E-Mail Spam Classification

YZV 311E Term Project Proposal

Abdullah Bilici
Artificial Intelligence & Data Engineering
Istanbul Technical University
bilicia@itu.edu.tr
150200330

Bora Boyacıoğlu
Artificial Intelligence & Data Engineering
Istanbul Technical University
boyacioglu20@itu.edu.tr
150200310

***Abstract*—This documents is the intermediate report for the YZV 311E Data Mining course term project. It is about an E-Mail classification model using techniques like Natural Language Processing. It is being held by Abdullah Bilici and Bora Boyacıoğlu, Istanbul Technical University. Team number is 8.**

***Index Terms*—email, spam, classification, natural language processing**

I. Introduction

Today, Email has become a ubiquitous mode of communication, bringing with it the pervasive issue of spam emails. These messages not only fill up inboxes but also bring significant security risks, ranging from scams to the spread of malware. As the volume of email traffic grows, effectively separating spam from harmless emails has born as a big challenge. The importance of this challenge is known by the fact that a significant portion of all email traffic is spam, bringing a necessity to develop effective spam detection systems.

This project, conducted as a term project of YZV 311E Data Mining course at Istanbul Technical University, aims to develop an email spam classification model. By combining techniques from Natural Language Processing (NLP) and Machine Learning (ML), our goal is to create a model that can accurately and efficiently differentiate spam emails from legitimate ones. The project is conducted by Abdullah Bilici and Bora Boyacıoğlu.

Our approach involves analysing a dataset of emails, each labeled as either **spam** or **ham** (non-spam), to train various classification models. These models include Naive Bayes, Support Vector Machines (SVM), Random Forest, and Logistic Regression. Each has unique strenghts on text classification tasks. Through experimentions and analysises, we aim to identify the most effective model for this purpose.

This report presents an intermediate overview of our project, detailing our methodologies, preliminary findings, and insights so far. It aims to represent our current progress and provide a basis for further development.

II. Related Work

A. Literature Review

There are examples throughout the internet about spam classification. One of them is a Kaggle competition [1]. It is a competition held in 2011. It is actually a text classification competition, not restricted by emails. Of course the idea is the same. Texts were collected from social media (blogs). There are 7086 lines labeled as spam or ham. And 33052 lines of unlabeled data. The goal is to classify the unlabeled data.

Another project is a research paper [2]. It is a comprehensive study about spam classification. Their dataset consits of 6047 messages, with 1897 of them being spam. They used additional models, and get around 98.9% accuracy. We will try to achieve similar results with our dataset and models.

Other than that, we had a lot of small projects to analyse. The models we tried are used in a work of Balaka Biswas [3]. The models are **Naive Bayes**, **Support Vector Machines (SVM)** and **Random Forest** and **Logistic Regression**. We considered to use this models in our project.

There is also a study conducted by Faisal Kureshi [4]. Only **Multinomial Naive Bayes** is used in this project and a 97.8% of accuracy is achieved.

B. Importance

Spam emails are a big problem for most of the Internet users, which is almost everybody from age 7 to 70. They are both annoying and also may be dangerous. Everyday, most of the emails we get are spam emails [5]. This will require the aware people to constantly block and delete spam emails. This is a waste of time and resources. In fact, I get around 14 spams everyday. This is a big problem for me and I am sure it is a big problem for most of the people.

However, this is a bigger problem for non-aware users and big companies, as security becomes an issue. Spam emails may contain viruses or phishing links. They may also be used for identity theft. This is a big problem for companies as well. They may be used to steal information or to send viruses to the company's computers. They may also be used to steal money from the company.

These are the resons on why a good classification is crucial. In this study, our goal is to build a model with a high accuracy. We will try to achieve this by using various models and techniques.

III. Proposed Work

A. Data Preparation

On transforming email data into a tabular format, we initially broke down sentences into individual words through tokenization, ensuring uniformity by converting them to lowercase and removing stopwords and punctuation.

