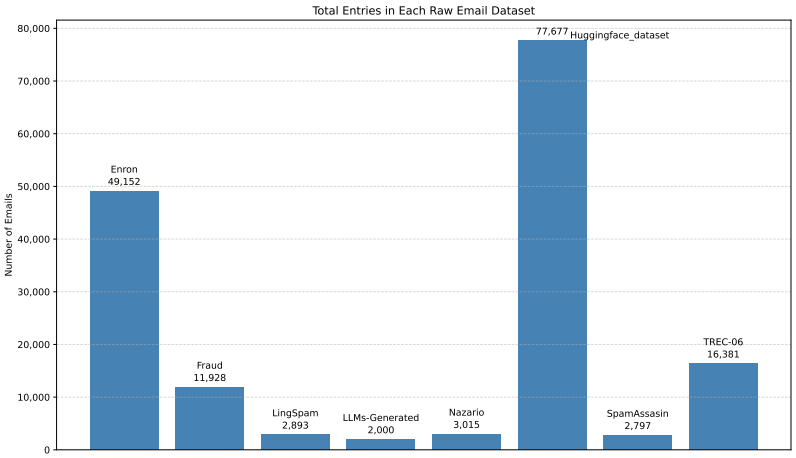


Figure 2. Entry Distribution Based on Email Sources (Total = 146,426).

3.2.2. Data Preparation for ML

After the cleaning process, the text data was normalized to maintain consistency. Two methods of vectorization techniques were tested: TF-IDF and Word2Vec [23]. While both were effective, the TF-IDF feature extraction yielded slightly better results for phishing email detection. In addition, as supported by previous research, the TF-IDF method was chosen as the primary method for creating the final machine learning models. Regarding URLs, a total of 30 engineered features were extracted, including the presence of HTTPS, URL length, number of dots, and the use of special characters, to distinguish phishing links from legitimate ones [24]. Feature extraction was applied only to traditional ML models. In contrast, LLMs operate directly on raw URL text and leverage contextual and

semantic representations without requiring manual feature engineering. A complete list of all features used in this study is provided in Appendix B. These numerical features were then normalized using MinMaxScaler to scale all values to a standard range between 0 and 1, improving model performance and consistency. Figure 3 shows the feature distributions after normalization.

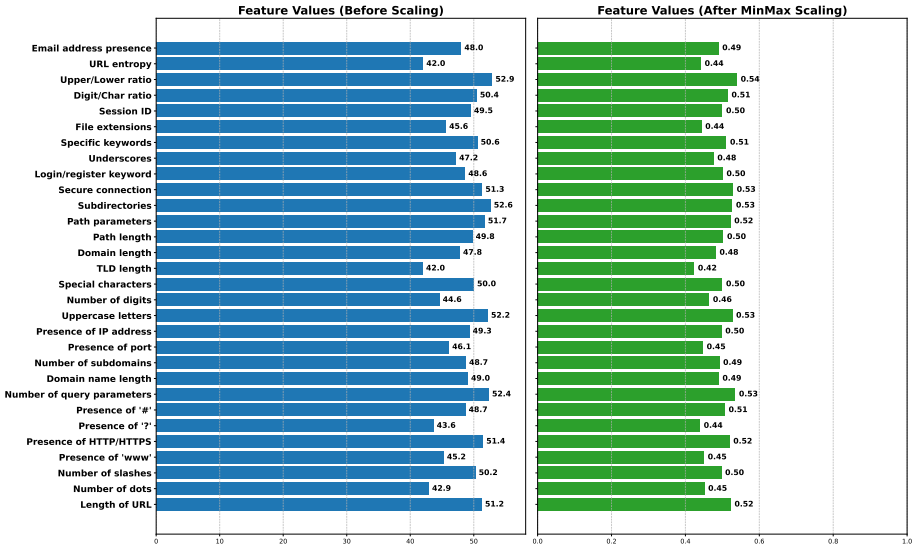


Figure 3. Normalization Effect of URL Features Using MinMaxScaler.

Phishing datasets are retained and fine-tuned through our filtering approach to optimize and match computational efficiency resource limitations.

3.2.3. Data Preparation for LLMs

Tokenization is a fundamental preprocessing step in LLMs that transforms raw text into smaller units called tokens for model processing. LLMs rely on subword tokenization methods such as WordPiece, Byte Pair Encoding (BPE), and SentencePiece to break words into meaningful subword units. This approach allows the model to handle rare, compound, or misspelled words effectively. The HuggingFace Transformers library offers the AutoTokenizer class, which automatically loads the appropriate pretrained tokenizer for each model architecture (e.g., BERT, RoBERTa), maintaining alignment with the model's vocabulary and ensuring accurate text representation.

The Max_seq_length parameter, which determines the maximum number of tokens processed from each input, was set to 256 tokens for email texts and 128 for URL inputs. These values were applied consistently across all transformer LLMs to ensure consistent performance. The choices were based on the token length distributions observed in the datasets, as shown in Figure 4.

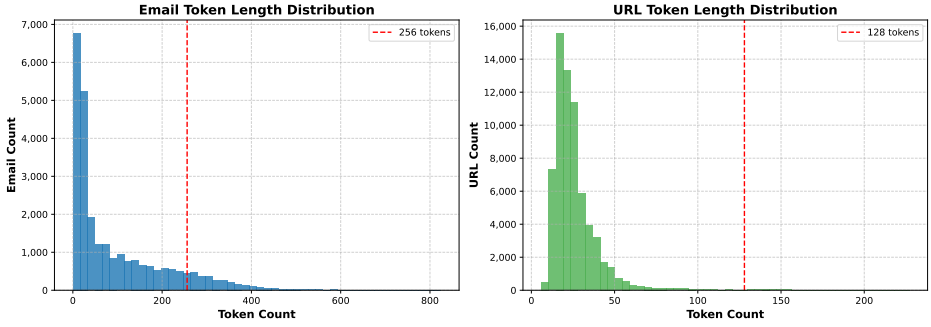


Figure 4. Managing Sequence Lengths in Tokens.

3.3. Model Selection for ML and LLMs

Research in the field of email fraud detection using machine learning and deep learning technologies dates back as early as 2015. Since then, various supervised learning algorithms [25] have set the baseline for detection. Unfortunately, comparing findings across studies remains challenging, as different researchers have employed varying algorithms, datasets, and evaluation methods, particularly in how accuracy is measured [26]. Although research works and literature on transformer-based approaches are comparatively limited, they are highly popular for dealing with natural language processing (NLP), and even include complex tasks involving sentiment analysis, text generation, and human emotion recognition [27].

ML models such as Decision Tree (DT), Linear Regression (LR), Support Vector Machine (SVM), Gradient Boosting, and Neural Networks, including deep learning (DL) architectures, have long formed the core of artificial intelligence (AI) and have been widely applied for decades. These models utilize statistical techniques and data mining to automatically identify patterns within data, enabling them to perform tasks like prediction and classification. Each of these models is designed for particular tasks and can be customized and fine-tuned according to specific requirements [17,18,28]. For more details on the implementation and parameters of these models, see Appendix C.

Generative AI is a branch of artificial intelligence that produces human-like content such as text, images, or code, based on patterns learned from large datasets by humans [20]. Recently, LLMs have demonstrated an extraordinary capacity in various NLP tasks, including question answering, text generation, and language translation. Moreover, LLMs can comprehend complex speech patterns and produce responses that are suitable and coherent with given contexts [29]. LLMs were first introduced with GPT-1 in 2018 by OpenAI as an initial prototype, demonstrating to us a new way for machines to comprehend and produce human-like text. ML, using the transformer architecture of these models, can learn from large amounts of available data to create coherent and contextually relevant outputs. These models can be easily integrated into major AI frameworks such as [30] and PyTorch 2.5.1. Furthermore, the development and economics of libraries, like those provided by Hugging Face, have made transformers accessible to a broad community of researchers and developers [31]. For more details about the models, see Appendix C.

3.4. Data Splitting

The dataset was split into 80% for training and 20% for testing, with 60% of the total data allocated for training and 20% for validation. The validation set was used after each epoch to monitor model performance and prevent overfitting. This consistent strategy was applied in both traditional machine learning and LLM fine-tuning, ensuring reliable results and improving prediction accuracy on unseen data. Figure 5 represents the dataset splitting process used throughout the experiments.

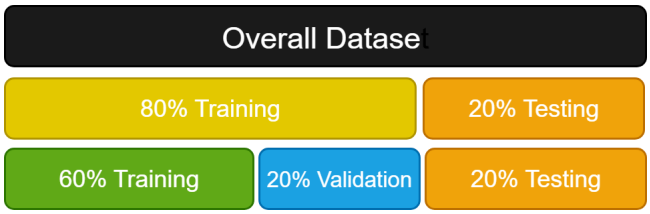


Figure 5. Data Splitting Strategy: Training, Validation, and Testing Sets.

