# Spam Email Detection Using Advanced NLP Techniques and Deep Learning Models

**Alisher Amerkhojayev**
230056918
Master of Data Science
alisher.amerkhojayev@city.ac.uk

## 1 Problem statement and Motivation

The overarching aim of this project is the development of an effective spam email classification system through state-of-the-art NLP techniques and deep learning models. Spam emails, unsolicited messages sent in their bulk, create burdensome challenges both for individuals and organizations. Mostly, unwanted communications clutter up inboxes, waste users' time, and sometimes cause serious security threats through phishing attacks. Modern spam filters, despite their improvements, cannot handle the sophisticated spamming techniques that develop continuously and hence require more advanced and adaptive solutions.

Traditional spam filters are usually a rule-based system and basic machine learning algorithms that are quite rigid and hence often less effective against advanced spamming techniques. These methods may not handle the high variability and subtlety of spam emails effectively to avoid false positives—legitimate emails marked as spam—or false negatives—spam emails passing through the filter. Therefore, this project seeks to address these limitations by incorporating the potentiality of state-of-the-art NLP models and deep learning techniques to ensure enhanced accuracy and robustness of spam detection systems. We want to train models on large datasets of labeled emails with the goal of developing a system that can understand the subtle differences between spam and legitimate emails, thereby enhancing overall detection rates and minimizing erroneous classifications.

The volume and sophistication of spam emails are growing. According to recent studies, around 49% of global email traffic is spam (Ellis, 2023). This is not only a nuisance but has significant financial and productivity costs for businesses. Moreover, the increasing incidents of phishing attacks and other email-based cyber threats drive home the critical need for more advanced spam detection mechanisms.

Furthermore, improvements in NLP and deep learning open the door to revolutionizing the field of spam detection systems. State-of-the-art models, such as transformer models (for example, BERT, DistilBERT), have shown fantastic successes in different NLP tasks—text classification, sentiment analysis, and language translation, among others. Applying these cutting-edge models to spam detection could provide us with an adaptive and accurate spam filter that would keep up with spammers' evolving strategies.

## 2 Research hypothesis

The proposed approach using advanced NLP approaches and deep learning models such as transformer architectures offers great potential for spam email categorization. These models have already shown the power to process and understand sophisticated language patterns. The semantic meaning of the messages captured by such models, particularly transformers like DistilBERT, captures fine details—information very critical in marking emails as spam or non-spam. In this regard, the hypothesis will be whether using DistilBERT will provide a massive increase in spam email detection accuracy and resilience over traditional machine learning.

We are going to test the hypothesis against traditional machine learning techniques such as SVM, Naïve Bayes, and Logistic regression coupled with vectorizes such as TF-IDF and BoW.

As a result, we expect a drastic reduction of both false positives and false negatives.

## 3 Related work and background

Wu et al. (2017) introduced an approach for Twitter spam detection by converting tweet text into word vectors using the Word2Vec technique. Their binary classification model constructed using machine learning algorithms like Random Forest, MLP, and DT, achieved an impressive accuracy of 95%, surpassing previous methods which averaged 87%. This study highlighted the efficacy of Word2Vec for feature extraction, demonstrating excellent performance across various datasets. Dada et al. (2019) conducted a study on how machine learning techniques can enhance email spam filtering by using tokenization to break down text into meaningful units. They evaluated models such as NB, Neural Networks, SVM, and DT, concluding that integrating sophisticated methods like deep learning and adversarial learning could significantly improve spam filters' efficiency in adapting to the dynamic nature of spam threats. Building upon these initial findings, Kontsewaya et al. (2021) study leveraged a variety of machine learning models, notably NB, KNN, SVM, LR, DT, and Random Forests. They found that Logistic Regression and Naïve Bayes achieved the highest accuracy 99% in email categorization. In parallel, Dedeturk et al. (2020) explored spam detection across diverse datasets, including TurkishEmail, CSDMC2010, and Enron. Using TF-IDF and a LR model augmented by an artificial bee colony algorithm, they demonstrated marked improvements in handling complex, multidimensional data, outperforming traditional classifiers like SVM and Naïve Bayes in terms of accuracy and efficiency.

Jain et al. (2019) enhanced text representation by including semantic information from WordNet and ConceptNet. They employed a hybrid deep learning model that integrated CNN and LSTM networks, leveraging both local and long-range textual relationships. Their model with the Sequential Stacked CNN-LSTM (SSCL) model attaining 99.01% accuracy and a 99.29% F1-score for the SMS spam dataset.