For this, we used an English language model. Using this model, we tokenized each sentence. Then, We collected the stopwords of English model and removed them from the tokenized sentences. We also removed the punctuations from the tokenized sentences. After that, we converted all the words to lowercase.

Next, in order to determine how frequently each term occurred in a particular email, the Term Frequency (TF) was computed. Additionally, uniqueness of phrases throughout the whole dataset was evaluated using the Inverse Document Frequency (IDF). We used TF and IDF to compute TF-IDF matrix, where each row corresponds to a document and each column corresponds to a unique term in the entire dataset. Values gives information about importance of token in a document amongst a collection of corpus.

In order to work with this data, we needed to reduce the amount of columns. Without this, we would have to work with 16908 columns, which would consume a lot of time and resources. We simply selected the top 10% of features and used them for our models. This reduced the number of columns to 1691.

B. Model Selection

There are four models defined in our project proposal: **Naive Bayes**, **Support Vector Machines (SVM)**, **Random Forest** and **Logistic Regression**. Before using them, it is crucial to know their strengths and weaknesses. We will explain them in this section.

1) Naive Bayes:

$$P(c|d) = \frac{P(c)\prod_{1\leq i\leq n} P(f_i|c)}{P(d)}$$

This model is based on Bayes' Theorem. It is a probabilistic model that uses the probability of each attribute belonging to each class to make a prediction. It is called naive because it assumes that all the attributes are independent of each other. This assumption is not true in real life but it is still a good model for classification problems. It is also a fast model to train and predict. For text classification problems, it is a preferable model.

2) Support Vector Machines (SVM):

$$\min_{w,b,\xi} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$\text{subject to } y_i(w\cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

This model is based on the idea of finding a hyperplane that separates the data into classes. It is a supervised learning model that can be used for both classification and regression problems. For complex problems, SVM is a powerful model to use.

3) Random Forest:

$$Y = \frac{1}{N}\sum_{i=1}^{N} h(X,\Theta_i)$$

This model is based on the idea of creating multiple decision trees and combining them to get a better result. It is a supervised learning model that can be used for both classification and regression problems. It is a powerful model that can be used for complex problems.

4) Logistic Regression:

$$P(Y=1|X) = \frac{1}{1+e^{-(\beta_0+\beta_1 X)}}$$

This model is based on the idea of finding the best fitting S-shaped curve for the data.

IV. Experimental Results

We tried four different models with the goal of finding the one that pursues our goal the best. As stated in the project proposal, our models are **Naive Bayes**, **Support Vector Machines (SVM)**, **Random Forest** and **Logistic Regression**. Each one has their own advantages and disadvantages. We will try to find the best one for our problem.

A. Naive Bayes

At the first trial, we used the **Multinomial Naive Bayes** model for our problem. It is a variation of the Naive Bayes model that is used for text classification problems. It is based on the multinomial distribution of the data. The results were not that great, however. One thing about Naive Bayes is that it performs poorly when the data violates the independence assumption.

We saw low results, especially with the recall value. You can see the Validation Set results in Table I and the Test Set results in Table II.

TABLE I
Naive Bayes Validation Set Results

Accuracy	Precision	Recall	F1 Score
0.8392	1.0	0.3881	0.5592

True Positives: 111
False Positives: 0
False Negatives: 175
True Negatives: 802

TABLE II
Naive Bayes Test Set Results

Accuracy	Precision	Recall	F1 Score
0.8566	1.0	0.4046	0.5761

True Positives: 106
False Positives: 0
False Negatives: 156
True Negatives: 826

B. Support Vector Machines (SVM)

In our experiment, we used **SVM** after we tried Naive Bayes. There definitely is an improvement. You can see the Validation Set results in Table III and the Test Set results in Table IV. However, the results may be better. We will continue trying to find a better model.

