BERT for Fraud Detection Tasks

James Sanii
Queen's University
17jcs9@queensu.ca

Deniz Aydin
Queen's University
21dha1@queensu.ca

Chris Wang
Queen's University
20ydw@queensu.ca

Matthew Pirie
Queens's University
20msp4@queensu.ca

***Abstract*—Large language models (LLMs) have emerged as powerful tools for fraud detection due to their ability to process vast amounts of textual data. We aim to show that LLMs can outperform traditional machine learning algorithms for three fraud detection tasks: fake news detection, SMS spam prediction and detecting phishing links. Our work shows that by scrutinizing linguistic patterns and sentiments, LLMs can detect anomalies, suspicious behaviours, and emerging fraud patterns with high accuracy and efficiency. BERT outperformed traditional machine learning methods for each of our tasks and has an F1-score of 0.9995, 0.9919 and 0.9606 respectively.**

I. Introduction

Predatory scams have been around since the internet was created and have aimed to prey on the trust or naivety of unsuspecting victims. Misinformation and predatory scams have become larger issues in the past few years, in large part due to the rise in social media usage. The spread of misinformation has led to the degradation of trust in the democratic process and the defamation of prominent individuals.

Misinformation spreads quickly due to the rapid dissemination of information through social media and messaging apps and it is often shared based on limited information aimed to invoke an emotional reaction out of the reader. Additionally, confirmation bias and the illusion of truth effect make individuals more likely to accept and share information that reaffirms their worldview, regardless of the accuracy of the claims made. The ability to flag misinformation promptly is essential to combat the rapid dissemination of misinformation.

Dishonest actors often use scams to enrich themselves while praying on the desperate and vulnerable. Online scams come in a variety of forms such as get-rich-quick schemes, phishing links to steal login credentials, and email spoofing where attackers mimic trusted sources to deceive recipients into disclosing sensitive information or transfer funds to fraudulent accounts. The repercussions of falling victim to scams can be devastating, resulting in financial loss, identity theft, compromised personal information, and profound emotional distress. Scams often evolve and adapt to technological advancements and changing socio-economic conditions, presenting ongoing challenges for cybersecurity experts in combating fraudulent activities and protecting public trust.

A. Motivation

We want to create a warning system to flag potentially fraudulent text to warn users to exercise caution when interacting with that piece of media. We will use machine learning methods to analyze misinformation and phishing scams to identify patterns in the general semantics of fraudulent text.

B. Problem Definition

This paper aims to show that there are general patterns in fraudulent text that machine learning models can detect.

Our projects aim to address the following: 1) Can natural language processing be used to detect fraud in news articles, Short Messaging Service (SMS), and phishing links? 2) Is there a significant performance gain from using LLMs over traditional machine learning methods for this task?

C. Contributions

We empirically investigate the viability of detecting disingenuous messages. We developed three machine learning systems to help protect users from predatory practices: an email scanner that flags any links in your inbox that are believed to be fraudulent, an SMS app that detects and filters out spam and a news summarizer that flags any article believed to be fake news.

We believe our systems show the practical application of using machine learning for cybersecurity and versions of these systems should be added to existing products.

II. Related Work

A. Fake News

Capuano et al. (2023) work delves into the pressing issue of fake news detection through the lens of machine and deep learning techniques. It underlines the insufficiency of manual fact-checking against an ever-increasing volume of information, further complicated by the scarcity of labelled datasets, the unreliability of human labellers, and a bias towards English language databases focusing primarily on political news. Our research addresses the urgent need for effective fake news detection by leveraging BERT. BERT's deep contextual capabilities allow for a deep understanding of complex linguistic patterns, addressing the review's concerns about biases inherent in traditional models. Our work builds on the review's emphasis on exploring efficient models, showcasing BERT's potential to improve news filtration systems.

Agarwal et al. (2019) investigates the effectiveness of various classifiers trained on labelled news statements using techniques like bag-of-words, n-grams, count vectorizer, and TF-IDF. The research acknowledges limitations, including the challenge of dealing with erratic data and the influence of training data on model performance. These anomalies within

the data set can have extreme effects including bias amplification, difficulty in generalization, and reduced accuracy leading to a decrease in user trust.