Further advancing the field, AbdulNabi and Yaseen (2021) optimized the BERT transformer model for email spam detection, utilizing its robust attention mechanisms to capture contextual word connections. Pre-trained on large text corpora and fine-tuned for spam classification, BERT demonstrated exceptional performance with an accuracy of 98.67% and an F1 score of 98.66%, highlighting its superior capability in distinguishing spam from legitimate emails. Expanding the application of transformer models, Xu et al. (2021) focused on social network spam detection using ALBERT, a more compact version of BERT. They employed preprocessing methods including tokenization, stop word elimination, and emoticon deletion. By combining ALBERT with a Bi-LSTM network enhanced by a self-attention mechanism, their model demonstrated superior precision and F1-score, reaching a maximum F1-score of 90.1% on the Weibo and microblogPCU datasets. Moreover, Tida and Hsu (2021) study preprocessing included tokenization, stop word elimination, and normalization. They employed BERT due to its bidirectional training capability and context comprehension efficiency. The model design featured fully connected linear layers, batch normalization, dropout layers, and ReLU activation, optimized through hyperparameter tuning. Tested on datasets like Enron, SpamAssassin, LingSpam, and SMS Spam Collection, their model achieved a remarkable accuracy rate of 97% and an F1-score of 0.96, corroborating BERT's effectiveness in spam detection. In a comparative analysis of transformer models, Kotni et al. (2022) utilized BERT, DistilBERT, and ALBERT for spam detection across datasets from SMS, Twitter, and URLs. Their trials revealed that DistilBERT surpassed other models, achieving the highest accuracy and F1-scores, emphasizing the efficiency and effectiveness of DistilBERT in detecting spam across different datasets.

Lastly, addressing the challenge of unbalanced data, Makkar and Kumar (2020) proposed the 'Split by Over-sampling and Train by Under-fitting' (SOTU) strategy. They used LSTM networks to analyze link properties and neural networks for content features. Validated on the WEBSPAM-UK2007 dataset, their approach

achieved an accuracy of 95.25%. The study highlighted the LSTM network's superior accuracy.

## 4 Accomplishments

- Task 1: Preprocess dataset – Completed
- Task 2: Build baseline models and train (NB, LR +BoW, TF-IDF) on collected dataset and examine its performance - Completed
- Task 3: Build and train SVM +Word2Vec, TF-IDF model and examine its performance - Completed
- Task 4: Task 3: Build and train DistilBERT model and examine its performance – Completed
- Task 5: Build and Train RNN-LTSM model and examine its performance – Failed due time constraints
- Task 6: Perform in-depth error analysis to figure out what kinds of examples our approach struggles with – Completed.
- Task 7: Build another model with SVM + BoW or TF-IDF - Completed

## 5 Approach and Methodology

The first model is an SVM Classifier in conjunction with TF-IDF and Word2Vec embeddings, which perform the task of discrimination between spam and non-spam emails. This hybrid feature extraction approach will ensure that both local and contextual information about the text is captured.

We begin by cleaning the email text, removing HTML tags, special characters, and extra whitespaces, followed by tokenization, stopword removal, and stemming to normalize the text. The vectorization of text is done by TF-IDF, and a Word2Vec model is trained to generate the word embeddings. Therefore, we can represent each email as an average of its word embeddings. We standardize the vectors of Word2Vec and concatenate them with the TF-IDF features to get a full feature set for each email, encoding the email categories into numerical labels.

To avoid class imbalance, the data is divided into training and testing sets. The minority class in the training set is upscaled. Then we train an SVM with balanced class weights by using GridSearchCV to find the best hyperparameters. The model is then tested on the test set, using classification metrics and a confusion matrix for evaluating its performance.

Our model is robust, but it is subjected to some limitations with the traditional baselines: handling highly imbalanced data, the potential overfitting problem due to high-dimensional feature space, and generalization problem with unseen data. But our hybrid approach tries to capture more subtle patterns than the simpler baselines use just one type of feature.

Libraries that we used include pandas for data manipulation, numpy for numerical operations, re for regular expressions, NLTK for text preprocessing, scikit-learn for machine learning utilities, gensim for Word2Vec, scipy for sparse matrices, and seaborn, matplotlib for data visualization.