TABLE III
SVM Validation Set Results

Accuracy	Precision	Recall	F1 Score
0.9586	0.9959	0.8462	0.9149

True Positives: 242
False Positives: 1
False Negatives: 44
True Negatives: 801

TABLE IV
SVM Test Set Results

Accuracy	Precision	Recall	F1 Score
0.9504	0.9906	0.8015	0.8861

True Positives: 210
False Positives: 2
False Negatives: 52
True Negatives: 824

C. Random Forest

This time, all scores are above 90%, which is a sign that this is the correct model. Especially with hyperparameter tuning, these scores may be improved. Of course we will try yet another model, but Random Forest seems to be the best one so far. You can see the Validation Set results in Table V and the Test Set results in Table VI.

We used `n_estimators=100` and `random_state=42` as our parameters for Random Forest.

TABLE V
Random Forest Validation Set Results

Accuracy	Precision	Recall	F1 Score
0.9779	0.9852	0.9301	0.9586

True Positives: 266
False Positives: 4
False Negatives: 20
True Negatives: 798

TABLE VI
Random Forest Test Set Results

Accuracy	Precision	Recall	F1 Score
0.9798	0.9839	0.9313	0.9569

True Positives: 244
False Positives: 4
False Negatives: 18
True Negatives: 822

D. Logistic Regression

After Random Forest, this model was a disappointment. It is a good model but it did not perform well for our problem. You can see the Validation Set results in Table VII and the Test Set results in Table VIII.

We used `max_iter=10000` as our parameters for Logistic Regression.

TABLE VII
Logistic Regression Validation Set Results

Accuracy	Precision	Recall	F1 Score
0.8860	1.0	0.5664	0.7232

True Positives: 162
False Positives: 0
False Negatives: 124
True Negatives: 802

TABLE VIII
Logistic Regression Test Set Results

Accuracy	Precision	Recall	F1 Score
0.8898	1.0	0.5840	0.7373

True Positives: 153
False Positives: 0
False Negatives: 109
True Negatives: 826

V. Conclusion

We applied various models to a TF-IDF matrix derived from email data. Our strategy is to model a neural network that can enhance base model's performance. Our approach is to feed tokenized data into Bert [6] model to get representations of each sentence then fitting a fully connected neural network to this data to improve performance of base model.

The BERT (Bidirectional Encoder Representations from Transformers) model is a type of transformer-based neural network architecture developed for natural language processing (NLP) tasks. BERT considers the context of each word by analyzing the entire sentence bidirectionally. This bidirectional understanding allows BERT to capture intricate relationships and dependencies between words, leading to more accurate representations of given data.

BERT is trained on large data and can be used for specific NLP tasks, such as text classification. This can be done by using tokens [CLS] [6]. The [CLS] token is a special token used to represent the entire input sequence in a sentence or document. This [CLS] tokens will help us to fit our data and make predictions.

We will use several metrices to compare base model and NN model. Accuracy, AUC-ROC curve and F1-score are few of the metrices we will use.

References

[1] Wolverine. (2011). UMICH SI650 - Sentiment Classification. Kaggle. kaggle→competitions→si650winter11

[2] Adnan, Muhammad & Imam, Muhammad & Javed, Muhammad & Murtza, Iqbal. (2023). Improving spam email classification accuracy using ensemble techniques: a stacking approach. International Journal of Information Security. 1–13. 10.1007/s10207–023–00756–1.

[3] Biswas, B. (2019). Email Spam Classification. kaggle.com→balaka18→email-spam-classification

[4] Kureshi, F. (2022). Email Spam Classification. kaggle→mfaisalqureshi→email-spam-detection-98-accuracy

[5] Cveticanin, N. (2023). 20 SPAM Statistics for 2023. dataprot→statistics→spam-statistics

[6] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). **BERT:** Pre-training of Deep Bidirectional Transformers for Language Understanding. North American Chapter of the Association for Computational Linguistics.