3.5. Evaluation Metrics

To evaluate the performance of the proposed models, a set of standard classification metrics is used. These metrics are standard benchmarks accepted in natural language

processing and machine learning for binary classification problems, such as distinguishing phishing (negative) and legitimate (positive) email content. These include True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

Accuracy is defined as the ratio of correctly predicted samples to all predictions made by the model.

$$\textit{Balanced Accuracy} = \frac{TP + TN}{TP + FN + TN + FP} \tag{1}$$

Balanced Accuracy is the average of true positive and true negative rates, offering a fair assessment in imbalanced classification tasks.

$$\textit{Balanced Accuracy} = \frac{1}{2} \times \left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP} \right) \tag{2}$$

Precision is the ratio of true positive predictions to the total predicted positives, indicating how many of the model's positive predictions were correct.

$$\textit{Precision} = \frac{TP}{TP + FP} \tag{3}$$

Recall is the ratio of true positives to all actual positive data and indicates the model's ability to retrieve all of the positive samples.

$$\textit{Recall} = \frac{TP}{TP + FN} \tag{4}$$

F1 Score combines both precision and recall into a single metric, offering a balanced evaluation of a model's performance, especially on imbalanced datasets.

$$\textit{F1 Score} \frac{2 \times TP}{2 \times TP + FP + FN} \tag{5}$$

3.6. Experimental Configuration

In this study, phishing email detection was examined by analyzing email content that may include URLs, using both ML models and fine-tuned lightweight LLMs. The traditional ML models applied include Decision Tree, Logistic Regression, Random Forest, Naive Bayes, Gradient Boosting, K-Nearest Neighbors (KNN), and Support Vector Machine. For transformer-based approaches, several efficient and compact models were fine-tuned, including ALBERT, BERT-Tiny, DistilBERT, ELECTRA-Tiny, MiniLM, and RoBERTa. The primary objective of this work is to evaluate and compare their performance based on classification accuracy and other relevant metrics to see whether ML models or transformer-based architectures LLMs work better for phishing email detection. The ML models were configured using text vectorization and feature extraction, while the LLMs relied on specialized text tokenization. These configurations are explained in Sections 3.6.1 and 3.6.2, respectively.

3.6.1. ML Model Training Configuration

Text Cleaning: This included removing unnecessary symbols, non-alphabetic values, and special characters. All of the characters in the text were also converted to lowercase in order to standardize it.

Vectorization: After the cleaning stage, the email content was tokenized, breaking down all email messages into their respective words or sub-word units so that meaningful patterns could be recognized during encoding. These tokens were then converted into numerical representations that ML algorithms can understand using TF-IDF [23], which represents text as weighted feature vectors. The vectorizer was configured with:

- max_features = 20,000
- ngram_range = (1, 2)

URL Extraction feature: URL content was represented using 30 structural and lexical features (e.g., length, special characters, keywords, entropy) and normalized to a [0, 1] range using MinMaxScaler. The Bernoulli Naive Bayes (BernoulliNB) algorithm was used due to the binary nature of many features. Classifier Parameter Configuration: Specific hyperparameters were set for each traditional ML model to enhance performance. The configurations used in this study are shown in Table 1.

Table 1. Hyperparameter Settings for ML.

Model Name	Configuration
Logistic Regression (LR)	max_iter = 2000, random_state = 42, class_weight = 'balanced'
Random Forest (RF)	n_estimators = 100, random_state = 42, max_features = 'sqrt'
Gradient Boosting (GB)	n_estimators = 150, learning_rate = 0.1, max_depth = 3
Support Vector Machine (SVM)	kernel = 'linear', C = 10, Gamma = 0.01, Probability = True, random_state = 42
K-Nearest Neighbors (KNN)	n_neighbors = 5

3.6.2. LLM Training Configuration

Text tokenization [32] is a crucial preprocessing step that converts raw text into smaller units called tokens, which the model can then process. LLMs utilize a method according to subword tokenization, including WordPiece, Byte Pair Encoding (BPE), and SentencePiece, to break words into subwords and meaningful units for the model to allow for handling of rare, compound, or misspelled words. The architecture of the model determines differences between tokenization methods. In practice implementations, tokenization is handled by widely used libraries and frameworks made for large language models. The HuggingFace Transformers library includes the useful AutoTokenizer class, allowing users to load the appropriate pre-trained tokenizer for a model architecture directly (i.e., BERT, RoBERTa). These tools make the tokenization easier, maintain alignment with the model's training vocabulary, and support productive batching and preprocessing [33]. Training was performed using the Adam optimizer by default. Each model was fine-tuned separately on the email and URL datasets, using consistent parameters, as illustrated in Table 2.

Table 2. Hyperparameter Settings for LLMs.

Dataset	Configuration
Email Dataset	Num_train_epochs = 3,Weigh_dcay = 0.01, Learning_rate = 2×10^{-5}, Batch_size = 8, Max_seq_length = 256,
URL Dataset	Num_train_epochs = 2, Weigh_decay = 0.01, Learning_rate = 2×10^{-5}, Batch_size = 16, Max_seq_length = 128,

3.7. A Pipeline for Phishing Email Detection Using Vectorization and Tokenization

A complete overview of the pipeline used for classifying email messages as either phishing or legitimate is presented in Figure 6, based on two parallel approaches: traditional ML and transformer LLMs. When a user enters email text through a web interface, the system first pre-processes the text to enable classification through both methods. In the ML pipeline, the raw text is transformed into numerical vectors using approaches like TF-IDF vectorization, which captures the frequency and significance of words within the document corpus. These vectors are then passed to classifiers such as Random Forest (RF), Support Vector Machine (SVM), and other models used in this study to generate predictions.

Concurrently, transformer-based models follow a tokenization process, in which the input text is split into subword tokens, mapped to token IDs, and supplemented with special tokens like [CLS] and [SEP]. The sequences are padded or truncated to a fixed length of tokens and then processed through transformer architectures (e.g., BERT) to classify the email. In the end, the predictions from both approaches are returned to the frontend interface and displayed to the user. The ML models and LLMs were applied to a dataset of 26,365 emails and 67,391 URLs.

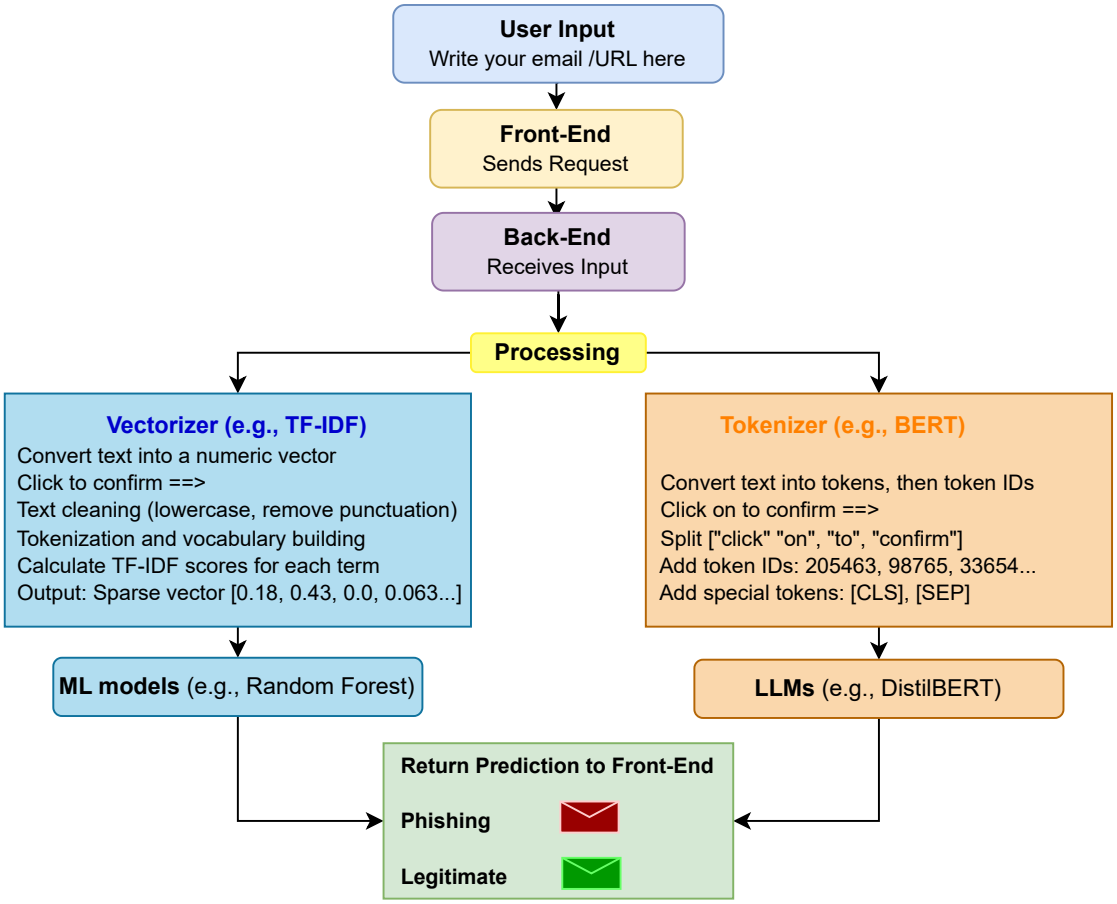


Figure 6. Email Prediction Workflow.