Our second model for detecting email spam is based on the DistilBERT model, a transformer model known for its efficiency in NLP tasks. The major steps in the methodology include: data preprocessing and label encoding, that is, spam labels converted into numerical values, splitting data into the training and test sets, and handling class imbalance by up-sampling the minority class for balanced training data. Further, we will be tokenizing the email texts using DistilBertTokenizer, making them ready for the DistilBERT model.

The model is fine-tuned by adapting and utilizing the Trainer API of Hugging Face: defining the arguments for training, evaluation strategies, and training based on the preprocessed data. After training, we will evaluate the performance of the model using different metrics, including accuracy, precision, recall, and F1-score. Additionally, we will generate a confusion matrix to visualize the classification performance and identify the

misclassified emails to understand the model's weaknesses.

Compared to the traditional models, such as Naive Bayes or SVM, DistilBERT captures information about the context in the emails better. On the other hand, it may face most of the same limitations, such as overfitting and class imbalance issues, despite resampling efforts. Transformer models also require remarkable computational power, which may be a limitation compared to other more simplistic models. We tried to mitigate these by using upsampling and training the models with proper care.

A complete working implementation was done, involving data preprocessing, tokenization, model training and evaluation, and visualization. All these components are strong parts of a framework for email spam detection. The implementation was done using the Hugging Face transformers library intensively for model training and tokenization, based on the available documentation. Relevant resources include the Hugging Face Transformers Documentation (Hugging Face).

The following libraries were used: pandas for data manipulation; numpy for numerical operations; scikit-learn for data splitting, label encoding, resampling, and evaluation metrics; transformers, from Hugging Face, for tokenization and model training; torch for the definition of custom datasets and manipulation of tensors; and seaborn and matplotlib for visualization. These libraries offered ease in the implementation of the email spam detection pipeline.

# 6 Dataset

## 6.1 Introduction to the dataset

The dataset utilized for this project was taken from Kaggle labelled as “Spam email classification” (Kaggle). It provides an Excel file which would immensely help our study as we would not need to make any structural changes. However, it was important to notice that the last row of the dataset did not contain an entry but a text which is not part of the dataset which we removed. The dataset consists of 5572 entries of various emails containing different structures, and lengths and addressed to numerous receivers. However, it is important to notice that there are duplicates of text, we will keep the duplicates for the sake of checking if the models will detect it or not, whether they would label it as spam since they repeat or not. For the classification, 87% (4851) of the total emails have been labelled as ham which is a genuine email, and 13% (749) are labelled as spam which are the spam emails shown in Figure 1. Examination of this dataset is of great importance as it would affect the accuracy, effectiveness, and outcome of our study. Choosing the right dataset or modifying it appropriately could potentially affect the results.

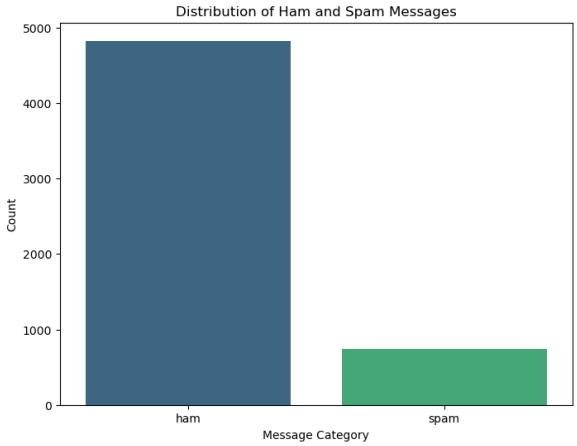



*Figure 1*

## 6.2 Examples

In this section, we will provide email examples of spam to illustrate the task we will be working on. Firstly, we demonstrate the spam emails:

- URGENT! Your Mobile No. was awarded Â£2000 Bonus Caller Prize on 5/9/03 This is our final try to contact U! Call from Landline 09064019788 BOX42WR29C, 150PPM

- UpgrdCentre Orange customer, you may now claim your FREE CAMERA PHONE upgrade for your loyalty. Call now on 0207 153 9153. Offer ends 26th July. T&C's apply. Opt-out available

- You have WON a guaranteed Â£1000 cash or a Â£2000 prize. To claim yr prize call our customer service representative on 08714712394 between 10am-7pm

- Hi I'm sue. I am 20 years old and work as a lapdancer. I love sex. Text me live - I'm i my bedroom now. text SUE to 89555. By TextOperator G2 1DA 150ppmsg 18+

These are only a few of the examples taken, however, we can already notice a few similarities that are present throughout. Firstly, we notice that there is a considerable amount of words that are capslocked highlighting the fact that something is of urgency. Secondly, there is some kind of indication that a person has won cash or could get a free item grabbing the attention of the reader. Thirdly, all of these emails contain either a phone number or a website link that needs to be called or went to in order to claim these prizes. If the other parts have been there to grab the attention of the reader, this is where the main purpose of these messages is. The caller would have to pay to use these landlines at increased prices where they will be talked to in a roundabout way to get as much money as possible and collect information on the user. Our task is to find these spam messages and label them appropriately.

