Comparative Analysis of Machine Learning Algorithm for Spam Email Detection

Atuwo David Hyelhirra
Zankli Research Centre
Bingham University
Karu, Nigeria
atuwodavid@outlook.com
0009-0002-1816-8015

Musa Yusuf
Computer Science Department
Bingham University
Karu, Nigeria
yusufmusa@binghamuni.edu.ng
0000-0002-7893-9531

Faki Silas Ageebee
Computer Science Department
Bingham University
Karu, Nigeria
faki.silas@binghamuni.edu.ng
0000-0003-1531-0119

Adamu Usman
Computer Science Department
Bingham University
Karu, Nigeria
adodass@binghamuni.edu.ng
0000-0002-4917-1455

Oluwasegun Ishaya Adelaiye
Computer Science Department
Bingham University
Karu, Nigeria
oluwasegun.adelaiye@binghamuni.edu.ng
0000-0002-4193-5641

Ibrahim Lawal
Computer Science Department
Bingham University
Karu, Nigeria
Lawibrahim3@gmail.com

Comfort Ayenopwa George
ICT Unit
Bingham University
Karu, Nigeria
comfort@binghamuni.edu.ng

Yakubu Aliyu Ibrahim
Department of Computer Science
Bingham University
Karu, Nigeria
yakubu.ibrahim@binghamuni.edu.ng
0000-0002-5529-5243

***Abstract*—Over recent years, emails, SMS, and URLs have become an integral part of communication used by individuals and businesses alike. For all the ease and comfort, it has brought to our lives, it has also brought along with it some unpleasantness in the form of spam that can be used to harm any user by wasting his/her time, computing resources, and stealing valuable information. The ratio of spam emails keeps increasing on a daily basis thus rendering the common ways of dealing with spam such as using alternative email addresses, marking emails as spam in your Gmail/Yahoo folder, etc. inadequate for present times. This project is aimed at comparing the detection performance of Random Forest, Decision Tree, Naïve Bayes, Support Vector Machine, XG Boost, and Adaboost in the detection of spam in email content over five different spam-related datasets. The experimental result shows that the Naïve Bayes algorithm outperformed the other four algorithms considered in this study by achieving an average accuracy of 81.88% which was the highest average recorded among the algorithms considered.**

Keywords—Machine learning, spam detection, Naïve Bayes, data mining, cybersecurity

I. INTRODUCTION

Information sharing has become borderless and seamless through the advent of information technology. Users can make use of various mediums in order to exchange information globally. Email is the easiest, most affordable, and swiftest mode of information sharing available to everyone around the globe. However, they are vulnerable to a variety of attacks because of their simplicity, with spam being the most prevalent and destructive. Undesired emails that are sent to those who are not interested in or require them are referred to as spam [18].

However, Spam messages are embedded with fake content which mostly consists of hyperlinks to additional risks, such as phishing in order to propagate different types of malwares on victims' gadgets and persuade them to click on bogus adverts, spam emails are sent in batches to a variety of people. Spammers can make money from spam mail. The major purpose of sending spam mail aims to acquire consumers' private data wrongly without their consent [6], which will later be exploited against their will to benefit from materialism [3].

Spam emails pose a significant threat to users as they can carry viruses and Trojans, resulting in data or financial fraud and identity theft. Attackers often use spam emails to lure users towards online services by attaching multiple files with packed URLs leading to malicious websites. Additionally, spam emails can cause server response delays by filling up server memory and capacity. To combat spam, organizations use various techniques to identify and analyze incoming emails for spam detection. Some common mechanisms include whitelisting/blacklisting, mail header analysis, and keyword checking. Each organization evaluates the available tools to tackle spam in its environment carefully. Therefore, it is crucial to stay vigilant while opening emails and avoid clicking on suspicious links or attachments to prevent falling prey to spam attacks [3]. Because the content of such emails is constantly changing, even the most advanced spam filtering systems and services have not been able to give a reliable method to discern between genuine and malicious emails [31]. Therefore there is clearly a need to develop a mechanism that can be used to detect the spam with little or no user effort which is what this project aims to accomplish.

II. LITERATURE REVIEW

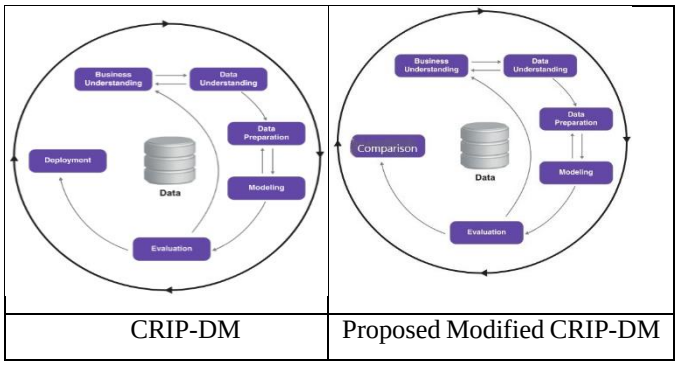
TABLE I. COMPARISON BASED ON THE DATASET, METHODS, FINDINGS, AND RECOMMENDATIONS FROM RELATED LITERATURE

