Integrated Multi-Model Approach for Spam Mail Detection: A Comprehensive Analysis

Mounif El Khatib
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
mae133@mail.aub.edu

Saadallah Itani
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
smi11@mail.aub.edu

Hussein Missilmani
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
ham65@mail.aub.edu

Samer Saade
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
shs32@mail.aub.edu

Jad Ghamloush
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
jag09@mail.aub.edu

Mariette Awad
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
mariette.awad@mail.edu.lb

Abstract – In the digital era, where email communication is pivotal, the incessant influx of spam mails presents a significant challenge. Spam mail, often unsolicited and irrelevant, not only clutter inboxes but also poses severe security risks, ranging from phishing scams to malware dissemination. Effective spam detection is thus crucial for maintaining the integrity and security of email communication. We trained various ML models to classify emails under "Phishing" or "Safe", such as artificial neural networks, SVM and logistic regression.

Keywords – Spam mail, Phishing mail, Machine Learning, Binary Classification, Logistic Regression, SVM, K-NN, Artificial Neural Networks, Bayes Model, TFIDF Vectorizer.

I. Introduction

Email is a key part of how we communicate today, but it has a big problem: spam mail. These are unwanted messages that fill up our inboxes, and they can range from annoying ads to dangerous scams and viruses. The issue of spam is not just annoying – it's a serious threat to our online safety and productivity.

Every day, tons of spam emails are sent around the world, making it hard for people to find the important emails they actually need to read. What's worse, some of these spam emails are used to steal personal information or spread harmful software. This is a big concern for everyone, from regular email users to big companies.

Spam emails are also getting smarter. They use tricks that can fool basic spam filters, making it harder to stop them. That's why it's important to create better ways to detect and block spam emails. This isn't just about keeping our inboxes clean; it's about protecting ourselves from online dangers and making sure email remains a safe and useful way to communicate. So, finding better solutions for spam mail is a big deal and something we need to focus on.

Various spam filtering approaches have been summarized, highlighting their accuracy and challenges. Naïve Bayes and SVM algorithms are commonly used for email spam filtering. Researchers are aiming to develop next-generation spam filtering mechanisms to handle large volumes of multimedia data effectively [1].

Deep learning algorithms have shown promise in improving the accuracy of intrusion and spam detection over traditional machine learning and lexicon models. These algorithms have been evaluated using diverse cyber datasets categorized based on different types of network traffic and applications [1].

The Naïve Bayes method is noted for its speed and decent precision in filtering spam emails, while SVM and ID3 methods offer greater precision but require more time for system construction. The choice of learning algorithm depends on the situation and the balance between accuracy and timing [1].

The following subsections present the methodology, a brief description of the dataset and analysis of our findings.

II. Dataset Description

The dataset we used was sourced from Kaggle. It is composed of three columns, where the first column indicates the mail number, the second column contains the mail itself and the third column specifies the label of each mail. It contains around 18600 entries where 61% of the mails are safe, and 39% are phishing mails. There are two possibilities for each mail: it's either a phishing mail or a safe mail. This database seemed the best fit for our project due to its large number of features.

III. Methodology

To carry out our study, we employed multiple ML models: Artificial Neural Networks, Logistic Regression, K-Nearest Neighbors, SVM and Naïve Bayes. For each of those models, we preprocessed the data by dropping the entries where the values are N/A and made sure that the rows picked fall within the labels "Phishing Email" and "Safe Email". We also transformed all uppercase letters into lowercase since text formatting does not matter in our case when tokenizing words. 80% of the data was used for training and 20% was used for testing. For each model, we converted text into numerical data using TFIDF vectorizing which assigns weights to each word in the document, highlighting important words which is crucial for classification.

Let's dive deeper into each model's implementation:

A. Overview of Experimental Design

Artificial Neural Networks (ANNs) are computing systems inspired by the biological neural networks that constitute our brains. They are composed of interconnected nodes which constitute an input layer, several hidden layers, and an output layer. In our study we used them to classify whether an email is considered safe or not.

Let's take a deeper look at the implementation of our neural network. The network is built using a sequential architecture consisting of an input layer, five hidden layers and an output layer. Regularization is applied to combat overfitting, alongside dropout layers. The input and hidden layers use a 'ReLU' activation function, while the output layer uses ‘sigmoid’ activation which is suited for binary classification.

We chose ReLU as our activation function because it allows the model to learn faster and perform better. The labels of our dataset can be mapped into binary values, hence our choice of the sigmoid function as the activation of the output layer. The sigmoid function maps output into the range [0, 1].

- The ReLU function is defined as such:

$$f(x) = \max(0, x)$$

- The Sigmoid function is defined as such:

$$y = \frac{1}{1 + e^{-z}}$$

The model is compiled using “binary_crossentropy” as the loss function which is suitable for binary classification tasks, “adam” optimizer and “accuracy” as a performance metric. We employed callbacks such as early stopping and reducing learning rate when a metric stops improving which helps prevent overfitting. The model is trained for 60 epochs with a size of 32, employing callbacks for early stopping and learning rate reduction.

The other proposed model is the **logistic regression** model. Its primary function is to discern between phishing emails and safe ones. This tool proves particularly valuable in understanding the contribution of various features towards an email’s likelihood of being malicious; it allows us not only to predict threats but also comprehend their underlying factors.

In this study – using logistic regression as our analytical approach – we scrutinize a dataset of emails with the aim of uncovering how different features influence the probability that an email could potentially be a phishing attempt.

The logistic regression model bases itself on the sigmoid function (defined earlier), commonly known as the logistic function. The coefficients represent the relationship between the features. The logistic regression model estimates the log-odds of an email being a phishing attempt – based on a linear combination of features. Higher log-odds elevate the probability that the email is a safe email.

Support Vector Machine (SVM) is a powerful supervised machine learning algorithm used for both classification and regression tasks. It works by finding the best hyperplane that separates different classes in the feature space. It’s chosen to maximize the margin between the classes.

We chose to use an SVM classifier due to its effectiveness in binary classification tasks. It can handle high-dimensional data, making it well-suited for text classification as it can be tuned with different kernel functions.

Linear Kernel: Once the data has been vectorized using TF-IDF, it resulted in a very high number of features. In this case, a linear approach is often good enough to tell apart different types of emails, like phishing and safe mails. Linear kernels are known to perform well in these cases. In addition, other kernels cause overfitting, and they result in much lower test accuracy compared to the linear kernels. Additionally, linear kernels have less parameters to tune, and are faster.

One of the other models that we adopted, **K-Nearest Neighbors** (KNN), is a simple but effective algorithm used for classification tasks in ML. It operates on the principle that similar things exist in close proximity. The algorithm measures the distance between points and classifies a new data point based on the majority class among its nearest neighbors.

We used grid search to identify the best combination of parameters that yield the best performance for our KNN model. Grid search uses cross-validation, which splits the data into 5 folds and performs training validation on these folds multiple times. These are the optimal parameters:

metric	cosine
n_neighbors	6
weights	distance

The last model we chose to implement is **Naïve Bayes**, which is a probabilistic ML algorithm based on Bayes' Theorem which assumes independence among predictors. It calculates the probability of each class and the conditional probability of each class given the input value, then selects the class with the highest probability. This is effective for large datasets – like the one we are using.

Naïve Bayes is effective in spam email detection due to its ability to handle large volumes of data efficiently and its effectiveness with text data. The algorithm can quickly train on a spam dataset and classify new emails by calculating probabilities of words.

B. Data Collection and Preprocessing

Before performing any computations, the data was split into training and testing sets (80%-20%). TF-IDF vectorization – which is responsible of converting email text into numerical data – was used to extract features. In the implementation of ANNs and Logistic regression, the labels were mapped into binary values.

The dataset contained a few N/A entries, so we decided to drop them, while also converting all text to lowercase.

C. Evaluation Metrics

All models were evaluated based on accuracy scores. A classification report was also generated.

The Accuracy Score – a vital metric in binary classification that reflects the model's correctness – signifies the ratio of correct predictions to total predictions. In our specific task, which is distinguishing phishing emails from safe emails, accuracy becomes even more important: it serves as an indicator for how well our model is performing.

IV. Results and Analysis

- Artificial Neural Network

Evaluation Metric	Neural Network
Accuracy	97.1
W-f1_score	97
W-recall	97
W-precision	97

- Logistic Regression

Evaluation Metric	Logistic Regression

Accuracy	97.2
W-f1_score	97
W-recall	97
W-precision	97

- Support Vector Machine

Evaluation Metric	Support Vector Machine
Accuracy	97.69
W-f1_score	98
W-recall	98
W-precision	98

- K-Nearest Neighbors

Evaluation Metric	K-Nearest Neighbors
Accuracy	93.69
W-f1_score	94
W-recall	94
W-precision	94

- Naïve Bayes

Evaluation Metric	Naïve Bayes
Accuracy	89
W-f1_score	89
W-recall	89
W-precision	90

In addition to this, we generated the confusion matrix for the Naïve Bayes model, and this is what we got:

True Positives: 2172 emails correctly identified as phishing.
True Negatives: 1154 emails correctly identified as safe.
False Positives: 364 safe emails incorrectly identified as phishing.
False Negatives: 38 phishing emails incorrectly identified as safe.

V. Conclusion

The ANN's classification report indicates that it is highly performant in spam detection. Its overall accuracy is 97.1%, which demonstrates the model's effectiveness.

The Logistic Regression model's accuracy is 97.13% which demonstrates its effectiveness in detecting phishing mails.

The Support Vector Machine model's accuracy is by far the highest among all models, coming in at 97.69%, which shows its high effectiveness in correctly classifying mails. This accuracy suggests a low number of false positives, which is important for preventing real emails from being wrongly classified as spam.

The KNN model achieved an accuracy of 93.6% which is decent, but we were not able to achieve a higher accuracy since the accuracy of K-Nearest Neighbors in text classification tasks may sometimes be lower than other models. This is due to a multiple of factors, such as high dimensionality of text data, parameter sensitivity, etc....

The Naïve Bayes model achieved the lowest accuracy of them all, which might be due to the fact that it assumes that features are independent of each other, which is often not the case in language data.

We can therefore conclude that the best model for this problem is SVM.

VI. References

[1] Bhuiyan, M. Z. A., Rahman, M. M., Billah, M. M., & Karim, A. (2020). Machine Learning Techniques for Spam Detection in Email and IoT Platforms: Analysis and Research Challenges. Hindawi.

“pandas documentation — pandas 2.1.3 documentation.” https://pandas.pydata.org/docs/

“Documentation scikit-learn: machine learning in Python — scikit-learn 0.21.3 documentation.” https://scikit-learn.org/0.21/documentation.html

“1.4. Support vector machines,” Scikit-learn. https://scikit-learn.org/stable/modules/svm.html

“sklearn.linear_model.LogisticRegression,” Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

“1.17. Neural network models (supervised),” Scikit-learn. https://scikit-learn.org/stable/modules/neural_networks_supervised.html

“Sklearn.feature_extraction.text.TfidFVectorizer,” Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

“1.9. Naive Bayes,” Scikit-learn. https://scikit-learn.org/stable/modules/naive_bayes.html

“1.6. Nearest neighbors,” Scikit-learn. https://scikit-learn.org/stable/modules/neighbors.html

Dataset: https://www.kaggle.com/datasets/subhajournal/phishingemails/data