### 6.3 Properties of the dataset that make this task challenging

When it comes to this dataset, there are a few caveats which need to be addressed:

**The Class Imbalance**: this is the first thing that could be noticed as the dataset has ham as the majority class where it heavily outweighs the spam emails by a considerable fraction (87% to 13%). This could lead to potential bias towards the majority class which could result in poor classification. Moreover, this could lead to overfitting as the spam emails has lower samples which could introduce bias where the accuracy scores would not be representative and would have poor generalisation.

**The Data Quality**: Some of the emails sent have a structure of accidental, unintelligible, or nonsensical writing. This could introduce noise to the models which could potentially label these messages as spam even though they were genuine. This is still an important piece of information as not all emails are structured right or could be of a formal or serious nature.

**Linguistic Nuances**: Text-based spam filters must handle the linguistic diversity found in misspellings, grammatical mistakes, and the utilization of non-standard vocabulary employed by spammers to avoid being detected. The messages have been adopted specifically to relate to the ham messages which would try to imitate them.

### 6.4 Data Preprocessing

For Data Preprocessing we utilize several different methods to ensure they fit our project for the detection of spam emails. Firstly, since we are dealing with various emails, we need to make sure we remove all the things that could potentially hinder our model. We clean the data from any URLs present as they do not contain any important information and only introduce noise. Next, we remove all the numbers and special characters which are also not needed for the purpose of detecting spam messages. Moreover, we remove any multiple steps that are created after we remove the URLs, special characters, and numbers.

After we have cleaned our data we can move on to the next stage of our preprocessing where we will be normalizing the data. Firstly, we tokenize the emails and ensure that all of the text is in lowercase. Secondly, we remove the stopwords from the text as these are the words most common in text which do not carry any meaning and only introduce noise to our data. Next, we apply stemming. Our implementation of stemming has proved to the better choice after we have tested it in our baseline models. We use snowballstemming as it is better over large datasets than the porterstemmer but slightly lacks in speed (Nagpal, 2024). The stemming is used to reduce vocabulary size and improve generalization.

Another important preprocessing task that we employ is the upsampling of the minority class, in our case it is the spam messages. The training is done on the upsampled data, while it is tested on the actual data. This is done to tackle the problem of class imbalance mentioned before. This was actually one of the difficulties imposed for our project as it could create bias in the models. We were left with a choice of using different methods of tackling this problem and have settled on using the upsampling method as SMOTE method could have also introduced as it would create random

messages which we would not be sure if it was actual spam or not.

## 7 Baselines

There are several baselines used for this project. Firstly, the baseline models which are used in this project are the Naïve Bayes and the Logistic regression. These were used in conjunction with two different feature extraction techniques, namely Bag of Words (BoW) and Term frequency-inverse document frequency (TF-IDF). In accordance to our preprocessing techniques, we cleaned and normalized the data and applied upsampling. The Naïve Bayes with BoW had an accuracy of 96.7%, precision of 97.1%, recall of 90.6%, and F1 Score of 93.8%. Interestingly, it had outperformed the TF-IDF with the same model within all of the evaluation metrics. The same is noticed with the Logistic Regression where in conjunction with the BoW it had outperformed its counterpart with the TF-IDF feature extraction method. Surprisingly, the baseline models have an outstanding performance when it comes to accuracy which is already hard to topple. Figure 2 and 3 show the Confusion Matrix for the NB and LR with BoW See the Appendix A for more detailed figures.

Naïve Bayes is particularly well-suited for spam classification because it is a probabilistic model that performs well with text data, handling the large dimensionality of feature space efficiently and making strong independence assumptions that often hold true in practice. Logistic Regression is advantageous due to its simplicity, interpretability, and ability to model the relationship between features and the probability of the target class.