Authors	Dataset	Methods	Findings	Recommendations
[4]	Sem Eval 2016 Sem Eval 2017 STS-Gold, and Sanders	Naïve Bayes, SVM, and DT. Ensemble learners include AdaBoost and Random Forest. Deep learning approaches include RNN, DBN, and CNN-LSTM.	For SemEval2016 and SemEval2017, RNN with GloVe performed best, while RNN with word2vec performed best for STS-Gold and Sanders.	The cross-lingual analysis allows for the transfer of analysis and learning from one language to another.
[27])	Enron Corpus dataset	Support vector machines (SVM), k-nearest neighbors (kNNs), and naive Bayes (NB).	79.6%, 91.5%, and 96.2% respectively.	Bigger datasets should be used to expand on the idea of quickly identifying the most intriguing factors.
[31]	spamFullUrdu Dataset	LSTM, CNN, Naïve Bayes, , and SVM	98%, 96%, 97.5%, and 98.4% respectively	One possible future effort is to compile a dataset of Urdu emails.
[11]	Real-world custom collected dataset on spam tweets as well as a dataset collected from SHP	Naive Bayes and Support Vector Machines (SVM)	83% and 92% respectively	Consider using multilingual tweet datasets and data mining on social media platforms. Add more classifiers to enhance Twitter spam detection for users.
[15]	email spam from the UCI machine learning repository	Recurrent Neural Networks (RNN) and Artificial Neural Networks, Naive Bayes, Random Forest (RF), KNN, and Gradient Boosting (GB) (ANN)	ML algorithms perform better than deep learning algorithms for tabular datasets to identify spam e-mails	Image spam, blank spam, and backscatter spam detection should be considered and text data direct from text content instead of tabular data using natural language processing.
[20]	Ready-made dataset	Naive Bayes, K-Nearest Neighbors, SVM, Logistic regression, Decision tree, and Random Forest	The most accurate methods are NB and logistic regression, both of which have a 99% accuracy rate.	Combining algorithms with filtering techniques can provide a spam detection classifier that is more sophisticated.
[14]	SMS spam collection dataset	K-nearest neighbor, decision tree, and logarithmic regression.	99%, 95%, and 98% respectively	Other forms of Spam content should be considered.
[23]	REFIT Smart Home dataset	Generalized Linear Model with Stepwise Feature Selection, Boosted Linear Model, XG Boost, and BGLM	79.81%, 83.22%, 84.35%, 88.9% and 91.8%	IoT device climatic and environmental characteristics should be taken into account.
[12]	Spam archive, phishing corpus, and Princeton spam image benchmark datasets	Adaboost, SVM, Random Forest, and Naïve Bayes.	Random Forest was the most accurate	Adversarial learning approaches should be considered.
[21]	Spam.csv dataset	Naïve Bayes, SVM, Decision Trees and KNN	98%, 95%, 95% and 92%	Spam can be filtered using reputable domain names and email categorization.
[13]	UCL Machine Learning repository	Spam can be filtered using reputable domain names and email categorization.	67.13%, 85.54%, and 92.97% respectively.	More algorithms should be run on multiple datasets in order to come up with models of higher accuracy.
[29]	Actual SMS spam from the UCI Machine Learning Database	Naïve Bytes, Support Vector Machines (SVM), K-nearest neighbor, and Random Forest	97.5%, 98.86%, 97.47%, and 98.57% respectively	Other forms of spam messages can be considered.
[2]	Spam database dataset and the WEKA data miner from the UCL machine learning repository	Radial Basis Function (RBF) Network, Bayesian Logistic Regression, Hidden Naive Bayes, Voted Perceptron, Naive Bayes, Multilayer Perceptron, Random Tree, J48, Logit Boost, Rotation Forest, Logistic Model Tree, REP Tree, and Lazy Bayesian Rule	The Rotation Forest algorithm yielded the highest accuracy of 94.2% when 10-fold cross-validation was employed on it, and the lowest accuracy of 89.1% was obtained when the REPTree algorithm was broken into six equal halves.	The outcomes will be compared with additional spam datasets from other sources and different machine-learning technologies. The analysis of further classification techniques using email spam datasets is also necessary.
[7]	"ECML" and the UCI dataset	Naive Bayes, SVM, K-Nearest Neighbour, Bayes Additive Regression, and KNN Tree	Predicted that Naïve Bayes and SVM algorithms are the most effective in the classification of spam.	Next-generation spam filtering processes should have the ability to consider a large number of multimedia data and filter spam emails more prominently.
[28]	Ling spam dataset	Naïve Bayes and J48 (decision tree)	83.5% and 91.5% respectively.	In order to enhance the system's performance and results, the concept of boosting approach could be considered for future work

III. METHODOLOGY

A. Conceptual Design of Framework

The conceptual design of this project is derived from the CRISP-DM model in which the whole process of software development is divided into separate phases. The major difference between the methodology used for this research and the CRISP-DM model has to do with the modeling phase which has been replaced with a comparison phase to suit the purpose of this study. A secondary source of information was used in the development of the proposed model where requirements were captured using the specified dataset. Thereafter the design of the system was modeled using machine algorithms, implemented, and then compared on parameters such as accuracy and F1-score. The phases of the modified methodology are explained below:



CRIP-DM | Proposed Modified CRIP-DM

Fig. 1. A Diagram Showing the Modification of the CRISP-DM Model

Following the adjusted methodology, the machine learning process that will be adopted has six steps:

1) Problem Understanding

This step entails identifying the existing problem with a view towards understanding how it can be solved, and deciding whether machine learning will be a useful tool for solving it.

2) Data Understanding

In this phase, available datasets will be analyzed and decisions will be taken as to determine whether they are sufficient enough to carry out analysis. In this section, we try to determine the data sources that can be used to resolve the issue. Then we examine and analyze the data to determine whether it is enough to resolve our problem.

The data used for analysis may be insufficient for a variety of reasons, including the dataset becoming too minimal for us to gain knowledge of any useful patterns, the dataset becoming too noisy, and the dataset containing too many invalid fields or an excessive number of empty fields, rendering it ineffective for the training of a machine learning model or the process for gathering data being broken, collecting only a small fraction of the data we want.

If the data we currently have is found to be insufficient, better data will have to be sourced, which in this case would be to collect or source additional spam mail-related datasets. It is also feasible that the discoveries made in this step will have an impact on the goal established earlier in the requirement analysis step, so we may need to return to that stage and adjust the goal based on our findings. When efficient information sources are identified, we move on to the data pre-processing stage.

3) Data Preparation

This phase involves transforming the data into a format that can be used as input for a machine-learning model. In this step, we clean the data by carrying out tasks such as filling missing fields with the mean of the values found there and also by carrying out pre-processing activities which include converting the strings or characters of the spam mail into numeric values which the machine can understand and generally involves modifying it so that it can be a model's contribution in a machine learning system. In general, we convert the dataset into a collection of attributes that we later feed into a model.