Using TensorFlow for fake news detection, as explored in Veeraiah et al. (2024) work demonstrates the effectiveness of TensorFlow, a comprehensive library for deep learning, in constructing models capable of analyzing datasets to identify deceptive information with the use of machine learning techniques to achieve a system capable of detecting fake news.

Khanam et al. (2021) explains how to leverage Python's scikit-learn library in the domain of cybersecurity. The utilization of Python's scikit-learn library emerges as a cornerstone for implementing machine learning models, like BERT, which are instrumental in fake news detection scenarios. The paper highlights the approach to feature extraction and vectorization using scikit-learn for fake news classification. The paper not only contributes to solving the spread of fake news but also aligns with the broader cybersecurity efforts aimed at developing adaptive, accurate, and real-time detection models.

B. SMS spam

Shirani-Mehr (2013) , aims to address the issue of detecting spam messages within SMS communications. This challenge is notably distinct from those encountered in conventional email spam filtering due to the succinct and informal nature of SMS communications. The study's main objective was to pinpoint the optimal machine learning algorithm for efficiently filtering out spam messages, taking into account SMS-specific features like short message length and casual language usage. Multinomial Naive Bayes showed significant accuracy improvements over previous works. The paper highlights the effectiveness of incorporating meaningful features, such as message length, numeric strings, dollar signs, length threshold flags, and non-alphabetic characters and symbols, to enhance classification outcomes. This study serves as a proof of concept for the feasibility and effectiveness of machine learning models in identifying SMS spam, showcasing potential strategies for enhancing spam detection in mobile communication systems.

Jain et al. (2022) delved into SMS spam detection, with a focus on comparing word embedding methodologies and sentence embedding approaches. They delineate a crucial distinction between two types of embeddings: word embeddings—such as Count Vectorizer, TF-IDF, Hashing Vectorizer, Word2Vec, and GloVe—focus on the meaning of individual words within a context whereas contextual sentence embeddings are designed to encapsulate the entire sentence into a single vector. This distinction underscores the enhanced utility of contextual sentence embeddings in SMS spam detection, given their ability to capture the full semantic scope of sentences. To generate contextual sentence embeddings, the researchers utilized BERT (Bidirectional Encoder Representations from Transformers. The successful application of BERT, particularly when used alongside neural network architectures like LSTM or BiLSTM, underscored the viability of these models for SMS spam detection. This finding serves as a key motivator for our work, highlighting the effectiveness of combining BERT with advanced neural network structures.

C. Phishing Links

Kumar et al. (2019) work highlights the effectiveness of Support Vector Machines (SVMs) alongside traditional algorithms in identifying phishing URLs, setting a foundational basis for subsequent exploration into Natural Language Processing (NLP) applications.

Recent efforts have expanded upon this foundation, utilizing NLP transformers such as BERT and ELECTRA for phishing detection. Haynes et al. (2021) exemplifies the potential of these models in enhancing real-time detection capabilities on mobile platforms, showcasing the practical applications of NLP in cybersecurity.

Further, the application of NLP techniques in phishing email detection is comprehensively surveyed in Salloum et al. (2021) . This survey delineates various NLP strategies for discerning phishing emails, emphasizing the critical role of linguistic analysis in detecting deception.

Salloum et al. (2022) further underscores the necessity for ongoing innovation in NLP methodologies. This review points out the evolving nature of phishing strategies and the imperative for adaptive detection models, a gap that the current study aims to bridge.

Together, these works illustrate the progression towards more sophisticated, NLP-based systems for phishing detection. Building on their contributions, our paper proposes a novel approach leveraging NLP to not only improve detection accuracy but also to adapt to the dynamic tactics of phishing attacks.

III. Methodology

A. Dataset

1) Fake News: The Fake News detection dataset [1] consists of four variables: the article's title, the author's name, the text of the article, and a label classifying the article's reliability. The author's name was dropped from the analysis since name semantics should not impact predictions since it could introduce racial biases if included.

The dataset has a slightly unbalanced class distribution, with a split of 48/52 between genuine and fake news articles. The total number of documents in the dataset is 44,898. On average, the average article contains 2469.11 words and the dataset contains a total of 122,002 unique words.