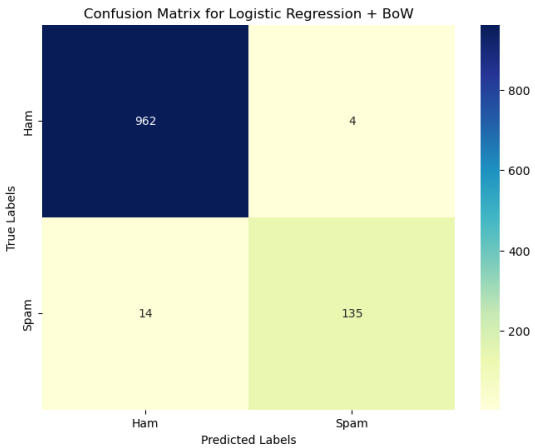



*Figure 2*

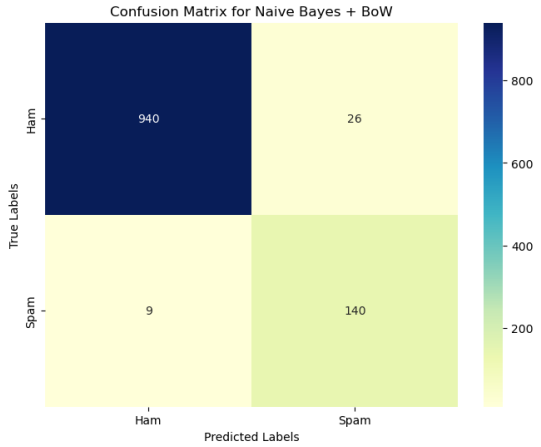



*Figure 3*

## 8 Results, error analysis

### The SVM model

Firstly, we start off by showing the results for fine tuned SVM model coupled with Word2Vec and TF-IDF vectorization techniques. The incorporation of TF-IDF and Word2Vec for text classification uses the benefits of both techniques, which really enhance the model performance. TF-IDF is a method that tries to find relevant phrases based on their frequency and distinctiveness in different emails, hence looking at important words. Word2Vec extends this facility to capture complex semantic connections and contextual nuances of words through its neural network-based embeddings. The combination results in a comprehensive feature space capable of detecting important terms and a sophisticated comprehension of language usage. We have tried different version of the word2vec in order to find the optimal and use the optimal through the manual search. The evaluation metrics for this model are the following: the accuracy of 92%, the recall of 92%, and the F1 Score of 90%. This performance is worse than that of the baseline models across all of the metrics. It would be expected that with fine-tuning the SVM model would outperform the baseline models.

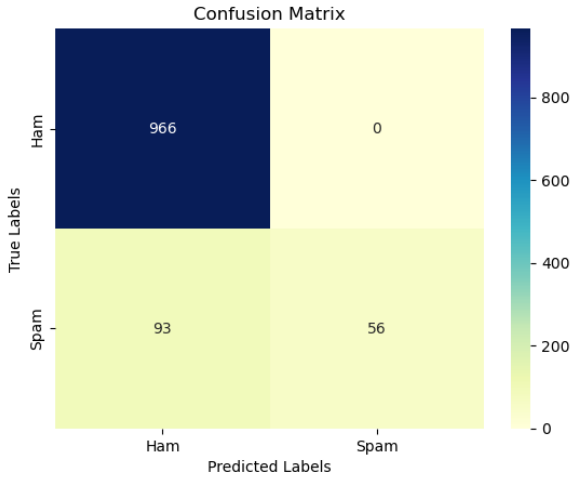



*Figure 4*

Figure 4 indicates that all of the ham emails have been correctly identified, however, there is a significant misclassification when it comes to the identification of spam messages. The performance over the ham message identification considerably skews the accuracy of the model. Since the model is made to classify spam messages it severely lacks.

On the other hand, if we use the SVM model with TF-IDF or BoW as proposed for the baseline models. It significantly improves across all of the evaluation metrics. The best performance is seen in SVM coupled with BoW model where the accuracy is 98.2%, the precision is 97.8%, the recall is 88.6%, and the F1 Score is 93.0%. The performance in accuracy and precision slightly outperforms the baseline models but is worse in recall and F1 Score.