4. Results

The results in this section cover both traditional ML and LLMs. Results include evaluations on both email and URL datasets under balanced and imbalanced conditions.

4.1. Results for ML Approaches

4.1.1. Email Content Analysis Using ML

The evaluation results, as shown in Table 3, indicate that Random Forest achieved the best performance on both imbalanced and balanced email datasets, with the highest accuracies of 0.9947 and 0.9959, respectively. Support Vector Machine (SVM) followed closely, with substantial precision and recall values and a balanced accuracy of 0.9954 on the balanced dataset. Logistic Regression also showed competitive results, particularly in the balanced dataset with an accuracy of 0.9846. While Decision Tree and K-Nearest Neighbors (KNN) demonstrated solid performance, Naïve Bayes and Gradient Boosting recorded lower accuracy and recall, making them less suitable for phishing email detection in this context. Overall, Random Forest proved to be the most effective and reliable traditional ML model across both dataset distributions.

Table 3. Comparison of ML Results for Email Dataset (Balanced vs. Imbalanced).

Imbalanced Email Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
Decision Tree	0.9884	0.9749	0.9850	0.9799	0.9874
Gradient Boosting	0.9498	0.9667	0.8554	0.9077	0.9217
K-Nearest Neighbors	0.9722	0.9581	0.9448	0.9514	0.9640
Logistic Regression	0.9747	0.9802	0.9311	0.9550	0.9617
Naïve Bayes	0.9521	0.9631	0.8672	0.9126	0.9269
Random Forest	**0.9947**	0.9926	0.9890	0.9908	0.9930
Support Vector Machine	0.9935	0.9842	0.9934	0.9888	0.9935
Balanced Email Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
Decision Tree	0.9896	0.9848	0.9946	0.9897	0.9896
Gradient Boosting	0.9545	0.9333	0.9875	0.9596	0.9545
K-Nearest Neighbors	0.9755	0.9743	0.9769	0.9756	0.9755
Logistic Regression	0.9846	0.9774	0.9921	0.9847	0.9846
Naïve Bayes	0.9584	0.9823	0.9257	0.9531	0.9584
Random Forest	**0.9959**	0.9943	0.9976	0.9959	0.9959
Support Vector Machine	0.9954	0.9943	0.9964	0.9954	0.9954

Note: The best results across all models are highlighted in bold.

4.1.2. URL Content Model Analysis Using ML

Random Forest consistently delivered the highest performance across imbalanced and balanced datasets. It achieved a good accuracy of 0.9947 and 0.9959, respectively, along with consistently high precision and recall, as shown in Table 4. In contrast, SVM achieved accuracies of 0.9935 on the imbalanced set and 0.9954 on the balanced set. Gradient Boosting had the lowest performance in both datasets, despite an improvement being noticeable with balancing.

4.1.3. ROC Curves of ML Models on Email and URL Data

There was not a significant difference among the models, as shown in Figure 7, as all of them showed a high ability to distinguish between phishing and non-phishing emails. Logistic Regression, Random Forest, and Support Vector Machine all displayed comparable results. The ROC curve for email classifiers illustrates that all models performed exceptionally well, with AUC values above 0.96. Support Vector Machine (AUC: 0.9979), Random Forest (AUC: 0.9989), and Logistic Regression (AUC: 0.9972) achieved the highest scores, while Naïve Bayes and Gradient Boosting also performed well; Decision Tree had the lowest score (AUC: 0.9695). Overall, SVC and Random Forest were the most reliable for phishing email classification.

In contrast, the ROC curve for URL classifiers demonstrated strong performance in most models, with Random Forest (AUC: 0.9931) and Gradient Boosting (AUC: 0.9923) leading. K-NN and SVC also performed well, demonstrating high sensitivity and low false positive rates. Logistic Regression had the lowest (AUC: 0.8584), indicating challenges in distinguishing between classes. Naïve Bayes performs less effectively. Overall, ensemble models like Random Forest and Gradient Boosting are the most reliable for URL-based phishing detection.

Table 4. Comparison of ML Results for URL Dataset (Balanced vs. Imbalanced).

Imbalanced URL Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
Decision Tree	0.9837	0.9606	0.9690	0.9648	0.9785
Gradient Boosting	0.9738	0.9338	0.9537	0.9437	0.9667
K-Nearest Neighbors	0.9761	0.9387	0.9591	0.9488	0.9702
Logistic Regression	0.8147	0.5619	0.8921	0.6895	0.8418
Naïve Bayes	0.8726	0.6817	0.8396	0.7525	0.8611
Random Forest	**0.9881**	0.9668	0.9822	0.9745	0.9861
Support Vector Machine	0.9601	0.8895	0.9442	0.9160	0.9545
Balanced URL Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
Decision Tree	0.9843	0.9829	0.9857	0.9843	0.9843
Gradient Boosting	0.9713	0.9662	0.9767	0.9741	0.9713
K-Nearest Neighbors	0.9748	0.9692	0.9807	0.9749	0.9748
Logistic Regression	0.8415	0.8111	0.8905	0.8489	0.8415
Naïve Bayes	0.8835	0.8767	0.8925	0.8846	0.8835
Random Forest	**0.9984**	0.9855	0.9915	0.9885	0.9884
Support Vector Machine	0.9619	0.9605	0.9634	0.9620	0.9619

Note: The best results across all models are highlighted in bold.

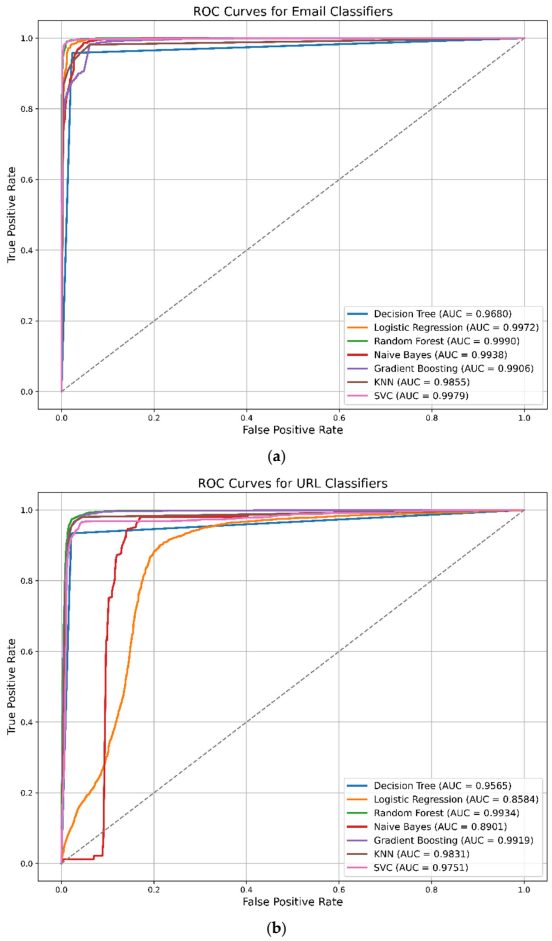


Figure 7. (**a**) ROC curve for Email Classifiers in ML; (**b**) ROC curve for URL Classifiers in ML.

4.2. Results for LLMs Approaches

4.2.1. Email Content Analysis Using LLMs

LLMs demonstrated exceptional performance across both imbalanced and balanced datasets. DistilBERT achieved the highest accuracy at 0.9844 on the imbalanced dataset and 0.9835 on the balanced dataset, followed closely by ALBERT and MiniLM. RoBERTa and ELECTRA-Tiny also performed well, with accuracy above 0.95. Although BERT-Tiny had the lowest accuracy at 0.9203 on the imbalanced dataset and 0.9198 on the balanced data, it still performed reasonably well considering its smaller size and faster inference speed, as shown in Table 5.

Table 5. Comparison of LLMs Results for Email Dataset (Balanced vs. Imbalanced).

Imbalanced Email Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
ALBERT	0.9797	0.9621	0.9678	0.9650	0.9762
BERT-Tiny	0.9203	0.8530	0.8719	0.8623	0.9055
DistilBERT	**0.9844**	0.9771	0.9686	0.9728	0.9797
ELECTRA-Tiny	0.9704	0.9632	0.9331	0.9480	0.9593
MiniLM	0.9769	0.9750	0.9442	0.9594	0.9672
RoBERTa	0.9743	0.9657	0.9446	0.9550	0.9655
Balanced Email Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
ALBERT	0.9782	0.9775	0.9789	0.9782	0.9782
BERT-Tiny	0.9198	0.9143	0.9264	0.9203	0.9198
DistilBERT	**0.9835**	0.9846	0.9823	0.9835	0.9835
ELECTRA-Tiny	0.9632	0.9614	0.9653	0.9653	0.9632
MiniLM	0.9741	0.9743	0.9738	0.9740	0.9741
RoBERTa	0.9708	0.9809	0.9604	0.9705	0.9708

Note: The best results across all models are highlighted in bold.

4.2.2. URL Content Analysis Using LLMs

Models exhibited excellent performance on both imbalanced and balanced datasets. All models achieved an accuracy above 0.9960, indicating a strong ability to capture structural and lexical patterns in URLs. DistilBERT achieved the best accuracy of 0.9979 on the imbalanced dataset and 0.9971 on the balanced dataset. MiniLM and ELEC-TRA-Tiny achieved identical accuracy scores, and the dataset has strong competitors as well. Even BERT-Tiny, despite being the smallest model, remained competitive with 0.9965 on the imbalanced dataset and 0.9947 on the balanced dataset as a good option for lightweight applications. Overall, all LLMs offered both accuracy and reliability across different data distributions, as detailed in Table 6.

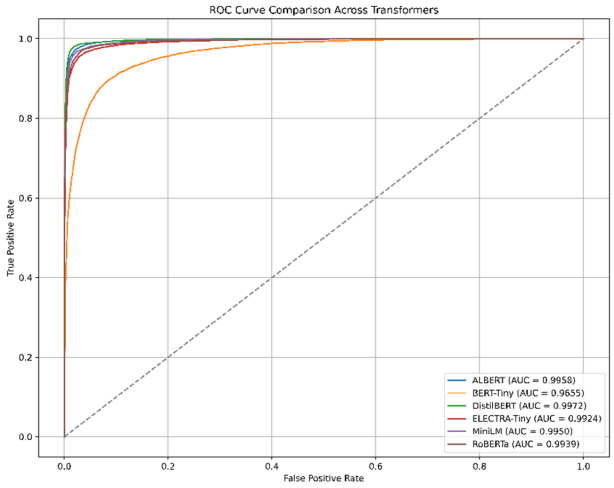
4.2.3. ROC Curves of LLMs on Email and URL Data

The ROC curve comparison, illustrated in Figure 8, shows that all transformer models exhibited strong performance on email phishing detection, achieving AUC scores close to 1. DistilBERT had the highest (AUC: 0.9972) while those of ALBERT, MiniML, and RoBERTa were also very close. Even though BERT-Tiny had lower discrimination power (AUC: 0.9655), it still showed good performance for this task. Overall, the ROC curves indicate that all the models had strong classification skills with a low rate of false positives and a high rate of true positives.

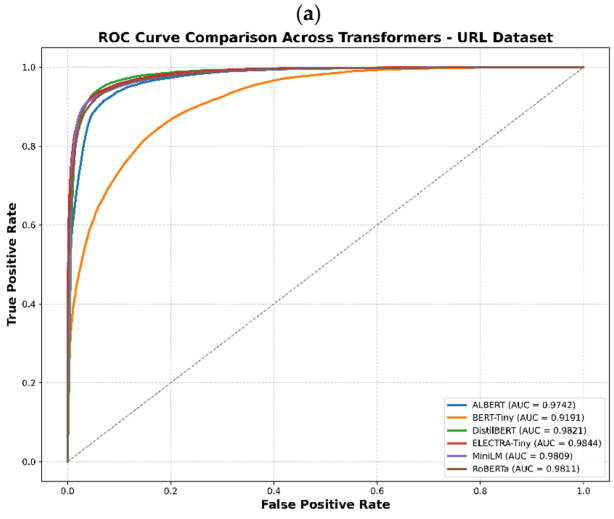
Table 6. Comparison of LLMs Results for URL Dataset (Balanced vs. Imbalanced).

Imbalanced URL Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
ALBERT	0.9974	0.9967	0.9922	0.9945	0.9956
BERT-Tiny	0.9965	0.9916	0.9932	0.9924	0.9953
DistilBERT	**0.9979**	0.9970	0.9938	0.9954	0.9965
ELECTRA-Tiny	0.9975	0.9967	0.9926	0.9947	0.9958
MiniLM	0.9975	0.9961	0.9933	0.9947	0.9960
RoBERTa	0.9977	0.9977	0.9926	0.9952	0.9960
Balanced URL Dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced-Accuracy**
ALBERT	0.9961	0.9964	0.9958	0.9961	0.9961
BERT-Tiny	0.9947	0.9966	0.9927	0.9947	0.9947
DistilBERT	**0.9971**	0.9979	0.9963	0.9971	0.9971
ELECTRA-Tiny	0.9963	0.9967	0.9959	0.9963	0.9993
MiniLM	0.9963	99.81	99.46	99.63	99.63
RoBERTa	0.9968	0.9979	0.9957	0.9968	0.9968

Note: The best results across all models are highlighted in bold.



(**a**)



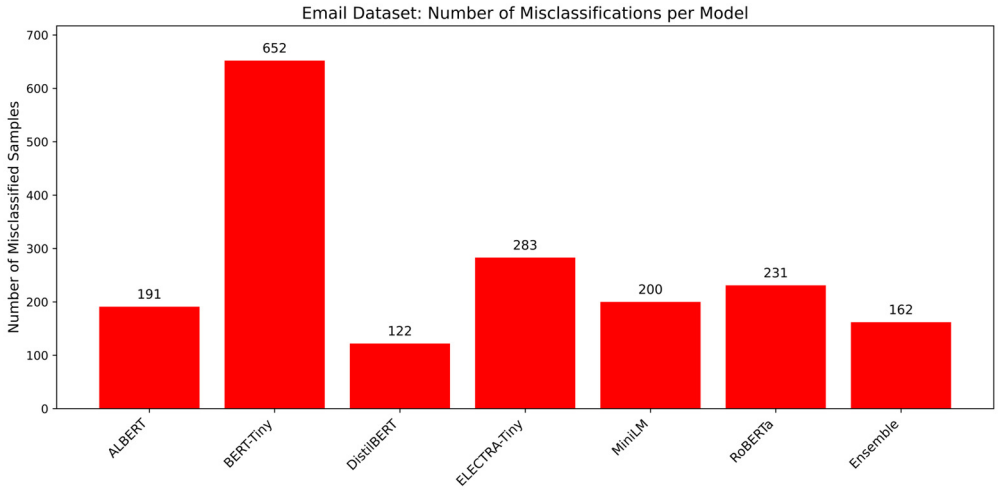
(**b**)

Figure 8. (**a**) ROC curve for Email Dataset in LLMs; (**b**) ROC curve for URL Dataset in LLMs.

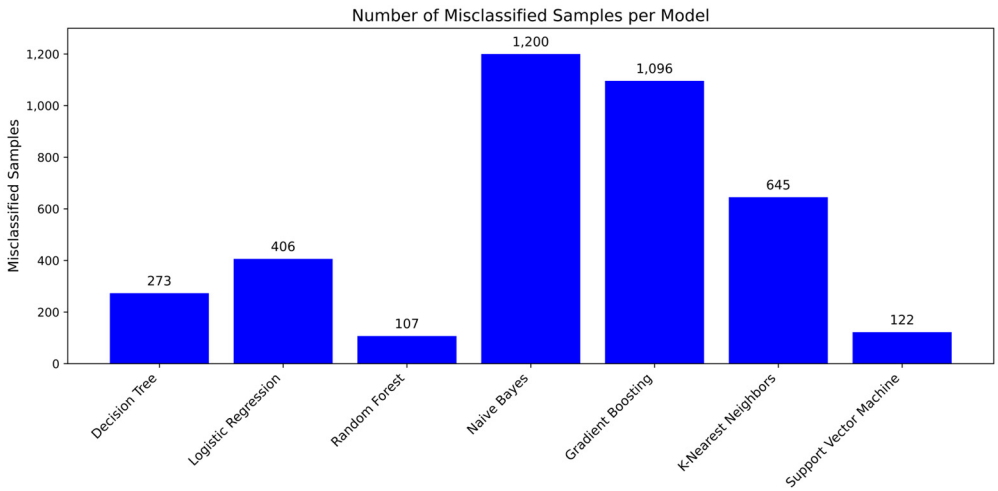
In URL, the ROC curve demonstrates strong performance across all transformer models. DistilBERT again leads with the highest score (AUC: 0.9972), followed closely by MiniLM, ELECTRA-Tiny, ALBERT, and RoBERTa. BERT-Tiny showed relatively lower performance (AUC: 0.9191) but still maintained acceptable classification ability.

4.3. Analysis of Misclassifications

Figure 9 presents the quantitative values of misclassification counts for each model during the evaluation phase, providing a clear basis for comparing model performance. The analysis was conducted using the cleaned emails.csv dataset, which contains 26,365 record samples. Both ML algorithms and LLMs were evaluated to determine their total number of incorrect classifications.



(**a**)



(**b**)

Figure 9. (**a**) LLM Misclassified Instances; (**b**) ML Misclassified Instances.

This analysis and results provide quantitative measures of the accuracy and reliability of each model in detecting phishing emails. Moreover, the GUI was developed to support batch predictions through an Excel file. Users can upload two Excel files: one containing labeled email samples for evaluation, and another containing unlabeled samples for prediction. The system allows the user to select either ML models or LLMs to perform the classification. After prediction, the accuracy is displayed as a percentage by comparing the predicted and actual labels. Additionally, it provides an option to download the full prediction results as an Excel sheet file.

The findings reveal that ML models averaged 550 misclassifications, while LLMs averaged only 263. An ensemble method was employed to aggregate predictions from all LLMs by averaging their outputs. These results indicate that LLMs achieved a 52.18% reduction in misclassifications, calculated using the following formula:

$$\textbf{Reduction Percentage} = \frac{\textbf{ML errors} - \textbf{LLMs errors}}{\textbf{ML errors}} \times \mathbf{100} \quad (6)$$

To evaluate the benefit of using an ensemble strategy with LLMs, the number of misclassified samples produced by the ensemble model was used from the total of 26,365 email samples; the ensemble model misclassified only 162 instances. The misclassification rate was calculated using the formula:

$$\textbf{Misclassification Rate} = \frac{\textbf{Misclasified Instance}}{\textbf{Total Samples}} \times \mathbf{100} \quad (7)$$

This low misclassification rate (approximately 0.61%) illustrates the strength of the ensemble method, which combines predictions from all individual LLMs by averaging their outputs.

4.4. Prediction Error Analysis

4.4.1. Prediction Error in ML Models

Upon reviewing the failed email classification predictions, several patterns were identified that impacted the model's accuracy. Correspondence that contained grammatical or informal writing styles was commonly misclassified as phishing. This is probably due to the ML models' low weighting of these terms, misspellings, and their inability to comprehend context.

Detecting email content formatted in HTML poses another challenge for ML models, which often misclassifies it as legitimate due to the scarcity of such examples in the training data. This scarcity makes it difficult for the model to accurately determine if a phishing attempt is using HTML or malicious scripts. In addition, some misclassifications were related to mixed language content, in which an email comprising English in addition to one or more other languages, making it difficult for the model to understand the message and classify it accurately. As shown in Table 7, traditional machine learning models often underperform when handling emails with grammatical errors, multilingual text, or HTML-formatted content.

Table 7. Examples of Misclassified Emails by ML Models.

Text	Actual Label	Predict Label	Model
Grammatical Errors			
Dear user, ur accnt info is missing plz verify fast.	legitimate	phishing	DT, LG, RF, SVC, KNN
HTML-format			
You are a winner, your phone is not among the <200> lucky winners' code Call Michael JOHN on: <08167566152> for a claim	phishing	legitimate	LG, GB
Multilingual Text			
Hello, Por favor, update your password to keep your account secure. Gracias.	phishing	legitimate	DT, NB, GD

4.4.2. Prediction Errors in LLMs

Even though LLMs generally outperform traditional classifiers in contextual understanding, they still struggle with a noticeable number of misclassified emails. This is

especially evident in cases where emails containing common phrases like 'password', 'login', or 'click the link' were incorrectly classified as phishing. As shown in Table 8, the issue appears to be misclassified. Emails with formal or lengthy formats, including multiple line breaks, are often misclassified as phishing. This suggests the model may be focusing on the structure or formatting of the email, rather than its actual content, leading to misclassification.

Table 8. Examples of Misclassified Emails by LLMs.

Text	Actual Label	Predict Label	Model
Common words			
Subject: Password Reset Request—University Portal Dear Student, We received your request to reset the **password** for your University Portal **login** account. To proceed, please **click the link** below to create a new password: https://example.com/university/reset-password?token=SIM-2025-EMAIL (accessed 23 September 2025) If you did not request this change, please ignore this email or contact IT Support immediately. Sincerely, University IT Support Team it-support@university.edu	legitimate	phishing	ALBERT, ELECTRA, MiniLM, RoBERTa
Punctuations Marks			
Welcome! Ready to code with fresh updates from Tech Insight? Doesn't look right? Just click here! Courses, Tools, Tutorials...	legitimate	phishing	DistilBERT, ELECTRA, MiniML
Formal or Lengthy Formats			
I saw your advertisement and I must say, the item looks exactly like what I've been looking for. The pictures are clear and the description is satisfactory. Please send me the exact current condition, any issues I should be aware of, and the final asking price. Regarding payment, I would prefer to use PayPal, as it is quick and secure for both of us. Once I make the payment, I will arrange for a private courier service to come to your location for pickup. They will handle everything signing any documents and collecting the item. There's no need for you to worry about shipping or extra costs. Just let me know your PayPal email address so I can proceed immediately. If you're not already using PayPal, you can easily register at www.paypal.com it takes a minute. Please also include your full name and pickup address in your reply so my courier can coordinate properly. Looking forward to your response. Kind regards, Derek Mason	phishing	legitimate	BERT-Tiny, RoBERTa

4.5. Impact of Dataset Balancing on Accuracy

After performing the balancing, there was a small increase in accuracy and balanced accuracy for the traditional machine learning models when classifying both emails and URLs. The slight improvements were likely due to the fact that traditional ML models are more significantly impacted by class imbalance. By balancing the dataset, the models were able to learn more equally from examples of phishing and legitimate, as opposed to only learning more from the majority class. In contrast, LLMs showed a slight accuracy decline. As shown in this study, "*Understanding the Effects of Language-Specific Class Imbalance in Multilingual Fine-Tuning*" [34], LLMs may overfit on added synthetic or underrepresented samples due to their sensitivity to nuanced patterns, leading to a small drop in raw accuracy.

It is noticeable that the values of accuracy and balanced accuracy become identical. This is because both classes, phishing and legitimate, are equally represented, eliminating any bias toward a majority class. This alignment indicates that the evaluation is fair and not influenced by class imbalance.

4.6. Impact of Dataset Size on Real-Time

A significant limitation of this research is that both the initial email and URL datasets were heavily reduced to fit within local computational resource limits. While this allowed us to train and evaluate the models effectively, it likely restricted the diversity and representativeness of our training data. Consequently, the results should be interpreted with caution when applying them to real-world scenarios that involve much larger and more diverse datasets. Models trained on such extensive data could potentially be more robust and better at capturing the full range of variability in both phishing attacks and legitimate communication patterns. Future research will build on this work by incorporating larger datasets and utilizing more powerful computing resources further to assess the scalability and real-world relevance of the approach. Prior studies, such as [35], have demonstrated that scalable machine learning approaches trained on larger datasets can achieve improved robustness and real-time detection capabilities

5. Discussion

5.1. Comparative the Effectiveness of ML and LLMs

The study indicated that LLMs like DistilBERT (accuracy: 0.9844) and ALBERT (accuracy: 0.9797) performed strongly. However, ML models like Random Forest (accuracy: 0.9947) and SVM (accuracy: 0.9935) slightly outperformed them in raw accuracy on email and an imbalanced dataset. Despite this, LLMs demonstrated superior performance on semantically complex emails, particularly in handling language variations and contextual cues.

5.2. Computational Training Demands ML vs. LLMs

ML models like Logistic Regression and Random Forest trained quickly with minimal resources, LLMs such as ALBERT and DistilBERT required more time and computational resources due to their complex architectures. As shown in Table 9, ML models completed training in under 10 min, while LLMs required several hours to days, depending on the dataset and balance. These results emphasize the actual trade-off between performance and resource requirements when deploying phishing detection systems. All models were fine-tuned locally on a personal machine rather than using cloud platforms like Google Colab, due to frequent session runtime interruptions and limitations. This ensured stable training for large language models and avoided disruptions that could compromise performance or reproducibility.

5.3. Handling Complexity Content

LLMs can accurately classify valid emails with complex structures and different language patterns, even when URLs are embedded. Valid emails of this sort often resemble phishing attempts due to their length, formal tone, or external links. These factors usually lead ML models to misclassify them. The following is an example of a complex, well-structured but legitimate message containing an embedded URL (Box 1). Most LLMs correctly identified it as legitimate, such as DistilBERT. Meanwhile, models like BERT-Tiny and RoBERTa misclassified it as phishing, demonstrating their stronger contextual understanding and ability to handle nuanced email content.

Table 9. Training Duration for ML models and LLMs (Before and after Balancing).

Models Name	Training Time (hh: mm: ss)	
	Imbalanced Set	**Balanced Set**
ML models		
LR, RF, SVC, DT, Naïve Bayes, Gradient Boosting, K-NN.	00:09:33	00:05:19
LLMs/Email Dataset		
ALBERT	36:07:01	21:35:14
BERT-Tiny	00:18:11	00:22:13
DistilBERT	26:54:40	19:10:18
ELECTRA-Tiny	07:41:02	14:38:43
MiniLM	13:35:01	12:47:20
RoBERTa	08:37:39	09:25:17
Total Time required	3 d 21 h 13 m 34 s	3 d 5 h 59 m 5 s
LLMs/URL Dataset		
ALBERT	36:35:29	18:32:08
BERT-Tiny	00:14:58	00:48:46
DistilBERT	10:04:18	14:56:19
ELECTRA-Tiny	05:08:03	04:32:46
MiniLM	22:49:26	06:42:32
RoBERTa	06:31:53	07:17:41
Total Time required	3 d 9 h 24 m 7 s	2 d 4 h 50 m 12 s

Box 1. Example of an Email Order Confirmation.

Dear Robert Smith,
We're excited to let you know that your recent order with NovaTech Electronics has been shipped and is on its way to you! Your order number is 847291037, placed on 21 June 2025, and includes a Logitech MX Master 3S Wireless Mouse—Graphite and an Anker 737 Power Bank (PowerCore 24K). Your shipment is via Express (2–3 Business Days) with an estimated arrival on 26 June 2025. You can track your package anytime here: https://www.amazon.com/progress-tracker/package/ref=ppx_yo_mob_b_track_package?orderId=847291037 (accessed 23 September 2025).
Your payment method is Visa ending in 3129, and the total charged is $189.47. To download your invoice, visit your account dashboard on NovaTech Electronics. If you have any questions or concerns, please visit our Support Center or contact us at support@novatechelectronics.com.
Thank you for shopping with NovaTech Electronics—we appreciate your business and hope you enjoy your new tech!

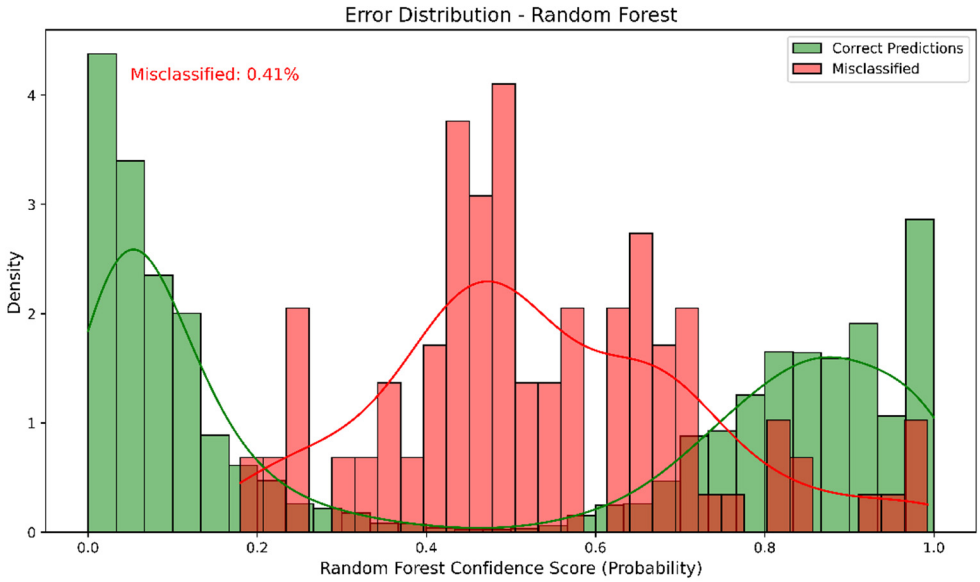
5.4. Distribution Error

This section analyzes misclassification patterns by reviewing the errors from the top-performing models. The misclassification rate was calculated by the formula:

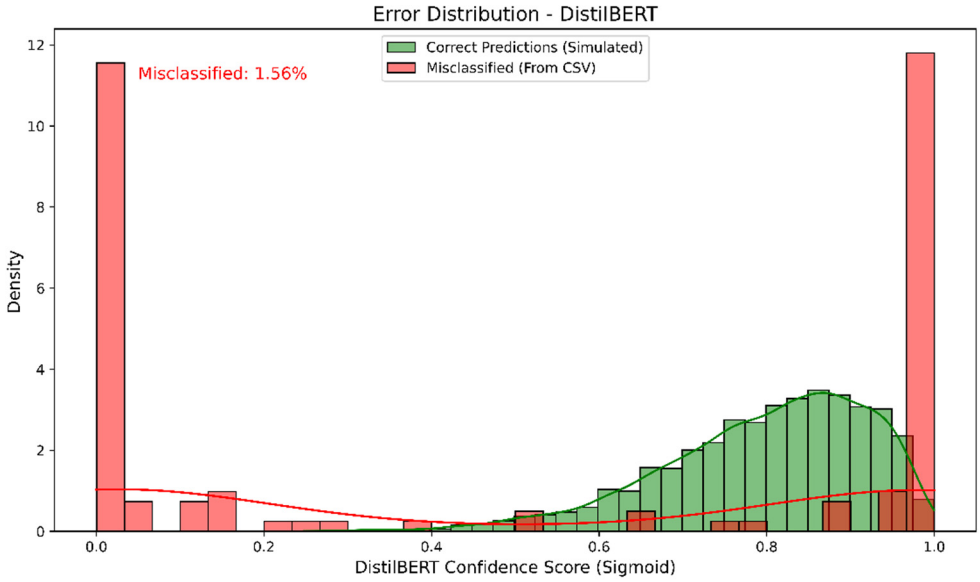
$$\textit{Error Distribution (\%)} = (1 - \textit{Accuracy}) \times 100 \quad (8)$$

Applying this equation, the Random Forest classifier achieved the highest accuracy of 99.59%, corresponding to an error rate of approximately 0.41%. While the SVM classifier performed similarly, attaining 99.54% accuracy (0.46% error rate). In contrast, the DistilBERT model reached a slightly lower accuracy of 98.44%, equating to an error rate of 1.56% this higher error rate reflects the increased complexity of LLMs in handling nuanced language structures. Figure 10 illustrates the distribution of errors across classes the green

bars show correct predictions at high confidence levels. In contrast, red bars indicate misclassified cases.



(**a**)



(**b**)

Figure 10. (**a**) Error Distribution for Top ML (RF); (**b**) Error Distribution for Top LLMs (DistilBERT).

6. Conclusions and Future Research Suggestions

6.1. Conclusions

This study investigated phishing email detection using standard ML models and transformer-based-architecture LLMs. A combined dataset of 26,365 emails and 449,271 URLs, collected from publicly available sources, was used to test both balanced and imbalanced cases. The results reported here are based on imbalanced datasets, which reflect real-world scenarios. Email and URL models were trained and fine-tuned separately to ensure accurate evaluation and specialization. The investigation involved a comparison of several ML models (Decision Tree, Random Forest, SVM, and Naïve Bayes) with lightweight LLMs (DistilBERT, ALBERT, MiniLM, and RoBERTa). The models were evaluated based on precision, recall, F1 score, balanced accuracy, and overall accuracy. DistilBERT and ALBERT

demonstrate strong performance among the LLMs, with accuracies of 0.9844 and 0.9797, respectively. ML models like Random Forest and SVM had higher accuracies of 0.9947 and 0.9935 but were less effective in handling complex patterns in phishing emails. Naïve Bayes had the lowest accuracy at 0.9521 and struggled with recall, misclassifying phishing emails. These results were consistent across both email and URL datasets. Error analysis indicated that ML models struggled with grammatical issues, mixed languages, and content with HTML formatting. LLMs struggled with formal phrasing in their emails and common phishing words, such as "password" and "click here." However, URL-based phishing detection was highly effective when the URLs could be extracted from the email content. In conclusion, both approaches were practical, with LLMs having better context handling and ML models providing high precision with lower computational demands.

6.2. Future Research Suggestions

Future work should focus on testing phishing detection models in real-time email systems. This study evaluated models offline using static datasets. Deploying and monitoring models in live environments would help assess their actual effectiveness and speed. Combining ML models with LLMs through ensemble techniques may also improve detection accuracy by leveraging their strengths together. Furthermore, including multimodal data such as text, images, and metadata could further boost detection performance by providing a deeper understanding of email content.

Author Contributions: Conceptualization, L.A. and W.M.; methodology, L.A., O.M.; software, L.A.; validation, S.T., S.A. and Y.H.; formal analysis, O.M., L.A.; investigation, O.M., Y.H.; re-sources, S.A., S.T.; data curation, W.M.; writing—original draft preparation, L.A., O.M.; writing—review and editing, O.M., S.A.; visualization, L.A., Y.H.; supervision, S.T., O.M.; project administration, S.T.; funding acquisition, S.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The datasets used and analyzed during this study are publicly available from multiple repositories. The curated phishing email datasets can be accessed via Zenodo: https://zenodo.org/records/8339691 (accessed on 21 September 2025). An alternative source for phishing email datasets is provided on Hugging Face: https://huggingface.co/datasets/zefang-liu/phishing-email-dataset. Additionally, Human-LLM generated phishing and legitimate emails are available on Kaggle: https://www.kaggle.com/datasets/francescogreco97/human-llm-generated-phishing-legitimate-emails?select=llm-generated. The phishing URL dataset can be accessed through Mendeley Data: https://data.mendeley.com/datasets/vfszbj9b36/1. All datasets are openly available for research purposes.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
AOL	America Online
AUC	Area Under the Curve
APWG	Anti-Phishing Working Group
CyBOK	Cyber Security Body Of Knowledge
DL	Deep Learning
DT	Decision Trees
FBI	Federal Bureau of Investigation's
HTML	Hyper Text Markup Language
IC3	Internet Crime Complaint Center

GPU	Graphics Processing Unit
GUI	Graphical User Interface
K-NN	K-Nearest Neighbors
LLMs	Large Language models
LR	Logistic Regression
ML	Machine Learning
NLP	Natural Language Processing
RF	Random Forest
ROC	Receiver Operating Characteristic

Appendix A

This research used various publicly available datasets to detect phishing and legitimate content. The datasets were sourced from trusted open-access repositories, including Zenodo, Kaggle, Hugging Face, and Mendeley Data. Below is a summary of the datasets analyzed. Email Datasets: Contain email texts labeled as either phishing or legitimate.

Table A1. Email Datasets from public resources.

No.	Dataset Name	Phishing	Legitimate	Total	File Size
Zenodo					
1	Enron	17,171	16,545	33,716	44.8 MB
2	Nigerian_Fraud	5186	6742	11,928	9.2 MB
3	LingSpam	481	2412	2893	9.3 MB
4	Nazario	1561	1454	3015	7.8 MB
5	SpamAssassin	1662	1135	2797	14.9 MB
6	TREC-06	3988	12,393	16,381	41.9 MB
Hugging Face					
1	Phishing-huggingface	32,702	44,975	77,677	5.71 MB
Kaggle					
1	LLMs-Generated Emails	1000	1000	2000	1.28 MB

Zenodo link: Phishing Email Curated Datasets (https://zenodo.org/records/8339691, accessed 21 September 2025). Hugging Face link: zefang-liu/phishing-email-dataset·Datasets at Hugging Face (https://huggingface.co/datasets/zefang-liu/phishing-email-dataset). Kaggle link: Human-LLM generated phishing-legitimate emails (https://www.kaggle.com/datasets/francescogreco97/human-llm-generated-phishing-legitimate-emails?select=llm-generated).

URL dataset: Contains website URLs labeled as either phishing or legitimate.

Table A2. URL Dataset from public resources.

No.	Dataset Name	Phishing	Legitimate	Total	File Size
Mendeley Repository					
1	URL Dataset	104,438	345,738	450,176	8.79 MB

Mendeley Repository link: Phishing URL dataset-Mendeley Data (https://data.mendeley.com/datasets/vfszbj9b36/1).

Appendix B

Feature extraction converts raw data into meaningful representations that machine learning models can use effectively. By identifying key patterns while minimizing noise and redundancy, well-designed features enhance model accuracy and interpretability. In phishing detection, extracting structured features from unstructured emails or URLs, such as lexical, syntactic, and semantic attributes, helps models better distinguish between

legitimate and malicious content. All features used for phishing detection in this study are summarized in the table below.

Table A3. Feature Categories for Email Classification.

No.	Feature	Description
1	Length of URL	Total length of the URL string.
2	Number of dots in the URL	Counts the number of periods in the URL that might indicate subdomains or unusual domain names.
3	Number of slashes in the URL	The count of slashes in the URL, excluding the protocol (http://or https://).
4	Presence of 'www'	Checks whether "www" is present in the domain.
5	Presence of HTTP/HTTPS	Identifies if the URL starts with "http" or "https".
6	Presence of a query string (?)	Checks for a query string in the URL, which indicates phishing attempt parameters.
7	Presence of a fragment (#)	Indicates if the URL contains a fragment, which is often used in phishing attempts to confuse users.
8	Number of query parameters	Counts the number of query parameters in the URL (indicated by ? and &).
9	Domain name length	Length of the domain name (excluding protocol and path).
10	Number of subdomains	Counts the number of subdomains in the URL's domain. More subdomains might suggest a suspicious URL.
11	Presence of a port number	Indicates if the domain contains a port number (e.g., example.com:8080).
12	Presence of an IP address	Detects if the domain is an IP address rather than a domain name, which is often used in phishing.
13	Number of uppercase letters	Counts the uppercase letters in the URL, as phishing URLs sometimes use unusual capitalization.
14	Number of digits	Counts the number of digits in the URL, often used in phishing attempts to mimic legitimate URLs.
15	Presence of special characters	Counts the special characters, which could be used to mislead or confuse users.
16	Top-level domain (TLD) length	The length of the top-level domain (e.g., .com, .org) can be indicative of domain legitimacy.
17	Length of the domain name	Length of the domain name without subdomains.
18	Length of the path	The length of the URL path after the domain.
19	Number of parameters in the path	Counts how many parameters are in the URL path, often used in phishing to mask malicious content?
20	Number of subdirectories in the path	Identifies how many subdirectories are in the URL path, often a characteristic of phishing sites.
21	Presence of a secure connection	Checks if the URL uses HTTPS, indicating a secure connection (or lack thereof).
22	Presence of login/register keyword	Detects if the URL contains "login" or "register", which might be used for phishing login pages.
23	Number of underscores	Counts the number of underscores in the URL, as phishing URLs may contain underscores to imitate legitimate domains.
24	Presence of specific keywords (login, admin, secure)	Flags URLs containing specific keywords often found in phishing attempts, like "admin" or "login".
25	Presence of file extensions	Checks if the URL ends with specific file extensions (e.g., .php, .html), common in phishing pages.
26	Presence of a session ID	Detects the presence of session identifiers in the URL, which could be used in phishing attacks.

Table A3. *Cont.*

No.	Feature	Description
27	Ratio of digits to characters	Measures the ratio of digits to other characters in the URL, which can help identify irregular or suspicious URLs.
28	Ratio of uppercase to lowercase letters	Measures the ratio of uppercase to lowercase letters, as phishing URLs often use odd capitalization patterns.
29	URL entropy (complexity)	Measures the entropy (complexity) of the URL, which can indicate whether the URL is randomly generated or suspiciously complex.
30	Presence of an email address	Checks if the URL contains an email address, which is often seen in phishing URLs to capture user data.

Appendix C

This study used a variety of traditional machine learning classifiers, each with different learning strategies and strengths. By applying this range of models, we aimed to capture different perspectives in classification, evaluate their suitability for phishing detection tasks, and establish reliable baselines for comparison with advanced methods such as large language models.

Table A4. Description of ML Models.

ML Models	Description
Decision Tree (DT)	A nonparametric model that classifies data by recursively splitting it into nodes based on impurity measures, ending with tree building and pruning phases.
Logistic Regression (LR)	A simple, widely used model for binary classification based on the logit function.
Random Forest (RF)	An ensemble model of multiple decision trees, where each tree votes for the most common class using randomly selected data and attributes.
Naïve Bayes (NB)	A probabilistic classifier based on Bayes' theorem, assuming that all features are independent. It is efficient, especially with large datasets.
Gradient Boosting (GB)	An ensemble model that builds predictors sequentially, with each new model correcting errors from the previous ones. It is powerful but sensitive to overfitting.
K-Nearest Neighbors (KNN)	A distance-based classifier that labels data using the majority vote of the K nearest neighbors, known for its simplicity and efficiency.
Support Vector Machine (SVM)	A supervised ML algorithm that finds the optimal hyperplane to classify data points by maximizing the margin between different classes.

Appendix D

This appendix provides a summary of transformer-based LLMs used in phishing email detection studies. The models vary in size and architecture: some are smaller and faster, while others are larger and more effective at capturing contextual information. The table below outlines each model's main features and includes links to its official Hugging Face page for further details.

Table A5. LLM Architecture Details.

Model Name	Size	Architecture Details	Hugging Face References
ALBERT	~12 M parameters	Shares parameters across all 12 layers and uses factorized embeddings (128 embedding size, 768 hidden size). Reduces redundancy and model size.	albert/albert-base-v2·Hugging Face (https://huggingface.co/albert/albert-base-v2)
BERT-Tiny	~4 M parameters	Minimal BERT variant with only 2 layers, 128 hidden size, and 2 attention heads. Designed for extremely lightweight tasks	prajjwal1/bert-tiny·Hugging Face (https://huggingface.co/prajjwal1/bert-tiny)

Table A5. *Cont.*

Model Name	Size	Architecture Details	Hugging Face References
DistilBERT	~66 M parameters	6-layer transformer distilled from BERT base (12 layers), with 768 hidden size. Offers ~97% of BERT's performance with a smaller size and faster inference.	distilbert/distilbert-base-uncased·Hugging Face (https://huggingface.co/distilbert/distilbert-base-uncased)
ELECTRA	~14 M parameters	12-layer transformer with 256 hidden size. Uses replaced token detection (generator/discriminator setup) for more sample-efficient training than BERT.	google/electra-small-generator·Hugging Face (https://huggingface.co/google/electra-small-generator)
MiniLM	~33 M parameters	12-layer transformer with a smaller hidden size (384). Trained using knowledge distillation from larger models. Balances speed and performance well.	microsoft/MiniLM-L12-H384-uncased·Hugging Face (https://huggingface.co/microsoft/MiniLM-L12-H384-uncased)
RoBERTa	~4 M parameters	A tiny version of RoBERTa trained on the IMDB dataset. Consists of 2 transformer layers, 128 hidden size and is designed for fast and lightweight tasks.	AntoineB/roberta-tiny-imdb·Hugging Face (https://huggingface.co/AntoineB/roberta-tiny-imdb)

References

1. Ripa, S.P.; Islam, F.; Arifuzzaman, M. The emergence threat of phishing attack and the detection techniques using machine learning models. In Proceedings of the 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), Online, 8–9 July 2021; pp. 8–9.
2. Rashed, S.; Ozcan, C. A Comprehensive Review of Machine and Deep Learning Approaches for Cyber Security Phishing Email Detection. *Al-Iraqia J. Sci. Eng. Res.* **2024**, *3*, 1–12. [CrossRef]
3. Mittal, K.; Gill, K.S.; Chauhan, R.; Joshi, K.; Banerjee, D. Blockage of Phishing Attacks Through Machine Learning Classification Techniques and Fine Tuning its Accuracy. In Proceedings of the 2023 3rd International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 29–31 December 2023; pp. 1–5.
4. Kaddoura, S.; Alfandi, O.; Dahmani, N. A Spam Email Detection Mechanism for English Language Text Emails Using Deep Learning Approach. In Proceedings of the 2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Bayonne, French, 10–12 June 2020; pp. 193–198.
5. Koide, T.; Fukushi, N.; Nakano, H.; Chiba, D. Detecting Phishing Sites Using ChatGPT. *arXiv* **2023**, arXiv:2306.05816. [CrossRef]
6. Franchina, L.; Ferracci, S.; Palmaro, F. Detecting phishing e-mails using text mining and features analysis. *CEUR Workshop Proc.* **2021**, *2940*, 106–119.
7. Salloum, S.; Gaber, T.; Vadera, S.; Shaalan, K. A Systematic Literature Review on Phishing Email Detection Using Natural Language Processing Techniques. *IEEE Access* **2022**, *10*, 65703–65727. [CrossRef]
8. CyBOK. University of Bristol. The Cyber Security Body of Knowledge (Version 1.1). 2021. Available online: https://www.cybok.org/knowledgebase1_1/ (accessed on 30 May 2025).
9. Chanti, S.; Chithralekha, T. A literature review on classification of phishing attacks. *Int. J. Adv. Technol. Eng. Explor.* **2022**, *9*, 446–476. [CrossRef]
10. 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*, 36429–36463. [CrossRef]
11. Aljofey, A.; Jiang, Q.; Qu, Q.; Huang, M.; Niyigena, J.P. An effective phishing detection model based on character level convolutional neural network from URL. *Electronics* **2020**, *9*, 1514. [CrossRef]
12. Al-Subaiey, A.; Al-Thani, M.; Abdullah Alam, N.; Antora, K.F.; Khandakar, A.; Uz Zaman, S.A. Novel interpretable and robust web-based AI platform for phishing email detection. *Comput. Electr. Eng.* **2024**, *120*, 109625. [CrossRef]
13. Anti-Phishing Working Group. Phishing Activity Trends Report 4th Quarter 2023. 2023. Available online: https://docs.apwg.org/reports/apwg_trends_report_q4_2023.pdf (accessed on 3 March 2025).
14. SLASHNEXT. The State of Phishing 2023. Available online: https://slashnext.com/wp-content/uploads/2023/10/SlashNext-The-State-of-Phishing-Report-2023.pdf (accessed on 6 March 2025).
15. Jaya, T.; Kanyaharini, R.; Navaneesh, B. Appropriate Detection of HAM and Spam Emails Using Machine Learning Algorithm. In Proceedings of the 2nd IEEE International Conference on Cognitive Informatics, ACCAI 2023, Washington, DC, USA, 18–20 August 2023.

16. Khalid, A.; Hanif, M.; Hameed, A.; Smiee, Z.A.; Alnfiai, M.M.; Alnefaie, S.M.M. LogiTriBlend: A Novel Hybrid Stacking Approach for Enhanced Phishing Email Detection Using ML Models and Vectorization Approach. *IEEE Access* **2024**, *12*, 193807–193821. [CrossRef]
17. Bagui, S.; Nandi, D.; Bagui, S.; White, R.J. Machine Learning and Deep Learning for Phishing Email Classification using One-Hot Encoding. *J. Comput. Sci.* **2021**, *17*, 610–623. [CrossRef]
18. An, P.; Shafi, R.; Mughogho, T.; Onyango, O.A. Multilingual Email Phishing Attacks Detection Using OSINT and Machine Learning. *arXiv* **2025**, arXiv:2501.08723. [CrossRef]
19. Sengar, S.S.; Hasan ABin Kumar, S.; Carroll, F. Generative artificial intelligence: A systematic review and applications. *Multimed. Tools Appl.* **2024**, *84*, 23661–23700. [CrossRef]
20. Bengesi, S.; El-Sayed, H.; Sarker, M.K.; Houkpati, Y.; Irungu, J.; Oladunni, T. Advancements in Generative AI: A Comprehensive Review of GANs, GPT, Autoencoders, Diffusion Model, and Transformers. *IEEE Access* **2024**, *12*, 69812–69837. [CrossRef]
21. Han, K.; Wang, Y.; Chen, H.; Chen, X.; Guo, J.; Liu, Z.; Tang, Y.; Xiao, A.; Xu, C.; Xu, Y.; et al. A Survey on Vision Transformer. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 87–110. [CrossRef]
22. Ahmed, N.; Khomh, F. Data Cleaning and Machine Learning: A Systematic Literature Review. *Autom. Softw. Eng.* **2024**, *31*, 54. [CrossRef]
23. Zhan, Z. Comparative Analysis of TF-IDF and Word2Vec in Sentiment Analysis: A Case of Food Reviews. In Proceedings of the 2024 2nd International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2024), Nanjing, China, 6–8 December 2024; p. 02013.
24. Elkholy, M.; Sabry, M.; Elbehiery, H. An Efficient Phishing Detection Framework Based on Hybrid. *Sustain. Mach. Intell. J.* **2025**, *11*, 11–19. [CrossRef]
25. Mahendru, S.; Networks, P.A. SecureNet: A Comparative Study of DeBERTa and Large Language Models for Phishing Detection. In Proceedings of the 2024 IEEE 7th International Conference on Big Data and Artificial Intelligence (BDAI), Beijing, China, 5–7 July 2024; pp. 160–169.
26. Wood, T.; Basto-fernandes, V.; Boiten, E.; Yevseyeva, I. Systematic Literature Review: Anti-Phishing Defences and Their Application to Before-the-Click Phishing Email Detection. *arXiv* **2022**, arXiv:2204.13054. [CrossRef]
27. Jamal, S.; Wimmer, H.; Sarker, I.H. An improved transformer-based model for detecting phishing, spam and ham emails: A large language model approach. *Secur Priv.* **2024**, *7*, e402. [CrossRef]
28. Karim, A.; Shahroz, M.; Mustofa, K.; Belhaouari, S.B.; Joga, S.R.K. Phishing Detection System Through Hybrid Machine Learning Based on URL. *IEEE Access* **2023**, *11*, 36805–36822. [CrossRef]
29. Raiaan, M.A.K.; Mukta, M.S.H.; Fatema, K.; Fahad, N.M.; Sakib, S.; Mim, M.M.J.; Ahmad, J.; Ali, M.E.; Azam, S. A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges. *IEEE Access* **2024**, *12*, 26839–26874. [CrossRef]
30. Khan, S.; Naseer, M.; Hayat, M.; Zamir, S.W.; Khan, F.S.; Shah, M. Transformers in Vision: A Survey. *ACM Comput. Surv.* **2022**, *54*, 1–30. [CrossRef]
31. Naveed, H.; Khan, A.U.; Qiu, S.; Saqib, M.; Anwar, S.; Usman, M.; Akhtar, N.; Barnes, N.; Mian, A. A Comprehensive Overview of Large Language Models. *ACM Trans. Intell. Syst. Technol.* **2025**, *16*, 1–72. [CrossRef]
32. Wang, D.; Li, Y.; Jiang, J.; Ding, Z.; Luo, Z.; Jiang, G.; Liang, J.; Yang, D. Tokenization Matters! Degrading Large Language Models through Challenging Their Tokenization. *arXiv* **2025**, arXiv:2405.17067. [CrossRef]
33. Lu, Y.; Ji, Z.; Du, J.; Shanqing, Y.; Xuan, Q.; Zhou, T. From LLM-anation to LLM-orchestrator: Coordinating Small Models for Data Labeling. *arXiv* **2025**, arXiv:2506.16393. [CrossRef]
34. Jung, V.; van der Plas, L. Understanding the effects of language-specific class imbalance in multilingual fine-tuning. In Proceedings of the EACL 2024—18th Conference of the European Chapter of the Association for Computational Linguistics Find EACL 2024, St. Julian's, Malta, 17–22 March 2024; pp. 2368–2376.
35. Zia, M.F.; Heath, M.; Heath, M. Web Phishing Net (WPN): A scalable machine learning approach for real-time phishing campaign detection. *arXiv* **2025**, arXiv:2502.13171. [CrossRef]