For preprocessing, NaN values in the 'text' column were dropped since the article text is essential for model development. Next, the TfidfVectorizer transforms the text into a TF-IDF features matrix, applying this process to both training and testing data to maintain consistency.

2) SMS spam: The SMS Spam Collection dataset[2] encompasses a total of 5,572 SMS messages. Each message within this collection has been manually categorized into 'spam' for unsolicited messages and 'ham' for genuine messages. The

[1]https://github.com/SushwanthReddy/Fake-News-Detection-using-Machine-Learning
[2]https://archive.ics.uci.edu/dataset/228/sms+spam+collection

dataset showed a notable class imbalance with a significantly higher count of 'Ham' messages (4,825) compared to 'Spam' messages (747). This imbalance reflects real-world scenarios where spam messages are less frequent than regular messages. The adoption of metrics that can handle imbalanced data such as F1-score and ROC-AUC is necessary for this problem.

An analysis of message lengths before processing revealed that the average length of 'Ham' messages was approximately 71 characters, while 'Spam' messages averaged around 139 characters. This significant difference suggests that spam messages are generally longer, potentially due to the inclusion of more detailed information, promotional content or links.

Building on these insights, we applied a series of preprocessing steps to the data designed to clean and prepare the text for analysis. This text preprocessing involved several key steps using NLTK and Scikit-learn. Initially, text was tokenized, then standardized by converting to lowercase and removing punctuation. Next, NLTK's English stopwords were eliminated to focus on significant content. Words were further normalized through PorterStemmer and WordNetLemmatizer. Finally, Scikit-learn's TfidfVectorizer transformed the text into a TF-IDF feature matrix, quantifying word importance across the corpus, readying the data for analysis or machine learning.

3) Phishing Links: The phishing link dataset[3] comprises 549,346 entries, each labelled as either "Good" (non-phishing) or "Bad" (phishing), based on the presence of malicious content within the URL. The class balance is 72% safe links to 28% phishing links. To mitigate computational constraints and ensure a balanced representation of both classes, we subsampled the dataset to include 15,000 entries from each category, thereby achieving an equal distribution between the two classes.

a) Data Preprocessing for Phishing Website Detection: Preprocessing is necessary for transforming raw URLs into a format amenable to machine learning analysis. The preprocessing pipeline outlined below was uniformly applied across all traditional machine learning models to facilitate a consistent basis for performance comparison.

b) Data Cleaning and Tokenization: The first step in our preprocessing pipeline involved cleaning each URL by removing protocols and subdomains (e.g., "http://" and "www."). This was followed by tokenization, where URLs were dissected into their constituent components using common delimiters, such as slashes ("/"), question marks ("?"), and equal signs ("="). This process breaks down the URLs into meaningful segments, enhancing the models' capacity to analyze the structural and semantic features of URLs critical for phishing detection. By isolating the segments most indicative of malicious intent, we significantly bolstered the predictive power of our models.

c) Text Vectorization and Balancing: Following tokenization, the URLs were transformed into a numerical format using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique. Selected for its proficiency in emphasizing the significance of tokens that are uniquely characteristic of phishing attempts, TF-IDF vectorization enhances the model's capability to distinguish between phishing and legitimate URLs. This choice reflects our strategic approach to highlight the most relevant features within the URLs, thereby improving the accuracy and efficiency of phishing detection.

This preparatory work lays the groundwork for our exploration into various machine learning algorithms, setting the stage for a detailed analysis of their effectiveness in identifying phishing URLs based on the nuanced features extracted through our preprocessing steps.

[3]https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls

B. Experiment Setup

1) General model training: For each experiment, 5-fold cross-validation was used to minimize the risk of overfitting and more accurately assess how the model will perform on unseen data.

For each task six different models were tested: Logistic Regression (LR), Multinomial Naive Bayes (MNB), Support Vector Machine (SVM), Random Forest (RF), AdaBoost (ADA), Bidirectional Encoder Representations from Transformers (BERT). All data prepossessing steps were applied to the data for all models except BERT where the raw text data could be inputted.

A parameter grid search was implemented to train each model to find optimal model parameters.

The performance of all models was assessed using: Accuracy (eq 1), Precision (eq 2), Recall (eq 3), F1-score (eq 5), and the Area under the ROC Curve (AUC). All equations can be found in the Appendix.