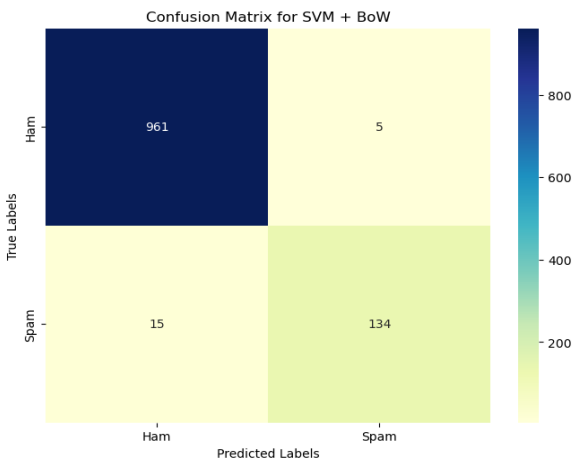



*Figure 5*

In Figure 5 we can see that this accuracy is not biased and does indeed does the intended purpose of detecting spam emails. The misclassification of spam emails is 15 out of 139 and 5 out of 966 ham emails.

The spam messages are the following:

- A link to your picture has been sent. You can also use http://alto18.co.uk/wave/wave.asp?o=44345
- Bloomberg -Message center +447797706009 Why wait? Apply for your future http://careers. bloomberg.com
- Sorry I missed your call let's talk when you have the time. I'm on 07090201529
- Email AlertFrom: Jeri StewartSize: 2KBSubject: Low-cost prescripiton drvgsTo listen to email call 123
- Hello darling how are you today? I would love to have a chat, why dont you tell me what you look like and what you are in to sexy?

These might have been misclassified due to different reasons. Firstly, some professional credibility has been used that could have indicated that the email was legitimate. Secondly, the tones used in these emails are more informal and casual that might have confused the model.

**The DistilBERT model**

Our second model used for spam classification is DistilBERT. We used upsampling for this model as there is a big class imbalance in the data and tokenize the data. The DistilBERT tokenizer converts text into token IDs that the model can process. This pretrained tokenization, combined with padding and truncation, ensures uniform input size, which is crucial for batch processing and effective learning. We utilized the pytorch for creating a dataframe suitable for our model from the HuggingFace, in order to efficiently utilize the model. We experimented with the model configuration and have come up with the best model possible. The Trainer class from Hugging Face simplifies training and evaluation by handling optimization, logging, and evaluation internally. This enables rapid experimentation and fine-tuning of models. Moreover, we implemented the L2 regularization to prevent any overfitting which would help in generalizing for the unseen data. This model has performed exceptionally as its accuracy

is 99.3%, the precision is 97.8%, the recall is 97.3% and the F1 Score is 97.6%. This is higher than any baseline model used before. Even with the high performance for the baseline models, DistilBERT outperformed them. Surprisingly, the highest accuracy is seen with 1 epoch meaning that our dataset is small enough that 1 epoch is enough to provide the highest accuracy. This is one of the limitations of the dataset mentioned which is evident here.

| Epoch | Training Loss | Validation Loss | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| 1 | 0.021800 | 0.042090 | 0.993722 | 0.979730 | 0.973154 | 0.976431 |
| 2 | 0.000300 | 0.063123 | 0.989238 | 0.947712 | 0.973154 | 0.960265 |
| 3 | 0.041000 | 0.073230 | 0.991031 | 0.966443 | 0.966443 | 0.966443 |
| 4 | 0.000000 | 0.076910 | 0.991031 | 0.966443 | 0.966443 | 0.966443 |
| 5 | 0.000000 | 0.078351 | 0.991031 | 0.966443 | 0.966443 | 0.966443 |

*Figure 6*

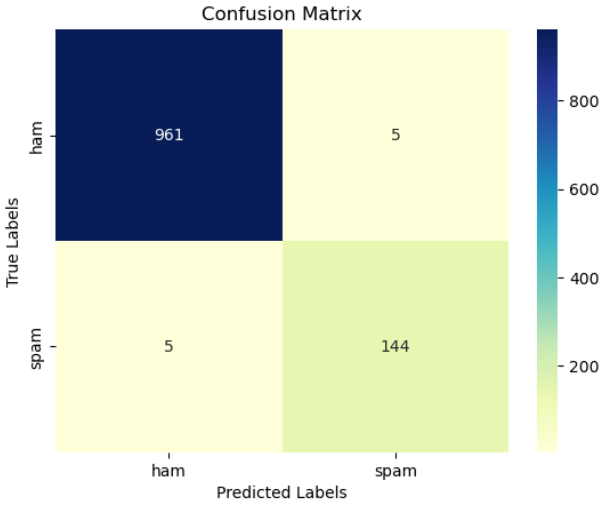



*Figure 7*

From Figure 7, we can see that the DistilBERT only incorrectly identified the 10 emails of 1115, 5 of them being from the spam and another 5 from the ham section. We will now analyze why this might be the case.