4) Modeling

After the data has been prepared, we proceed to the modeling stage, where we train a model. In this step, machine learning models/algorithms are applied to the training sets of the various datasets to measure their performance. For the spam datasets, we will look at how well the models predicted spam messages and take note of their accuracy, precision, recall, and F1 score. After we develop the model, we go to the evaluation step.

5) Evaluation

Following the training, there is an evaluation step in which we evaluate the model to see if it helps solve the original challenges and evaluate its success in doing so. In this step, we determine whether the model meets expectations. When we defined the goal in the requirement analysis step, we also described the method for determining whether the goal was met. Usually, we do this by examining a critical business metric and ensuring that the model needs to move this same metric in the desired direction. In this research, the metric could be the accuracy of the models in detecting spam mail across the various datasets used

To effectively carry out this comparison analysis for this research paper brings about the modifications found in this framework which takes us to the next step: Comparison.

6) Comparison

Finally, in the comparison step, we bring out the results of all the algorithms run on the various datasets to classify them and find out which algorithm performed best in terms of accuracy in identifying spam e-mails consistently across the various datasets that they were run on

After the comparative analysis, we use everything we learned in all the steps and go back to the first step to reflect on what we achieved (or didn't achieve). We may realize that our initial goal was not achieved. So we go back to the business understanding step to redefine our goal or increase our scope. Then, when we evaluate the model again, we use a different business metric to measure its success.

IV. IMPLEMENTATION PROCEDURES

A. Tools to be Used in Implementation

1) *Anaconda*: Anaconda is a popular data science platform that provides various tools for data analysis and machine learning. It includes several pre-installed libraries and packages for processing and analysing data, which can be used for detecting spam in SMS, email, and URLs. The platform also provides an integrated development environment (IDE) for writing code and developing applications.
2) *Python Programming Language*: Python is a high-level programming language that is widely used in data science and machine learning. It provides various libraries and packages that can be used for detecting spam in SMS, email, and URLs. Python has a large community of developers and users, which makes it easier to find and use relevant tools and resources.
3) *Jupyter Notebook*: Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It is commonly used for data science and machine learning, including detecting spam in SMS, email, and URLs. Jupyter Notebook supports several programming languages, including Python, R, and Julia.

4) *Spam-Related Datasets*: Spam-related datasets are collections of data that are used for training and testing machine learning algorithms for detecting spam in SMS, email, and URLs. These datasets contain labelled examples of spam and non-spam messages or URLs, which can be used to train and evaluate machine learning models.
5) *Machine Learning Algorithms*: Machine learning algorithms are used for detecting spam in SMS, email, and URLs. These algorithms learn from labelled examples of spam and non-spam messages or URLs to identify patterns and features that can distinguish between them. Some popular machine learning algorithms for spam detection include Naive Bayes, Support Vector Machines (SVM), and Random Forests.
6) *Colab*: Colab is a free web-based service provided by Google that allows users to write and run Python code in a Jupyter Notebook environment. It provides access to powerful hardware resources, such as GPUs and TPUs, which can be used for training machine learning models for detecting spam in SMS, email, and URLs. Colab also provides access to several pre-installed libraries and packages, including those for data analysis and machine learning.

B. Sources of Data Collection

Secondary sources of data were used in this study where datasets containing various forms of text-based spam messages in the form of email, sms, and URLs with their respective classifications as spam or ham messages were used and they include:

1) SMS Spam Collection

The SMS Spam Collection is a compilation of SMS-tagged messages gathered for research on SMS Spam. It comprised of 5,574 SMS messages in English, classified as either legitimate or spam. The messages are stored in separate files with one message per line, each line consisting of two columns: v1 contains the label (ham or spam) and v2 contains the raw text.

This collection was obtained from various sources on the internet, including a Grumbletext website where 425 SMS spam messages were manually extracted. The identification of spam messages was a challenging and time-consuming task as it required thorough scanning of hundreds of web pages. A subset of 3,375 randomly selected ham messages from the NUS SMS Corpus, a dataset of 10,000 legitimate messages collected for research at the Department of Computer Science at the National University of Singapore, was also included. These messages were collected from volunteers who were informed that their contributions would be made public.

The SMS Spam Corpus v.0.1 Big, which consists of 1,002 ham messages and 322 spam messages, was also incorporated into this collection. The dataset is available at https://www.kaggle.com/datasets/mfaisalqureshi/spam-email/.

Fig. 2. *A Diagrammatic Representation of the Ling Spam Dataset Csv File*

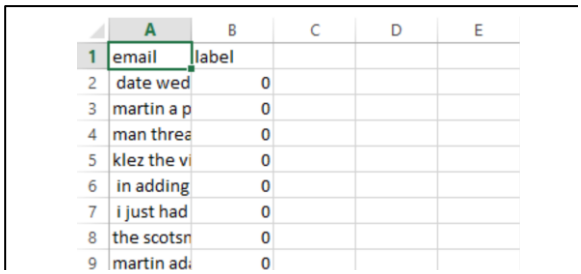
2) Spam Full Urdu Dataset

The Spam Full Urdu Dataset is a collection of 5000 emails written in Urdu script, which has been translated into Urdu from Roman Urdu. The dataset is labeled with two columns: "type" which contains the values of spam or ham and is intended for email classification, and "e-mail Text" which contains various types of email content. It's worth noting that Urdu script is distinct from Roman Urdu script and uses the Arabic alphabet. The dataset can be found in the GitHub repository at https://github.com/zeeshanbinsiddique/Urduemaildataset.

Fig. 3. *A Diagrammatic Representation of the Ling Spam Dataset Csv File*

3) Spam or Not Spam Dataset

The Spam or Not Spam Dataset includes two compressed files, '0030228easyham.tar.bz2' and '20030228_spam.tar.bz2', which were taken from the public corpus of Apache Spam Assassin. The dataset contains a total of 2500 legitimate (ham) emails and 500 spam emails. It's worth noting that all numbers and URLs in the emails were converted to "NUMBER" and "URL" strings, respectively. This simplified dataset can be accessed via the Kaggle dataset link at https://www.kaggle.com/datasets/ozlerhakan/spam-or-not-spam-dataset

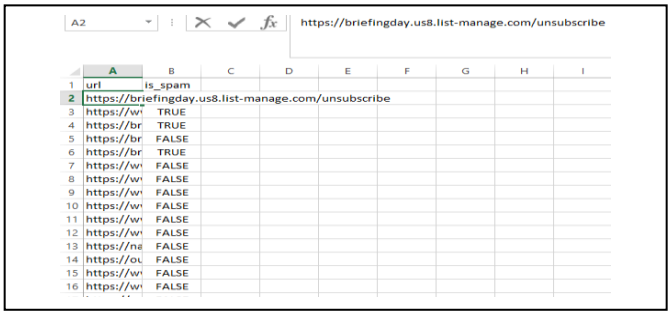


	A	B	C	D	E
1	email	label			
2	date wed	0			
3	martin a p	0			
4	man threa	0			
5	klez the vi	0			
6	in adding	0			
7	i just had	0			
8	the scotsn	0			
9	martin ad	0			

Fig 4. A Diagrammatic Representation of the Ling Spam Dataset Csv File

4) Spam URLs Classification Dataset

This dataset contains about 87.5K URLs of which one-third are flagged as spam URLs and restrict are not spam. It can be used to create a binary classification model. [22]



A2 | https://briefingday.us8.list-manage.com/unsubscribe

	A	B
1	url	is_spam
2	https://briefingday.us8.list-manage.com/unsubscribe	
3	https://w	TRUE
4	https://br	TRUE
5	https://br	FALSE
6	https://br	TRUE
7	https://w	FALSE
8	https://w	FALSE
9	https://w	FALSE
10	https://w	FALSE
11	https://w	FALSE
12	https://w	FALSE
13	https://na	FALSE
14	https://ou	FALSE
15	https://w	FALSE
16	https://w	FALSE

Fig. 4. *A Diagrammatic Representation of the Spam Url C*

5) Ling Spam Dataset

The Ling-Spam dataset is a compilation of 2,893 messages, which include both spam and non-spam messages that were gathered from the Linguist List. The messages are primarily focused on linguistics topics such as job postings, research opportunities, and software discussions.

The dataset contains 2412 legitimate messages and 481 spam messages, with header information removed for ease of analysis. It is a valuable resource for researchers who are interested in exploring and developing machine-learning algorithms for spam detection and classification.

The Ling-Spam dataset is publicly available on Kaggle, making it easily accessible to researchers and developers. The dataset's inclusion of both spam and non-spam messages and its focus on linguistics topics make it a unique and valuable resource for the research community. Researchers can use the dataset to evaluate and compare different spam detection algorithms and techniques, and to develop new methods for more accurate spam detection and filtering.

job posting - apple-iss research center

	A	B	C	D
1	subject	message	label	
2	job postin	content -	0	
3		lang	0	
4	query : let	i am	0	
5	risk	a	0	
6	request b	earlier	0	
7	call for ab	content -	0	
8	m . a . in s	m . a . in	0	
9	call for pa	call for	0	
10	foreign lai	content -	0	
11	fulbright a	fulbright	0	
12	gala ' 95 : (	groninge	0	

Fig. 5. *A Diagrammatic Representation of the Ling Spam Dataset Csv File*

C. Machine Learning Algorithms Considered

1) Random Forest (RF)

The Random Forest algorithm is a type of Decision Tree-based classifier that is commonly used for spam detection. This algorithm employs an ensemble learning technique, in which multiple decision trees are built and combined to improve the accuracy of the final classification [32]Specifically, the Random Forest algorithm selects the best classification tree by voting, which involves aggregating the predictions of multiple decision trees and selecting the most common classification.

Mathematically, the Random Forest algorithm can be represented as follows:

Given a training dataset D = $\{(x_1,y_1), (x_2,y_2),..., (x_n,y_n)\}$, where x_i represents the i-th sample and y_i represents the corresponding class label, the algorithm performs the following steps:

1. Randomly selects a subset of the features from the dataset.
2. Builds a decision tree for each subset of features.
3. Aggregates the predictions of all the decision trees through voting.
4. Selects the most common classification as the final output.

The Random Forest algorithm has been shown to provide higher accuracy in spam detection tasks through cross-validation. This is because the algorithm is less prone to overfitting, which occurs when the model is too complex and fits the training data too closely, leading to poor performance on new, unseen data.[2]).

2) Decision Tree (DT)

The decision tree algorithm is a classification method that uses a hierarchical structure of rules to classify data. This model can be represented as a tree, which makes it easily interpretable. Additionally, the decision tree algorithm automatically selects relevant features and eliminates irrelevant and redundant ones. The learning process of a decision tree model involves three steps: feature selection, tree generation, and tree pruning. During training, the algorithm selects the most appropriate feature to split the data into two child nodes from the root node. The splitting process continues recursively until all the data is classified. The decision tree is a simple and basic classifier. However, advanced algorithms like Random Forest and XGBoost use multiple decision trees to achieve better performance [22]

Let X be the input features and Y be the output labels. Given a set of training data $\{(x_1,y_1), (x_2,y_2),..., (x_n,y_n)\}$, the goal of the decision tree algorithm is to learn a function f such that f(X) = Y. The algorithm learns this function by recursively partitioning the data into subsets based on the feature that provides the most information gain. The decision tree can be represented as T(X) where T is a tree that maps the input feature X to the output label Y.

3) Support Vector Machine (SVM)

Support Vector Machines (SVMs) are powerful classification algorithms that aim to find the optimal hyperplane that separates the data points into different classes [33] The hyperplane with the maximum margin between the classes is the one that best separates the data. The mathematical notation for the hyperplane is given by:

$$w \cdot x + b = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

where w is the weight vector and b is the bias term. The decision boundary is determined by the sign of the expression $w \cdot x + b$. If the expression is positive, the sample is classified as belonging to one class, and if it is negative, it is classified as belonging to the other.

SVMs work well even when the training set is small because they only depend on a few support vectors. These support vectors are the data points closest to the decision boundary and are used to determine the optimal hyperplane. The mathematical notation for this optimization problem is:

$$\min \tfrac{1}{2}||w||^2 + C\sum\xi \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

subject to:

$$y(w \cdot x+b) \geq 1-\xi \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3)$$

$$\xi \geq 0$$

where C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error, ξ are slack variables that allow for the misclassification of data points, and y is the class label (either +1 or -1).

SVMs are sensitive to noise near the hyperplane, which can cause overfitting. To handle nonlinear data, kernel functions are used to map the data to a higher dimensional space where it can be linearly separable. The mathematical notation for the kernel function is:

$$K(x, x') = \varphi(x) \cdot \varphi(x') \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4)$$

where $\varphi(x)$ is a function that maps the original input space into a higher dimensional space. The kernel function allows for the computation of the dot product of the mapped data points without explicitly computing the mapping function, which can be computationally expensive.

4) Naïve Bayes (NB)

The Naïve Bayes algorithm is commonly used due to its effectiveness in dealing with high-dimensional feature spaces, such as the bag-of-words representation of text data. The algorithm calculates the probability of a given sample belonging to a particular class, given its feature values.

Mathematically, the Naïve Bayes algorithm estimates the conditional probability of a class, given the feature values of a sample, using Bayes' theorem:

$$P(\text{class} \mid \text{features}) = \frac{P(\text{features} \mid \text{class}) * P(\text{class})}{P(\text{features})} \ldots (5)$$

where P(class | features) is the posterior probability of the class given the features, P(features | class) is the likelihood of the features given the class, P(class) is the prior probability of the class, and P(features) is the probability of the features. The Naïve Bayes classifier assumes that the features are conditionally independent given the class, allowing it to simplify the calculation of the likelihood term:

$$P(\text{features} \mid \text{class}) = P(\text{feature1} \mid \text{class}) * P(\text{feature2} \mid \text{class}) * \ldots * P(\text{features} \mid \text{class}) \quad \ldots\ldots\ldots\ldots \quad (6)$$

where feature1, feature2, ..., and features are the feature values of the sample. The prior probability P(class) can be estimated from the training data, while the likelihood P(features | class) can be estimated using the frequency of each feature value in the training samples of the corresponding class.

In the case of spam detection, the Naïve Bayes algorithm can be trained on a set of labeled emails, where the classes a"e "s"am" a"d "non-s"am" and the features are typically the frequencies of words or word n-grams in the email. The trained model can then be used to classify new emails as spam or non-spam based on their feature values [22].

*5) Extreme Gradient Boosting (XG Boost (**XB**))*

Extreme Gradient Boosting (XGBoost) is an effective supervised learning algorithm that employs ensemble trees to achieve high classification accuracy. It optimizes a cost objective function, which is made up of a loss function (d) and a regularization term (β). The objective function can be represented mathematically as:

$$L(\Theta) = \sum(i=1 \text{ to } n)\, d(yi, F(xi)) + \beta\Omega(F) \quad \ldots\ldots \quad (7)$$

where Θ represents the model parameters, n is the number of training samples, xi is the ith training sample, yi is the corresponding class label, F(xi) is the predicted output of the model for xi, and Ω(F) is the regularization term. The loss function d measures the difference between the predicted output and the actual output for a given input. The regularization term Ω(F) is used to penalize complex models that are prone to overfitting. XGBoost combines the outputs of multiple weak learners (decision trees) to create a strong ensemble model that can accurately classify spam messages. During training, the model parameters are iteratively updated to minimize the cost objective function using gradient descent. XGBoost is known for its speed and scalability, making it a popular choice for large-scale spam detection tasks [9]

6) AdaBoost (AB)

Adaboost is a popular classification algorithm used for detecting spam emails. It is well-known for its adaptability for cascade classification. The Adaboost algorithm employs a supervised learning approach where a large dataset of labeled positive (spam) and negative (ham) emails are used for training. In this process, the training emails are pre-processed to extract relevant features that can be used for classification.

The training process of Adaboost Involves iteratively adjusting the weights of the training samples based on the

performance of the weak classifiers. This process is aimed at improving the classification accuracy of the algorithm. In mathematical notation, the Adaboost algorithm can be represented as follows:

Given a set of training samples $\{(x_1,y_1), (x_2,y_2),..., (x_n,y_n)\}$, where xi represents the features of the i-th email and yi represents the corresponding label (spam/ham), the Adaboost algorithm iteratively builds a strong classifier h(x) as follows:

1. Initialize the weights of the training samples as wi=1/n, where n is the number of training samples.
2. For t=1 to T:
   a. Train a weak classifier ht(x) on the training samples using weights wi.
   b. Calculate the error of ht(x) as $\varepsilon t=\sum(i=1 \text{ to } n)$ $wiI(yi\neq ht(xi))$, where I() is the indicator function.
   c. Calculate the weight of ht(x) as $\alpha t=0.5\ln((1-\varepsilon t)/\varepsilon t)$.
   d. Update the weights of the training samples as $wi=wiexp(-\alpha tyiht(xi))$, where yi represents the true label of the ith sample.
3. Output the final classification as $h(x)=sign(\sum(t=1 \text{ to } T) \alpha t*ht(x))$.

Thus, the AdaBoost algorithm is a powerful tool for detecting spam emails and can provide high classification accuracy with relatively small training sets [17]. Metrics to be Used In Evaluating the Algorithms

7) Accuracy

This metric measures the proportion of correct predictions made by a classification model. It is calculated as:

$$\text{Accuracy} = \frac{\text{True Negatives} + \text{True Positives}}{\text{Total Number of Examples}} \quad \ldots\ldots\ldots (8)$$

8) Precision

This statistic computes the percentage of valid affirmative identifications. It is calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{False Positives} + \text{True Positives}} \quad \ldots\ldots\ldots\ldots\ldots (9)$$

9) Recall

This metric evaluates the proportion of actual positives that were identified correctly. It is calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{False Positives} + \text{True Positives}} \quad \ldots\ldots\ldots\ldots\ldots (10)$$

10) F1-score

This measure, which is understood as the weighted harmonic mean of accuracy and recall, seeks a balance across both recall and precision . It varies from accuracy in that accuracy only accounts for genuine positives and negatives, while false positives and negatives are ignored. The F1-score is determined as follows:

$$\text{F1-Score} = \frac{(\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} \quad \ldots\ldots\ldots\ldots\ldots (11)$$

True Positive – An example in which the model correctly predicted the positive class. A scenario in which the model guessed/predicted correctly in the positive class.

False Positive – An example in which the model mistakenly predicted the positive class. A scenario in which the model guessed/predicted wrongly in the positive class.

True Negative – An example in which the model correctly predicted the negative class. A scenario in which the model guessed/predicted correctly in the negative class.

False Negative – An example in which the model wrongly predicted the negative class. A scenario in which the model guesses/is predicted wrongly in the negative class.

D. Limitations of the Study

This research was limited to the use of only the ML algorithms used on the datasets chosen as well as only text-based spam messages. Furthermore, only ML algorithms were compared on the dataset with no artificial neural network and deep learning algorithms specifically used on the datasets.

V. RESULT AND DISCUSSION

The evaluation of the detection models was done base on detection accuracy and F1-Score presented in Table II and Table III respectively.

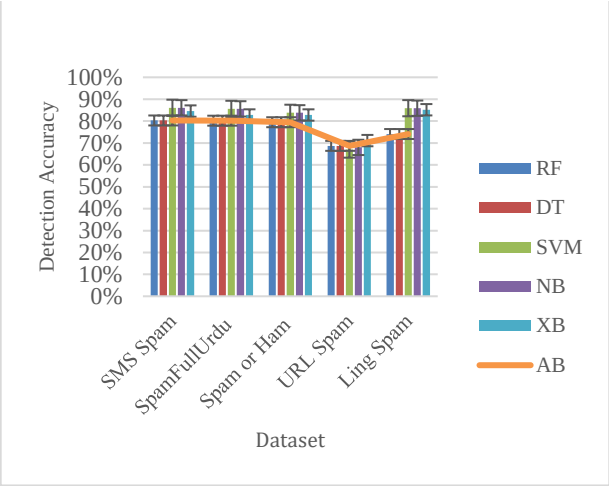
TABLE II. DETECTION ACCURACIES OF THE SELECTED ALGORITHMS ON THE SELECTED DATASETS

	ML Algorithms					
Datasets	**RF**	**DT**	**SVM**	**NB**	**XB**	**AB**
Sms spam	0.80.%	80.3%	86.1%	86.1%	84.6%	80.3%
SpamFullUrdu	80.2%	80.2%	85.6%	85.6%	82.8%	80.2%
Spam or Ham	79.5%	79.5%	83.8%	83.8%	82.8%	79.5%
URL Spam	68.7%	68.7%	67.0%	68.0%	71.1%	68.7%
Ling Spam	74.1%	74.1%	85.9%	85.9%	85.2%	74.1%

A. Detection Accuracy

1) Model Performance Comparison

The SVM model achieves the highest accuracy on the S spam dataset (86.1%) and the Ling Spam dataset (85.9%). It appears to be particularly effective at handling linguistically challenging spam. The (NB) model performs well on datasets like SMS spam, SpamFullUrdu, and URL Spam, with accuracy ranging from 68.0% to 86.1%. It seems to excel at detecting certain types of spam, especially those with URL-like patterns. The RF, DT, and AB models show consistent accuracy across datasets, ranging from 68.7% to 80.3%. These models seem to offer stable and reliable performance across different spam types. The XB model achieves relatively high accuracy on the Ling Spam dataset (85.2%) and the URL Spam dataset (71.1%). It might be more suitable for detecting linguistically challenging spam and spam with URL patterns.



2) Dataset Influence

Different datasets present varying degrees of difficulty in spam detection, which is evident from the variation in accuracy scores across models and datasets. For instance, the Ling Spam dataset appears to be more challenging, as reflected by the relatively higher accuracy of the SVM, NB, and XB models on this dataset compared to others. The URL Spam dataset seems to be relatively easier for the XB model, as it obtains higher accuracy compared to other models.

3) Model Robustness and Generalization

The RF, DT, and AB models consistently perform well across multiple datasets, indicating a higher level of generalization and robustness. These models might be preferred when dealing with diverse spam types. The SVM and XB models, while achieving high accuracy on specific datasets, display more variation in performance across datasets. This suggests they may be more sensitive to variations in data characteristics.

4) Need for Model Selection

The study highlights the importance of selecting the appropriate machine learning model based on the specific dataset and task requirements. Depending on the dataset characteristics, different models might yield varying levels of accuracy, and choosing the right model is crucial for effective spam detection.

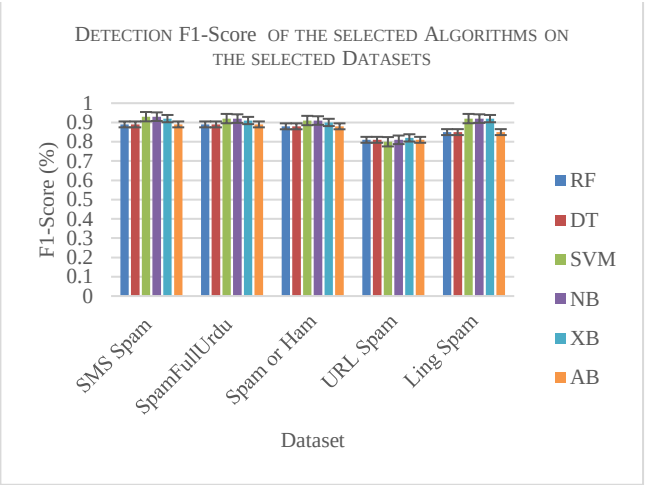
B. F1- Score

TABLE III. DETECTION F1-SCORE OF THE SELECTED ALGORITHMS ON THE SELECTED DATASETS

	ML Algorithms					
Datasets	**RF**	**DT**	**SVM**	**NB**	**XB**	**AB**
SMS Spam	0.89	0.89	0.93	0.93	0.92	0.89
SpamFullUrdu	0.89	0.89	0.92	0.92	0.91	0.89
Spam or Ham	0.88	0.88	0.91	0.91	0.90	0.88
URL Spam	0.81	0.81	0.80	0.81	0.82	0.81
Ling Spam	0.85	0.85	0.92	0.92	0.92	0.85

1) Model Performance Comparison

Among the six spam detection machine learning models (RF, DT, SVM, NB, XB, and AB), SVM consistently demonstrates the highest F1-Scores across all datasets, ranging from 0.80 to 0.93. It outperforms other models, indicating its effectiveness in spam detection. DT and RF models also exhibit competitive and stable F1-Scores, ranging from 0.85 to 0.93, showcasing their reliability in spam detection tasks. NB and XB models demonstrate competitive F1-Scores ranging from 0.80 to 0.92, with NB performing well on SMS Spam and SpamFullUrdu datasets, while XB excels on Ling Spam and URL Spam datasets. AB model achieves F1-Scores ranging from 0.85 to 0.89, showing consistent but slightly lower performance compared to other models.



2) Dataset Influence

The variation in F1-Scores across datasets indicates that different datasets pose varying levels of complexity for spam detection. Ling Spam and SMS Spam datasets appear to be more challenging, as evidenced by the lower F1-Scores achieved by some models on these datasets. Understanding the dataset influence is crucial for selecting the most appropriate model and devising dataset-specific strategies for improved spam detection accuracy.

3) Model Robustness and Generalization

SVM, DT, and RF models demonstrate stable and robust performance across datasets, suggesting their strong generalization capabilities. NB and XB models also exhibit competitive F1-Scores, indicating their potential for generalization, although their performance may vary slightly across datasets [35]. The AB model shows consistent but slightly lower performance compared to other models, suggesting a moderate level of robustness. Evaluating model robustness and generalization is important to ensure reliable spam detection in real-world scenarios and to avoid overfitting to specific datasets.

4) Need for Model Selection

The choice of the best model depends on the specific dataset characteristics and desired performance metrics (F1-Score in this case). SVM emerges as the top performer with the highest F1-Scores, making it a suitable choice when optimizing for F1-Score and high accuracy. DT and RF models offer balanced and robust performance, making them attractive choices for spam detection tasks that require both precision and recall. NB and XB models are also competitive options, especially when F1-Score is crucial but may slightly vary depending on the dataset characteristics. The AB model performs well but achieves

slightly lower F1-Scores compared to other models, necessitating consideration of the trade-offs involved in using an ensemble model.

VI. Summary

After successfully evaluating the earlier identified machine learning algorithms on the chosen datasets and comparing them based on the metrics of accuracy, precision, recall and F1-score it was observed that Naïve Bayes is the most accurate machine learning algorithm in detecting spam messages as it achieved an average accuracy of 81.88% which was the highest amongst the algorithms considered after which it was closely followed by Support Vector Machines which had an average accuracy of 81.68%. XG Boost achieved an average accuracy of 81.29% after which Random Forest, Decision Tree, and AdaBoost had the same average accuracy of 76.56%. From the results, it was also observed that the algorithms were most effective in detecting SMS-based spam and least effective in detecting URL-based spam with only XG Boost attaining accuracy above 70%.

It was also observed that Naïve Bayes and support vector machines generally had the same results with the exceptions being their performance on the URL Spam dataset in which Naïve Bayes performed slightly better than SVM across metrics like accuracy and recall thus leading the former to have the highest average accuracy in the study. Random Forest, Decision Tree, and Adaboost on the other hand had the same results consistently and ended up having the same average across all the metrics that were considered in this study.

VII. Conclusion

This research was undertaken in order to find out which machine learning algorithm is the most accurate in detecting spam-based messages and I believe this goal has been achieved by: Developing a suitable methodology that enables comparative analysis to be done. Making use of up to 5 datasets containing various forms of spam messages in the form of email, sms and URL spam. Testing up to six (6) different machine learning algorithms on each of the datasets in order to gauge their performance. Evaluating the algorithms based on various metrics and bringing out the results in order to observe various similarities/differences. These processes help add credibility to the procedure followed that identified Naïve Bayes as the most efficient machine learning classification algorithm for the identification of spam-based messages.

VIII. Recommendations

This research work has to a large extent expanded on the various works done by other researchers in terms of machine learning detection of spam-based messages and has given key insight into how the most common algorithms being used perform on other forms of text spam. Future works in this regard can aim to determine why Naïve Bayes is the most accurate in the detection of spam-based messages as well as testing out additional algorithms such as LSTM, CNN, KNN, etc. to further enhance the parameters for the comparative analysis.

References

[1] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[2] Abdulkareem, N. M., & Abdulazeez, A. M. (2021). Machine Learning Classification Based on Random Forest Algorithm: A Review. International Journal of Science and Business, 1-15.

[3] Ahmed, Amin, HamzaAldabbas, DeepikaKoundal, BaderAlouffi, & Shah, a. T. (2022). Machine Learning Techniques for Spam Detection in Email and IoT Platforms: Analysis and Research Challenges. 1-19.

[4] Ayyub, K., Iqbal, S., Nisar, M. W., Munir, E. U., Alarfaj, F. K., & Almusallam, N. (2022). A Feature-Based Approach for Sentiment Quantification Using Machine Learning. Electronics 2022, 11, 846, 1 - 17.

[5] BAKTIR, N., & ATAY, Y. (2022). Comparative Analysis of Machine Learning Approaches in the Spam-Mail Classification Problem. Journal of Information Technologies, 1-5.

[6] Bauer E. 2022. Outrageous email spam statistics that still ring true in 2022. Available at https://dataprot.net/statistics/spam-statistics/ (accessed 17 August 2022).

[7] Bhuiyan, H., Ashiquzzaman, A., Juthi, T. I., Biswas, S., & Ara, J. (2018). A Survey of Existing E-Mail Spam Filtering Methods Considering Machine Learning Techniques. 1-11.

[8] Bi, Q., Goodman, K. E., Kaminsky, J., & Lessler, J. (2019). What is Machine Learning? A Primer for the Epidemiologist. American Journal of Epidemiology, 1-18.

[9] Cherif, I. L., & Kortebi, A. (2019). On using eXtreme Gradient Boosting (XGBoost) Machine Learning algorithm for Home Network Traffic Classification. 1-6.

[10] Crawford, M., Khoshgoftaar, T. M., Prusa, J. D., & Najada, A. N. (2019). Survey of review spam detection using machine learning techniques. 1-24.

[11] Falak, A., Ghous, D. H., & Malik, D. M. (2021). Twitter Spam Detection Using Machine Learning. 1-17.

[12] Gangavarapu, T., Jaidhar, C. D., & Chanduka, B. (2020). Applicability of machine learning in spam and phishing email filtering: review and approaches. Artificial Intelligence Review, 1-63.

[13] Gaurav, D., Tiwari, S. M., Goyal, A., Gandhi, N., & Abraham, A. (2019). Machine intelligence-based algorithms for spam filtering on document labeling. Soft Computing https://doi.org/10.1007/s00500-019-04473-7(0123456789().,-volV)(0123456789().,- volV), 1-14.

[14] GuangJun, L., Nazir, S., & Haq, H. U. (2020). Spam Detection Approach for Secure Mobile Message Communication Using Machine Learning Algorithms. 1-6.

[15] Hossain, F., Uddin, M. N., & Halder, R. K. (2021). Analysis of Optimized Machine Learning and Deep Learning Techniques for Spam Detection. 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS) | 978-1-6654-4067-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEMTRONICS52119.2021.9422508, 1-7.

[16] Hubera, S., Wiemera, H., Schneidera, D., & Ihlenfeldta, S. (2019). DMME: Data mining methodology for engineering applications – methodology to analyze the functional and physical architecture. Science Direct, 1-6.

[17] Jabri, S., Saidallah, M., Alaoui, A. E., & Fergougui, A. E. (2018). Moving Vehicle Detection Using Haar-like, LBP, and a Machine Learning Adaboost Algorithm. IEEE International Conference on Image Processing, Applications, and Systems (IPAS), 1-4.

[18] Kaddoura, S., Ganesh, C., Popescu, Elena, D., Duraisamy, & Hemanth, J. (2022). A systematic literature review on spam content detection and classification. 1-28.

[19] Karim, A., Azam, Sami, Shanmugam, Bharanidharan, Kannoorpatti, K., . . . Mamoun. (2019). A Comprehensive Survey for Intelligent Spam Email Detection. IEEE Access, 1-35.

[20] Kontsewayaa, Y., Antonova, E., & Artamonovb, A. (2021). Evaluating the Effectiveness of Machine Learning Methods for Spam Detection. Procedia Computer Science 190 (2021) 479–486, 1-8.

[21] Kumar, N., & Sonowal: Nishant, S. (2020). Email Spam Detection Using Machine Learning Algorithms. IEEE Xplore Part Number: CFP20N67-ART; ISBN: 978-1-7281-5374-2, 1-6.

[22] Liu, H., & Lang, B. (2019). Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. Applied Sciences, 1-28.

[23] Makkar, A., Garg, S., Kumar, N., Hossain, M. S., Ghoneim, A., & Alrashoud, M. (2020). An Efficient Spam Detection Technique for IoT Devices using Machine Learning. IEEE Transactions on Industrial Informatics, 1-9.

[24] Mart´ınez-Plumed, F., Contreras-Ochando, L., Ferri, C., andez-Orallo, J. `., Kull, M., Lachiche, N., . . . Flach, P. (2019). CRISP-DM Twenty Years Later:From Data Mining Processes to Data Science Trajectories. IEEE Transactions on Knowledge and Data Engineering, 1-14.

[25] Murugan, N. S., & Devi, G. U. (2018). Detecting spams in social networks using ML algorithms – a review. 1 - 16.

[26] Nigam, K., Mccallum, Kachites, A., Thurn, Sebastien, Mitchell, & Tom. (2000). Text Classification from Labeled and Unlabeled Documents using EM. Machine Learning, 39, 103–134, 2000., 103-134.

[27] Rapacz, S., Chołda, P., & Natkaniec, M. (2021). A Method for Fast Selection of Machine-Learning Classifiers for Spam Filtering. Electronics 2021, 10, 2083, 1 - 23.

[28] Sharma, P., & Bhardwaj, U. (2018). Machine Learning-based Spam E-Mail Detection. International Journal of Intelligent Engineering and Systems, Vol.11, No.3, 2018, 1 - 10.

[29] Shirani-Mehr, H. (2019). SMS Spam Detection using Machine Learning Approach. 1-4.

[30] Shuaib, S. M., & Osho, O. (2018). Comparative Analysis of Classification Algorithms for Spam Detection. I. J. Computer Network and Information Security, 60 - 67.

[31] Siddique, Bin, Z., Khan, Ali, M., Din, Ud, I., . . . Shah, I. M. (2021). Machine Learning-Based Detection of Spam Emails. 1-11.

[32] Zhou, Z.-H. (2021). Machine Learning. 1-20.

[33] Adugna, T., Xu, W., & Fan, J. (2022). Comparison of random forest and support vector machine classifiers for regional land cover mapping using coarse resolution FY-3C images. Remote Sensing, 14(3), 574.

[34] F. Silas, Y. Musa, and S. Joyce, "Empirical Performance of Internal Sorting Algorithm," British Journal of Mathematics & Computer Science, vol. 20, pp. 1-9, 2017.

[35] Y. Musa, B. Rotimi, and O. Peter, "Application of Lasso regression to model national development indicators and national internet usage," Journal of Scientific Research and Reports, vol. 21, pp. 1-10, 2019.