True positives (TP) count the correctly classified fraud attempts, false negatives (FN) the incorrectly classified fraud as being safe, true negatives (TN) the correctly classified safe texts and false positives (FP) the incorrect labelling of safe text as fraud. AUC is calculated as the Area Under the $Sensitivity$-$(1 - Specificity)$ Curve.

2) Fake News: To train the model for fake news detection, the model begins by loading the news article dataset using pandas. The dataset is split into training and testing partitions which are then converted to a TensorFlow Dataset. The base of this model architecture is the BERT encoder from the TensorFlow library which is a large language model commonly used for text processing. It is optimized using binary cross-entropy as the loss function. The model uses AdamW as an optimizer and sets the hyperparameters by using a learning rate of 3e-5. Finally, after training on all the folds and epochs, the model is saved for future use. This is the overall approach taken to train the fake news detection model.

3) SMS Spam: In our experimental design for SMS spam detection, we conducted a comprehensive evaluation of our BERT model against a range of traditional machine learning algorithms specified above. This comparative analysis aimed to benchmark the performance and efficacy of our BERT-based model in the context of spam detection.

Our BERT model is structured around the TensorFlow Hub's "small bert/bert en uncased L-4 H-512 A-8" variant, a

streamlined version of BERT optimized for text classification. The model's pipeline and hyperparameters are outlined in Table V, showcasing our approach to optimize BERT for spam detection.

For the traditional models, we followed a standardized training procedure using the preprocessed SMS Spam collection dataset. The dataset was divided into training and testing sets, with "tfidf features" serving as input and "data['label']" as the target labels. Finally, the performance of both our BERT-based and traditional models was rigorously evaluated using the aforementioned evaluation metrics. This evaluation provided a holistic view of each model's effectiveness in SMS spam detection, highlighting their respective strengths and areas for improvement in addressing the nuances of spam classification.

4) Website Phishing: In this study, we set out to compare the effectiveness of a BERT-based model against traditional machine learning techniques in the context of phishing website detection. Our goal was to evaluate how the newer BERT model performs in comparison to these established methods. For our deep learning approach, we selected the `small_bert/bert_en_uncased_L-4_H-512_A-8` model from TensorFlow Hub for its optimal balance between computational efficiency and accuracy in text classification. The model setup integrated an input layer for text processing, a BERT encoder for generating text embeddings, a dropout layer (set at 10%) to mitigate overfitting, and a dense output layer for binary classification (phishing or legitimate). It was trained using the AdamW optimizer with a learning rate of 3×10^{-5} over three epochs, including a warm-up phase to optimize training. For the traditional algorithms, we utilized a dataset preprocessed and divided into training and test sets, with features derived using TF-IDF vectorization and default hyperparameters from the respective libraries or specific settings based on documentation and best practices. This strategy assessed the models' out-of-the-box capabilities in detecting phishing websites.

IV. Results and Discussion

A. Fake News

On the test set, the BERT model achieved an accuracy of 0.9987, an F1 score of 0.9987, a recall of 0.9984, a precision of 0.9991, and an AUC of 0.9992. In contrast, the training set performance showcased slightly higher metrics with an accuracy of 0.9992, an F1 score of 0.9991, a recall of 0.9988, a precision of 0.9996, and an AUC of 0.9997. These results indicate a strong model performance on unseen data, with only a marginal decrease in metrics from training to testing. The close alignment of these values suggests that the model has a good balance between learning from the training data and generalizing to new, unseen data, with a slight advantage in precision and AUC on the training set, indicating a slightly better ability to distinguish between classes and predict the positive class correctly. However, the minimal differences highlight the model's effective generalization, suggesting it is well-tuned and not significantly overfitting to the training data.

A comparison between the top-performing BERT model and other machine learning models can be found in Table I. BERT was the top-performing model on the test set.

TABLE I
Performance Comparison of Fake News Detection Models

Model	Precision	Recall	F1-Score	Accuracy	AUC
MNB	0.9226	0.9180	0.9203	0.9239	0.9711
LR	0.9774	0.9822	0.9798	0.9803	0.9887
SVM	0.9808	0.9856	0.9832	0.9834	0.9893
RF	0.9858	0.9851	0.9854	0.9856	0.9898
ADA	0.9872	0.9872	0.9872	0.9873	0.9898
BERT	0.9994	0.9996	0.9995	0.9994	0.9995

B. SMS Spam

The results of our comparative analysis, presented in Table II, demonstrate that the BERT outperformed all other models tested across every evaluation metric. BERT demonstrated exceptional performance, achieving weighted Precision, Recall, F1-score, and Accuracy scores of 0.9919, alongside an AUC score of 0.9974. These key values not only underscore the model's remarkable consistency across various evaluation metrics but also highlight its robustness in accurately distinguishing between spam and non-spam messages.

TABLE II
Performance Comparison of SMS Spam Detection Models

Model	Precision	Recall	F1-score	Accuracy	AUC
MNB	0.9721	0.9719	0.9720	0.9719	0.9870
LR	0.9799	0.9797	0.9790	0.9797	0.9903
SVM	0.9807	0.9809	0.9806	0.9809	0.9873
RF	0.9796	0.9791	0.9783	0.9791	0.9938
ADA	0.9708	0.9713	0.9709	0.9713	0.9701
BERT	0.9919	0.9919	0.9919	0.9919	0.9974

While the margin of improvement over traditional models may not appear vast at first glance, even minimal increases in model performance can translate into significant real-world benefits. For instance, the enhanced accuracy afforded by the BERT model could potentially lead to thousands of spam messages being correctly identified and filtered out, markedly improving user experience and fostering greater trust in communication platforms.

C. Website Phishing

We evaluated the performance of our final BERT model alongside other machine learning models traditionally used for phishing website detection. The outcomes are summarized in Table III, which presents a performance comparison of machine learning models in phishing website detection.

TABLE III
Performance comparison of phishing website detection models

Model	Precision	Recall	F1-Score	Accuracy	AUC
MNB	0.9366	0.9778	0.9567	0.9367	0.9807
LR	0.8593	0.9455	0.9003	0.8970	0.9674
SVM	0.8628	0.9461	0.9026	0.8995	0.9716
RF	0.8468	0.9400	0.8910	0.8868	0.9655
ADA	0.7874	0.9173	0.8474	0.8375	0.9265
BERT	0.9675	0.9488	0.9606	0.9578	0.9935

The BERT model outperformed traditional machine learning models in accuracy, precision, F1-score and AUC, while only being outperformed on recall by MNB indicating its superior capability in distinguishing between phishing and non-phishing URLs.

Our experimentation reveals that BERT's ability to capture contextual information significantly enhances phishing website detection. Traditional models were less effective due to missing linguistic patterns that BERT can identify, demonstrating the model's robustness against evolving cyber threats.

1) Computational Cost Consideration: Given the similar performance but higher computational and storage costs of using BERT over traditional models, we propose considering low and high computation cost modes for deployment. This dual approach allows leveraging BERT's advanced capabilities where resources permit, while still maintaining effective phishing detection with traditional models in resource-constrained environments. Model storage size is reported in Table X.

BERT was the largest model for SMS and phishing detection problems. RF and SVM were larger for the fake news detection task which is due to how the models scale with the number of input variables.

D. Ethical Considerations

Each model should be used to protect users, in reality, the same models being created to protect users can be used to generate better scams. Crafting adversarial examples without access to the model's internal parameters can be done by querying the model. With enough queries a text that tricks the model can be produced. This text is a bigger threat to users since they are more likely to trust it since the system did not flag it.

Additionally, the more information is given to users on why the text may be fraudulent the easier it is for adversarial examples to be generated. The best possible solution is to constantly upgrade models and to have multiple high-performance models in which a subset is chosen at random to handle predictions. By adding some noise to the output is it significantly more difficult to generate adversarial examples.

When users get used to system protections they get less diligent about protecting themselves which increases their susceptibility to scams that are not filtered by the system. The promotion of healthy media literacy and critical thinking is essential to protect oneself long-term.

False flagging of text as spam, fraudulent, or fake news may negatively impact the creator of the content. To protect against this the user should always be allowed to bypass protects and access the potentially fraudulent content if desired. If the systems are put into practice, periodic analysis should be conducted on fraudulent cases to make sure safe text is not being unfairly filtered.

False flagging may also have political biases and can be used to silence oppositions depending on system implementation. The ability to control what information is trusted can have a significant long-term impact on people's beliefs and a democratic country's leadership.

Lastly, false flags may result in critical information not being acquired on time. If a time-sensitive text/email is filtered as spam it may not be seen until after the critical window resulting in a lost opportunity.

E. Replication Package

The replication package can be found at our Github[4].

V. Conclusion

Our research sheds light on the relative merits of using natural language processing for fraudulent text detection tasks. By providing a clear comparison between a large language model BERT and traditional machine learning methods we display the benefits of using LLMs for fraud detection tasks. Lastly, we highlight the cost-to-performance trade-off of using BERT over traditional machine learning methods.

VI. Future Work

Given more time creating a model to analyze the content of emails would enhance the robustness of the email filtering system to aid in flagging suspicious or fraudulent emails. The current system only detects fraudulent links. If a user can pose as a trustworthy actor they may be able to get the information they want out of the user without any links being present.

Future work would explore more large language models and the size-to-performance trade-off of increased model complexity. Hybrid models that combine the strengths of traditional algorithms and BERT through voting classifiers could be tested to see if they improve performance.

All the models would benefit from protection against adversarial attacks. Augmenting the training data with adversarial examples would accomplish this goal. Input sanitation or normalization to reduce the effectiveness of adversarial perturbations may also help combat this issue. Lastly, adding regularization terms to the model's objective function to discourage sensitivity to small input variations would also help combat this issue.

VII. Limitations

The datasets are older and do not contain the most recent scams. Change over time which will cause model performance to decrease over time. This can be alleviated with periodic model retraining on new data.

The project scope was changed due to time and the number of group members. This led to the creation of multiple projects within the cybersecurity domain rather than one cybersecurity system that chose which model to use based on the domain of the data.

[4]https://github.com/JamesSanii9/QMIND-cybersecurity-projects

VIII. Appendix

A. Performance Metrics

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

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = Sensitivity = \frac{TP}{TP + FN} \tag{3}$$

$$Specificity = \frac{TN}{FP + TN} \tag{4}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{5}$$

B. Model Parameters

TABLE IV
TABLE OF HYPERPARAMETERS USED IN THE FINAL BERT MODEL FOR FAKE NEWS CLASSIFICATION

Hyperparameter	**Value**
Preprocessing URL	bert_en_uncased_preprocess/3
Encoder URL	small_bert/bert_en_uncased_L-2_H-128_A-2/2
Train/Test Split	0.8/0.2
Batch Size	32
Dropout Rate	0.1
Initial Learning Rate	3e-5
Epochs	5
K-Fold Cross Validation	5
Optimizer	AdamW
Loss Function	Binary Crossentropy
Metrics	Accuracy, F1-Score, Recall, Precision, AUC

TABLE V
HYPERPARAMETERS USED IN THE FINAL BERT MODEL FOR SMS SPAM CLASSIFICATION

Hyperparameter	**Value**
Preprocessing URL	/small_bert/bert_en_uncased_preprocess/3
Encoder URL	small_bert/bert_en_uncased_L-4_H-512_A-8
Train/Test Split	0.8/0.2
Validation Size	0.2
Initial Learning Rate	3e-5
Epochs	5
Optimizer	Adam
Dropout Rate	0.1

TABLE VI
TABLE OF HYPERPARAMETERS USED IN THE FINAL BERT MODEL FOR PHISHING SITE URL CLASSIFICATION

Hyperparameter	**Value**
Preprocessing URL	bert_en_uncased_preprocess/3
Encoder URL	small_bert/bert_en_uncased_L-4_H-512_A-8/1
Train/Test Split	Training: 80%, Test: 20%
Batch Size	32
Dropout Rate	0.1
Initial Learning Rate	3e-5
Epochs	3
Optimizer	AdamW
Loss Function	BinaryCrossentropy
Metrics	Accuracy, Precision, Recall, F1-Score, AUC

TABLE VII
TABLE OF OPTIMAL HYPERPARAMETERS USED FOR OTHER MACHINE LEARNING ALGORITHMS FOR FAKE NEWS DETECTION

Model	**Parameters**
MNB	alpha: 1.0, class_prior: None
LR	max_iter = 1000
SVM	C: 1.0, kernel: 'linear'
RF	n_estimators: 100, random_state = 42
ADA	n_estimators: 100, random_state = 42

TABLE VIII
TABLE OF OPTIMAL HYPERPARAMETERS USED FOR OTHER MACHINE LEARNING ALGORITHMS FOR SMS SPAM DETECTION

Model	**Parameters**
MNB	alpha: 0.1, class_prior: None, fit_prior: True
LR	C: 10.0, penalty: 'l2', solver: 'liblinear'
SVM	C: 10, gamma: 'scale', kernel: 'linear'
RF	max_depth: None, min_samples_split: 10, n_estimators: 200
ADA	learning_rate: 1, n_estimators: 200

TABLE IX
TABLE OF OPTIMAL HYPERPARAMETERS USED FOR MACHINE LEARNING ALGORITHMS FOR PHISHING SITE DETECTION

Model	**Parameters**
AdaBoost	n_estimators: 100, random_state = 42
Logistic Regression	max_iter = 1000
MultinomialNB	alpha: 1.0, fit_prior: True, class_prior: None
Random Forest	n_estimators: 100, random_state = 42
SVM (LinearSVC)	max_iter = 10000, random_state = 42

C. Model Size

TABLE X
MODEL SIZE COMPARISON

	Model Size		
Model	**Fake News**	**SMS**	**Phishing**
Multinomial Naive Bayes	313.3 KB	189 KB	57 KB
Logistic Regression	72.97 KB	48 KB	31 KB
Support Vector Machine	298.4 MB	110 KB	321 KB
Random Forest	23.16 MB	13 MB	57.4 MB
AdaBoost	60.75 KB	155 KB	31 KB
BERT	22.40 MB	353.9 MB	123.7 MB

references

REFERENCES

Agarwal, V., Sultana, H. P., Malhotra, S., & Sarkar, A. (2019). Analysis of classifiers for fake news detection [2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019]. *Procedia Computer Science*, *165*, 377–383. https://doi.org/https://doi.org/10.1016/j.procs.2020.01.035

Capuano, N., Fenza, G., Loia, V., & Nota, F. D. (2023). Content-based fake news detection with machine and deep learning: A systematic review. *Neurocomputing*, *530*, 91–103. https://doi.org/https://doi.org/10.1016/j.neucom.2023.02.005

Haynes, K., Shirazi, H., & Ray, I. (2021). Lightweight url-based phishing detection using natural language processing transformers for mobile devices. *Procedia Computer Science*, *191*, 127–134.

Jain, T., Garg, P., Chalil, N., Sinha, A., Verma, V. K., & Gupta, R. (2022). Sms spam classification using machine learning techniques. *2022 12th international conference on cloud computing, data science & engineering (confluence)*, 273–279.

Khanam, Z., Alwasel, B. N., Sirafi, H., & Rashid, M. (2021). Fake news detection using machine learning approaches. *IOP Conference Series: Materials Science and Engineering*, *1099*(1), 012040. https://doi.org/10.1088/1757-899X/1099/1/012040

Kumar, G. R., Gunasekaran, S., Nivetha, R., Shanthini, G., et al. (2019). Url phishing data analysis and detecting phishing attacks using machine learning in nlp. *International Journal of Engineering Applied Sciences and Technology-2019*, *3*(10).

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2021). Phishing email detection using natural language processing techniques: A literature survey. *Procedia Computer Science*, *189*, 19–28.

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2022). A systematic literature review on phishing email detection using natural language processing techniques. *IEEE Access*, *10*, 65703–65727.

Shirani-Mehr, H. (2013). Sms spam detection using machine learning approach. *unpublished) http://cs229. stanford. edu/proj2013/Shir aniMeh r-SMSSpamDetectionUsingMachineLearningApproach. pdf*.

Veeraiah, V., Ravikumar, G., Talukdar, V., Islam, S., Sharma, S., Tulasi, R., & Gupta, A. (2024). Fake news detection using natural language processing and tensorflow in iot system. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(10s), 199–207.