The Spam messages:

- Babe: U want me dont u baby! Im nasty and have a thing 4 filthyguys. Fancy a rude time with a sexy bitch. How about we go slo n hard! Txt XXX SLO(4msgs)
- Hello darling how are you today? I would love to have a chat, why dont you tell me what you look like and what you are in to sexy?
- Do you realize that in about 40 years, we'll have thousands of old ladies running around with tattoos?
- Will u meet ur dream partner soon? Is ur career off 2 a flyng start? 2 find out free, txt HORO followed by ur star sign, e. g. HORO ARIES
- Burger King - Wanna play footy at a top stadium? Get 2 Burger King before 1st Sept and go Large or Super with Coca-Cola and walk out a winner

And the ham messages:

- 645
- We are pleased to inform that your application for Airtel Broadband is processed successfully. Your installation will happen within 3 days.
- Is toshiba portege m100 gd?
- MY NO. IN LUTON 0125698789 RING ME IF UR AROUND! H*
- staff.science.nus.edu.sg/~phyhcmk/teaching/pc1323

When it comes to the spam messages, there are a few overlaps that we can see from the examples. Firstly, there is a question mark in between the sentences that are used in these texts. This could have gone under the radar due to the DistibertTokenizer retaining the punctuation marks within the sentences due to the fact that it influences the meaning of the sentences which are important for DistilBERT. Secondly, these messages are written in a subtle informal way which does not give away the commercial subplot of the email which is not detected. This kind of context is missed out by the model as they are very carefully disguised as if a friend is writing an email. The DistilBERT model, like other machine learning models, relies heavily on the context provided during training. The training set might have lacked similar examples or contexts, the model might struggle to accurately classify such texts. DistilBERT relies on explicit cues such as spammy phrases, excessive punctuation, or unnatural links. These are absent or well-disguised within text that appears normal or conversational, the model may fail to correctly classify the message as spam.

When it comes to the ham messages, there are also a few notable patterns that can be easily distinguished. Two of the messages contain numbers and URLs which can be easily put under spam as their no context to the message or any

previous conversations which could have led to the message of only numbers being sent. Another 2 messages sounds like a commercial spam email which would offer some kind of service which is usually a sign of spam message. It is a formal commercial message which might have not been present in the data.

It is also important to note, that the duplicates have not been assigned as spam even though some of the messages are repeating.

## 9 Lessons learned and conclusions

Throughout this project, we learned how effectively the DistilBERT model worked for spam email detection and outperformed older models such as the Naive Bayes and SVM. Key insights included the importance of preprocessing steps like cleaning and tokenization and handling class imbalances. Despite upsampling, the challenge of class imbalance remained a main concern, putting models in danger of overfitting. Hybrid feature extraction using TF-IDF and Word2Vec embeddings achieved superior performance for classical models but was, however, still far from the complexities of detection by transformer models.

The project pointed out that transformer models are hungry for computation, and proper management of resources is important. In-depth error analysis came in handy as it shed light on the model's weaknesses, indicating certain patterns and contexts that the models poorly detected.

We achieved our main aim of developing the best models for spam detection, and DistilBERT showed the best performance with respect to accuracy, precision, recall, and F1-score. However, due to time constraints, further exploration and fine-tuning were impossible.

For further work, I would find a larger, more up to date and more balanced dataset which would more efficiently implement DistilBERT. Moreover, I would utilize RNN-LTSM model in order to get more insights into how this model could detect spam emails. It is increasingly important to find new effective ways to filter out spam as it could negatively affect both individuals and businesses through flooding and phishing emails.

## References


AbdulNabi, I., & Yaseen, Q. (2021). Spam email detection using Deep Learning Techniques. Procedia Computer Science, 184, 853–858. doi:10.1016/j.procs.2021.03.107

Dada, E. G., Bassi, J. S., Chiroma, H., Abdulhamid, S. M., Adetunmbi, A. O., & Ajibuwa, O. E. (2019). Machine learning for email spam filtering: Review, approaches and open research problems. Heliyon, 5(6). doi:10.1016/j.heliyon.2019.e01802

Dedeturk, B. K., & Akay, B. (2020). Spam filtering using a logistic regression model trained by an artificial bee colony algorithm. Applied Soft Computing, 91, 106229. doi:10.1016/j.asoc.2020.106229

Distilbert. (n.d.). Retrieved from https://huggingface.co/docs/transformers/en/model_doc/distilbert

Ellis, C. (2023). Spam statistics 2024: Survey on junk email, AI Scams & Phishing. Retrieved from https://www.emailtooltester.com/en/blog/spam-statistics/#:~:text=162%20billion%20spam%20emails%20are,spam%20messages%20in%20some%20form.

Jain, G., Sharma, M., & Agarwal, B. (2019). SPAM detection in social media using convolutional and long short term memory neural network. Annals of Mathematics and Artificial Intelligence, 85(1), 21–44. doi:10.1007/s10472-018-9612-z

Kontsewaya, Y., Antonov, E., & Artamonov, A. (2021). Evaluating the effectiveness of machine learning methods for spam detection. Procedia Computer Science, 190, 479–486. doi:10.1016/j.procs.2021.06.056

Kotni, S., Potala, C., & Sahoo, L. (2022). SPAM DETECTION USING DEEP LEARNING MODELS. International Journal of Advanced Research in Engineering and Technology (IJARET), 13(5), 55–64.

Makkar, A., & Kumar, N. (2020). An efficient deep learning-based scheme for web spam detection in IOT environment. Future Generation Computer Systems, 108, 467–487. doi:10.1016/j.future.2020.03.004

Nagpal, M. (2024). Stemming in NLP- A beginner's guide to NLP mastery. Retrieved from https://www.projectpro.io/article/stemming-in-nlp/780#:~:text=Snowball%20stemmer%20has%20slightly%20better,languages%20to%20produce%20accurate%20results.

Tida, V. S., & Hsu, S. H. (2022). Universal spam detection using transfer learning of Bert Model. Proceedings of the Annual Hawaii International

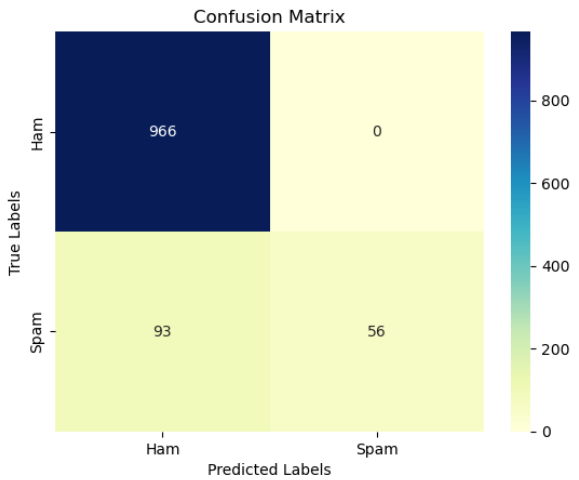
Conference on System Sciences. doi:10.24251/hicss.2022.921

Wu, T., Liu, S., Zhang, J., & Xiang, Y. (2017). Twitter spam detection based on Deep Learning. Proceedings of the Australasian Computer Science Week Multiconference. doi:10.1145/3014812.3014815

Xu, G., Zhou, D., & Liu, J. (2021). Social network spam detection based on Albert and combination of Bi-LSTM with self-attention. Security and Communication Networks, 2021, 1–11. doi:10.1155/2021/5567991

# A Appendices

Confusion Matrix for 300 vector size Word2Vec:




Evaluation Metrics for Baseline models and SVM model:

| | Model | Features | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|---|---|
| 0 | Logistic Regression | BoW | 0.983857 | 0.971223 | 0.906040 | 0.937500 |
| 1 | Naive Bayes | BoW | 0.968610 | 0.843373 | 0.939597 | 0.888889 |
| 2 | SVM | BoW | 0.982063 | 0.964029 | 0.899329 | 0.930556 |
| 3 | Logistic Regression | TF-IDF | 0.977578 | 0.924658 | 0.906040 | 0.915254 |
| 4 | Naive Bayes | TF-IDF | 0.969507 | 0.840237 | 0.953020 | 0.893082 |
| 5 | SVM | TF-IDF | 0.981166 | 0.977612 | 0.879195 | 0.925795 |

Confusion Matrices for NB, LR, and SVM +TF-IDF: