

# Spam Detection Model

**COMP 333: Data Analytics**

Instructor: Dr. Yaser Esmaeili Salehani

**Submitted by:**

Matteo Robidoux  
Student ID: 40282589

Raagav Prasanna  
Student ID: 40282749

**Date:**

April 7th, 2025

## Abstract

This study looks into the problem of finding spam in different types of input, such as SMS, email, and YouTube comments. Traditional spam detection commonly solely apply models such as Random Forest, or Naive Bayes. However, our study indicates that the identification of URLs embedded within messages plays a **critical role** in determining spam likelihood. Therefore, in addition to conventional text-based training approaches, we introduced a separate URL-based model to enhance prediction accuracy.

Furthermore, we observed **key differences across input datasets**. SMS messages consist solely of a body containing text, whereas emails include a subject line. Moreover, YouTube comments also often contained spam related to self-promotion, with messages similar to “Please subscribe to my channel”. Additionally, emails and comments provided unique attributes, such as email subjects and the authors of comments, which are also key pieces of data required to identify spam.

To address these variations, **we developed three specialized models for SMS, email, and comment spam detection**, each catering to the different datasets that we acquired. The individual model’s predictions were then combined with the output of the URL spam model whenever a URL was present. This hybrid approach **significantly improved classification accuracy**, significantly outperforming or offering similar results to traditional methods. This was especially shown in the SMS model, which **demonstrated an increase in accuracy of over 35%** compared to traditional methods. This study demonstrates the importance of URL detection in spam classification, and shows that a hybrid approach can effectively address the limitations of traditional spam detection methods. Finally, it also highlights the importance of dataset balancing through upsampling or downsampling, comparing our improved results to other studies that used the same datasets, but without any balancing.

## 1 Introduction

The rise of digital communication has led to an increase in the amount of unwanted and potentially harmful spam messages across various platforms. This includes SMS messages, emails, and content within online comment sections. Spam messages can range anywhere from harmless advertisements, to phishing attempts and even malware distribution. This emphasizes the need for effective spam detection, which is needed for ensuring the security of a user, as well as the integrity of a platform. The traditional spam detection techniques, **commonly known as “ham or spam”**, often rely solely on text based models, using an algorithm such as SVM, Random Forest, Decision Tree, KNN, or even Naive Bayes. While these methods are effective when used to identify many different spam messages given the right input dataset, they often fail to account for additional information catering to the input, leading to both false negatives as well as false positives. In short, on many occasions, messages that are not spam are marked as spam, and messages that are, are not.

One key aspect of spam identification that is not present within traditional methods is the identification of URLs. Many spam messages contain malicious links, requiring models to take these links into account. Furthermore, a message can also contain a valid url sent from a valid user, which means that messages containing urls should not be marked as spam simply because they contain a url. Instead, the url itself should be analyzed to determine whether or not it is spam. **Our study demonstrates that having a hybrid prediction approach**, having one model predict the spam likelihood of a url, and having another model predict the spam likelihood of the text body, **significantly improves the ability to detect spam messages** when there is a url present in the body.

Moreover, spam characteristics **differ across various platforms**. SMS messages primarily consist of just text, whereas emails include components like subject lines. YouTube comments, on the other hand, often involve engagement-driven spam, such as self-promotion; as well as peculiar author names that could indicate spam. These differences require the need to develop specific models for each input type, as opposed to just one model.

To address this, we propose a hybrid spam detection algorithm consisting of three specialized models for SMS, email, and YouTube comments. Each model is designed to uniquely identify characteristics of its respective dataset, while using a separate model for the embedded URLs in the event they exist. When a message contains a URL, the URL model’s prediction is combined with that of the text

model’s, determining the legitimacy of the message. **Our results demonstrate** that this approach significantly enhances detection accuracy, **outperforming traditional methods**.

The rest of this paper is organized as follows: Section 2 reviews **related work** in spam detection. Section 3 describes our **methodology**, including dataset preprocessing, model architectures, and **training procedures**. Section 4 discusses the Inference procedure. Finally, Section 5 discusses our **key findings**.

## 2 Related Work

A notable study by Gupta et al. follows the conventional ham or spam detection algorithm using TF-IDF vectorization [1]. Their research applies machine learning models to the UCI SMS Spam Collection dataset [2], using text-based features extracted from SMS messages. Their results show strong performance in spam classification, using traditional approaches such as Naïve Bayes and SVM, emphasizing that this is an effective technique.

Another study by Gawai and Salunke [8] compared the use of various different NLP and machine learning techniques to classify SMS messages as either spam or ham, specifically for the algorithms **SVM, Decision Tree, Random Forest, KNN and Naive Bayes**. Their study demonstrates the effectiveness of these approaches in improving spam detection accuracy, concluding that the Naive Bayes had the most successful results, achieving an accuracy of **98%** on the UCI SMS Spam Collection dataset. They also report quite good accuracy results for the rest of the models, with **97%** for SVM, **96%** for Decision Tree, **97%** for Random Forest, and **95%** for KNN.

However, while training models on the **same dataset** [2], we identified a **significant limitation** in the way spam messages were labeled, especially when a URL was present in the message body. The dataset shows a **strong bias**, as many SMS messages containing URLs are labeled as spam. This introduces a risk that models trained on this data may incorrectly generalize, leading to a high false positive rate for messages containing legitimate URLs, significantly impacting simple sms messages such as a getting sent a video link from a friend. Consequently, SMS messages may be disproportionately influenced by the presence of a URL rather than the actual content of the message.

Our solution resolves this issue by introducing a separate URL-based classification model, which is then used in conjunction with either the SMS message model, Email model, or YouTube comments model, to predict whether they are spam. By independently determining the likelihood of a URL being spam, this hybrid approach reduces dataset bias and ensures more balanced spam classification, particularly for messages containing URLs.

## 3 Methodology

### 3.1 Datasets and Preprocessing

For this study, in order to ensure that each of the models have more than enough data to train and eventually perform inference on, we used a wide variety of different datasets. Specifically, we used datasets for URLs, SMS messages, Emails, and YouTube comments.

#### URL Dataset

For the URLs, we decided to have a dedicated dataset containing plenty of addresses that were classified to be either spam or non spam. This way, the model had more than enough data to decide which urls were malicious, preventing common links such as from platforms like YouTube and TikTok from being marked as spam. The kaggle dataset that we used, titled Malicious and Benign URL’s [3], contained over **450,000 url’s**, that were marked as either being 0 for benign, or 1 for malicious.

In terms of preprocessing, we modified the names of the result and url columns to is\_spam and text respectively. Furthermore, we dropped all other columns as these were the only 2 that were needed. Additionally, we balance the dataset by dropping some urls, ensuring that there are equal number of

malicious and benign urls. This is done using random samples. Finally, we normalized the text column by removing trailing whitespaces, as well as non unicode characters identified by  $\backslash$ . In the end, we end up with **208,867 rows**.

Table 1: URL Dataset post Preprocessing

	text	is_spam
0	https://www.google.com	0
1	https://www.youtube.com	0
2	https://www.facebook.com	0
	$\vdots$	
1975	http://mytorsmired.ru/gate.php	1
1976	http://narbit.com/rss/feed/stream/	1
1977	http://narbit.com/rss/feed/stream	1

## SMS Datasets

For SMS messages, we utilized 2 datasets being the UCI SMS Spam Collection Dataset [2], containing a total of **5575 messages**, as well as another kaggle dataset named SMS SPAM DATASET [4], containing **10286 messages**. Both datasets contained the headers v1 and v2, where v1 categorized the row as either ham or spam, and v2 contained the text of the SMS message.

Table 2: UCI SMS Spam Collection Dataset

	v1	v2
0	ham	Go until jurong point, crazy only in ...
1	ham	Ok lar... Joking wif u oni
2	spam	Free entry in 2 a weekly comp to win FA Cup ...
		$\vdots$
5574	spam	WINNER!! As a valued network customer you have ...
5575	spam	Had your mobile 11 months or more? Then ...

Table 3: SMS SPAM DATASET

	v1	v2
0	spam	Congratulations! You’ve been selected for a luxury vacation getaway...
1	spam	URGENT: Your account has been compromised. Click here to reset your password immediately...
2	spam	You’ve won a free iPhone! Claim your prize by clicking on this link now...
		$\vdots$
9526	ham	Okie...
9527	ham	”Aight, I’m chillin in a friend’s room so text me when you’re on the way”

When preprocessing, we merged the two datasets into a single one, and then renamed the columns to text and is\_spam. Additonally, we also decide to remove the url’s that were present in the messages, as we would be using the URL dataset to classify them. This was done by using a regex pattern to identify and remove any url’s that were present in the text. Finally, we also decide to balance the dataset by ensuring that there were an equal number of spam and ham messages. This was done by removing a random sample of of the messages, until there were equal amounts of spam and ham messages. This was done to ensure that the model would not be biased towards one class or the other, as well as to ensure that the model would be able to generalize well to new data. In the end, we ended up with a total of **6306 rows**.

## Email Dataset

When extracting the email data, we went for a total of 2 datasets. The first kaggle dataset titled Email Spam Dataset [5], contained 3 data subsets. The second dataset was titled Spam Mails Dataset [6]. In total, these 4 csv files provided us with over **23000 rows of data**.

For preprocessing most of the methods stayed exactly the same as prior. However now we had the added challenge of needing to extract the subject from the email. This was done through simple regex matching and adding a subject column to our normalized dataset. In the end, we end up with **17,656** rows of data.

## YouTube Comments Dataset

For comments we used a single kaggle dataset titled YouTube Comments Spam Dataset. This dataset contained the following headers: COMMENT\_ID,AUTHOR,DATE,CONTENT,VIDEO\_NAME,CLASS, and contained 1962 rows of data.

When preprocessing, all previous methods were also applied. Furthermore, we decided to also add the author column in the normalized dataset, as we figured that it could potentially be useful in predicting spam. In the end, we ended up with **1902 rows** of data.

## 3.2 Model Features and Training

In this section, we discuss the specific architecture and training process of each of the four models used in our spam detection system: the URL model, SMS model, Email model, and YouTube Comments model. Each model is designed to capture specific characteristics of its respective data type while integrating with the URL classifier to improve overall accuracy.

### 3.2.1 URL Model

#### Feature Extraction

We extract the following features from each URL:

- **Structural features**
  - Length of the URL, number of dots, number of query parameters, etc.
- **Specific keywords**
  - Presence of words like “reward,” “win,” “gift,” “claim,” for spam detection.
- **Domain and subdomain features**
  - Length of the domain name, presence of suspicious top-level domains (TLDs) such as .xyz or .biz
- **Redirect and path features**
  - Presence of redirects, subdomain keywords like “auth” or “login,” and path length.

#### Training

We trained an **LightGBM model** (lighter alternative to XGBoost, to minimize training time), with 300 boosts per round. Additionally, we also utilized **Stratified K-fold**, using **5 folds** in total. The model was trained on 208876 rows of data. The following hyperparameters were used:

- Objective: binary
- Metric: Binary Logloss
- Boosting type: gbdt
- 31 leaves

- Learning rate of 0.1
- Feature fraction of 0.7
- Bagging fraction of 0.7
- Bagging frequency of 5

The model achieved the following results after training on a Lenovo Flex 5 Laptop with a AMD Ryzen 7 4700U Processor and 16GB of DDR4 memory, taking around **187 secs**:

- Average Accuracy: **98.55%**
- Average AUC: **99.82%**
- Average Precision: **99.85%**

Table 4: URL Model Classification Report

	Precision	Recall	F1-Score	Support
<b>Ham</b>	<b>0.98</b>	<b>0.99</b>	<b>0.99</b>	104438
<b>Spam</b>	<b>0.99</b>	<b>0.98</b>	<b>0.99</b>	104438
<b>Accuracy</b>			<b>0.99</b>	208876
Macro Avg	0.99	0.99	0.99	208876
Weighted Avg	0.99	0.99	0.99	208876

Additionally, the following ROC curve, Precision-Recall curve, and confusion matrix were generated to visualize the model's performance:

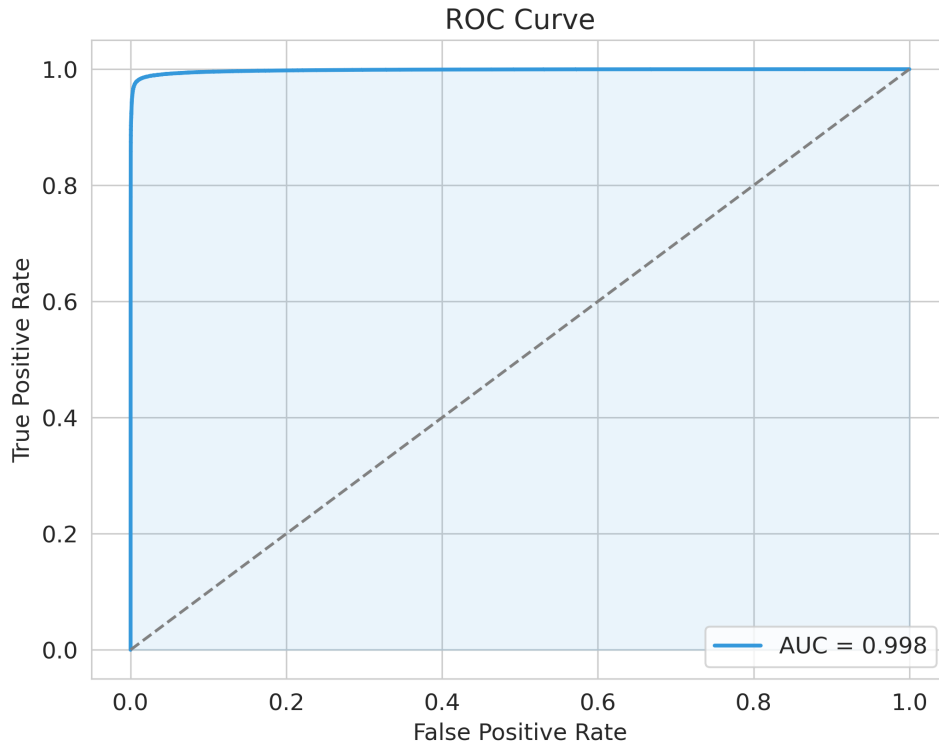


Figure 1: ROC Curve for URL Model

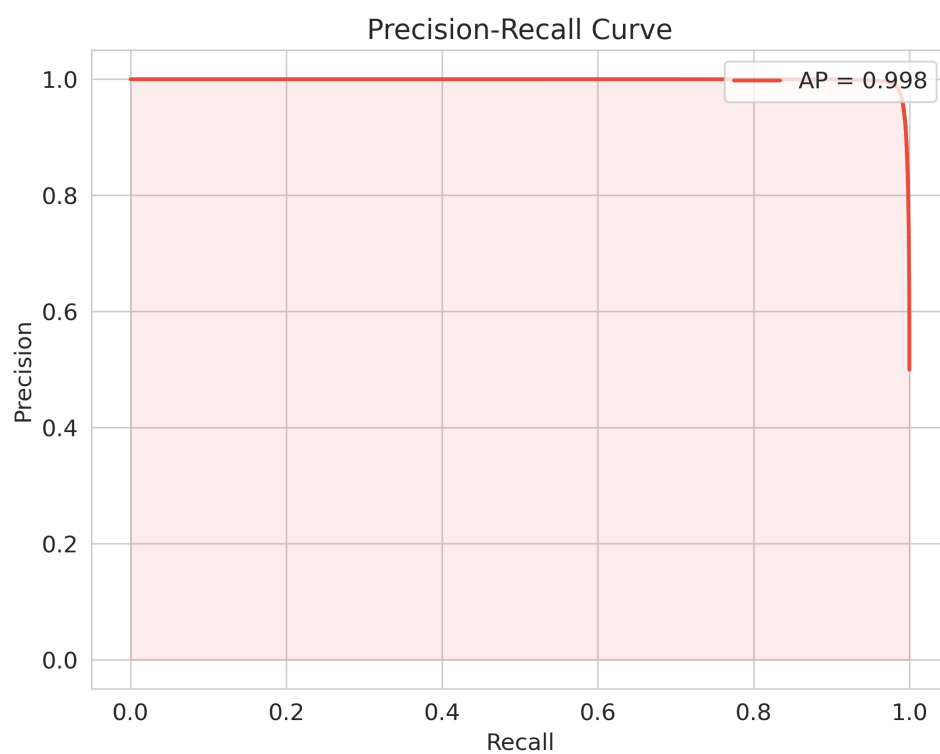


Figure 2: Precision-Recall Curve for URL Model

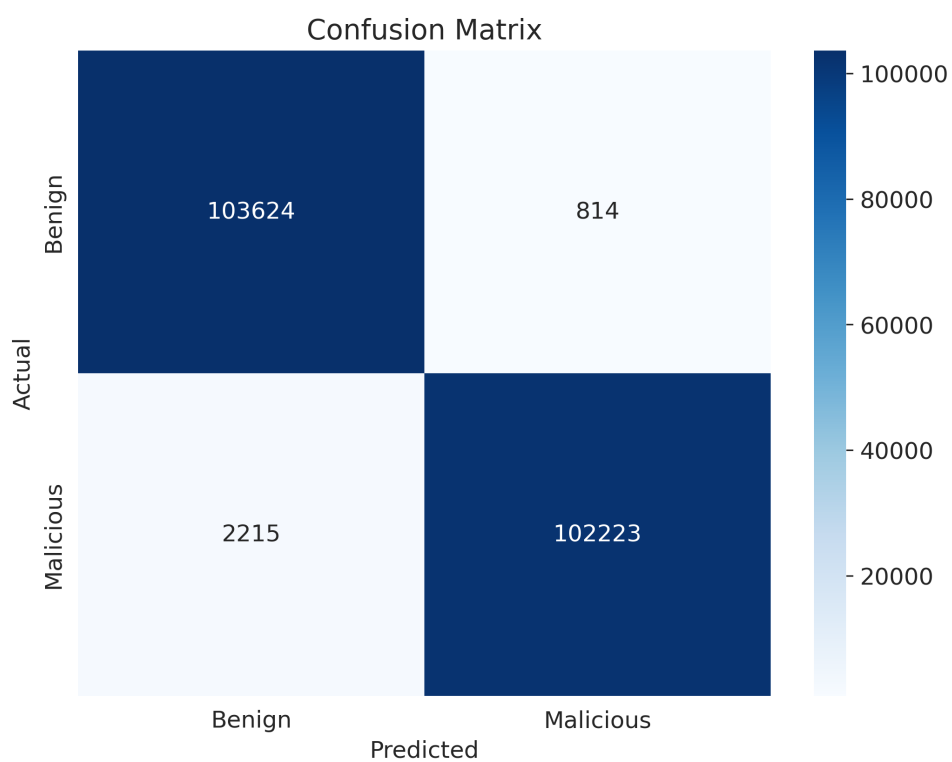


Figure 3: Confusion Matrix for URL Model

### 3.2.2 SMS Model

#### Feature Extraction

We applied TF-IDF vectorization to capture word patterns and important terms while filtering out common English stopwords. The vectorization settings included:

- Maximum features: 5000
- Minimum document frequency: 2
- Maximum document frequency: 0.95

Training was done on the same laptop using **Stratified K-fold with 10 folds**. However, this time a **Random Forest Classifier** was used with the following hyperparameters:

- 200 estimators
- Max depth of 30

The model achieved the following results after training 6306 rows, taking around **54 secs**:

- Accuracy: **96.73%**
- AUC: **99.16%**
- Average Precision: **99.34%**

Table 5: **SMS model Classification Report**

	Precision	Recall	F1-Score	Support
<b>Ham</b>	<b>0.96</b>	<b>0.97</b>	<b>0.97</b>	3153
<b>Spam</b>	<b>0.97</b>	<b>0.96</b>	<b>0.97</b>	3153
<b>Accuracy</b>			<b>0.97</b>	6306
Macro Avg	0.97	0.97	0.97	6306
Weighted Avg	0.97	0.97	0.97	6306

Additionally, the following ROC curve, Precision-Recall curve, and confusion matrix were generated to visualize the model's performance:



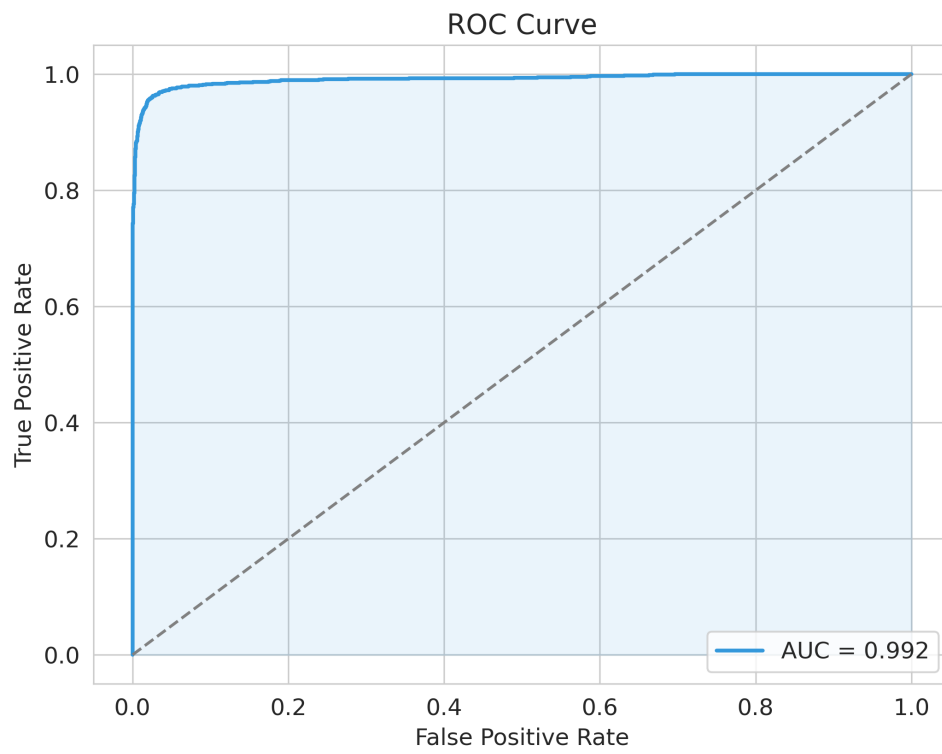


Figure 4: ROC Curve for SMS Model

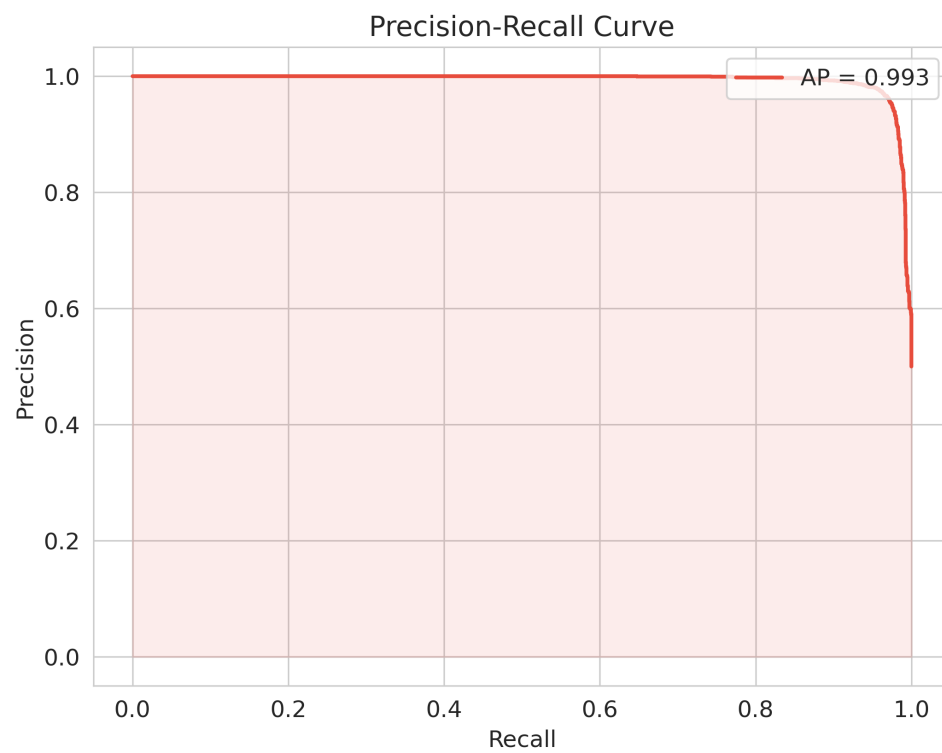


Figure 5: Precision-Recall Curve for SMS Model

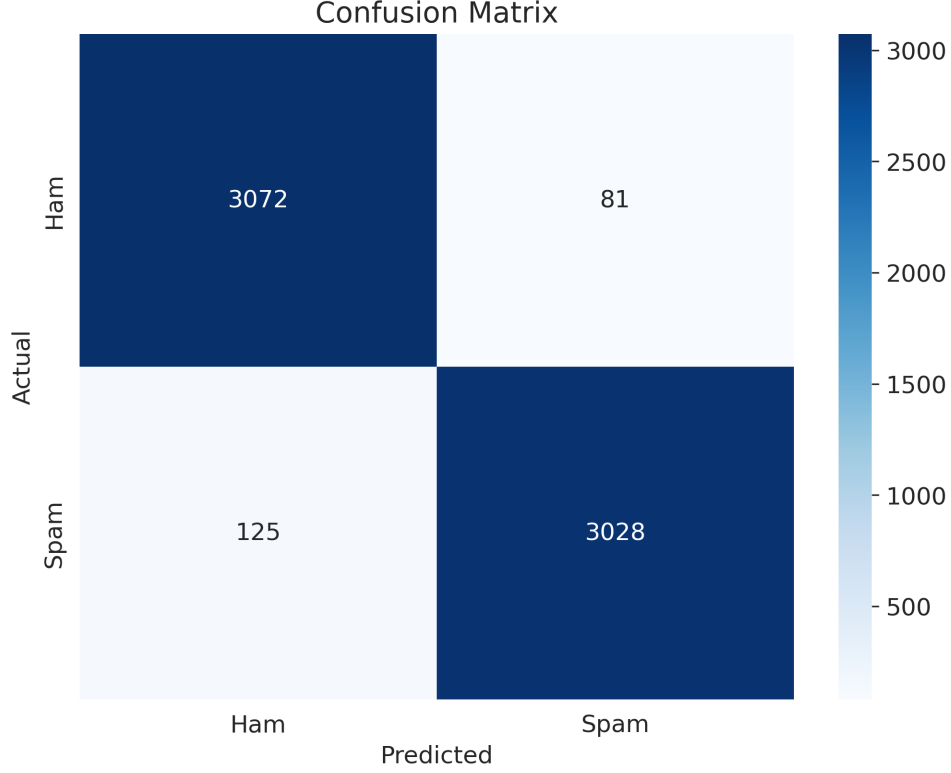


Figure 6: Confusion Matrix for SMS Model

Furthermore, when training the model, we also decided to use another approach where we only train using the **SMS UCI Spam Collection Dataset** [2], using the same normalization approach (Balancing spam and non spam values, removing whitespaces, urls, etc). This was done to compare our results with the results of the study by Gawai and Salunke [8]. The model achieved the following results after training **1494 rows** (Much smaller than original SMS UCI dataset as there is a heavy bias), taking around **17 secs**:

- Accuracy: **97.19%**
- AUC: **99.34%**
- Average Precision: **99.48%**

	Precision	Recall	F1-Score	Support
<b>Legitimate</b>	<b>0.96</b>	<b>0.99</b>	<b>0.97</b>	747
<b>Spam</b>	<b>0.99</b>	<b>0.96</b>	<b>0.97</b>	747
<b>Accuracy</b>			<b>0.97</b>	1494
Macro Avg	0.97	0.97	0.97	1494
Weighted Avg	0.97	0.97	0.97	1494

We also generated the following ROC curve, Precision-Recall curve, and confusion matrix to visualize the model's performance:

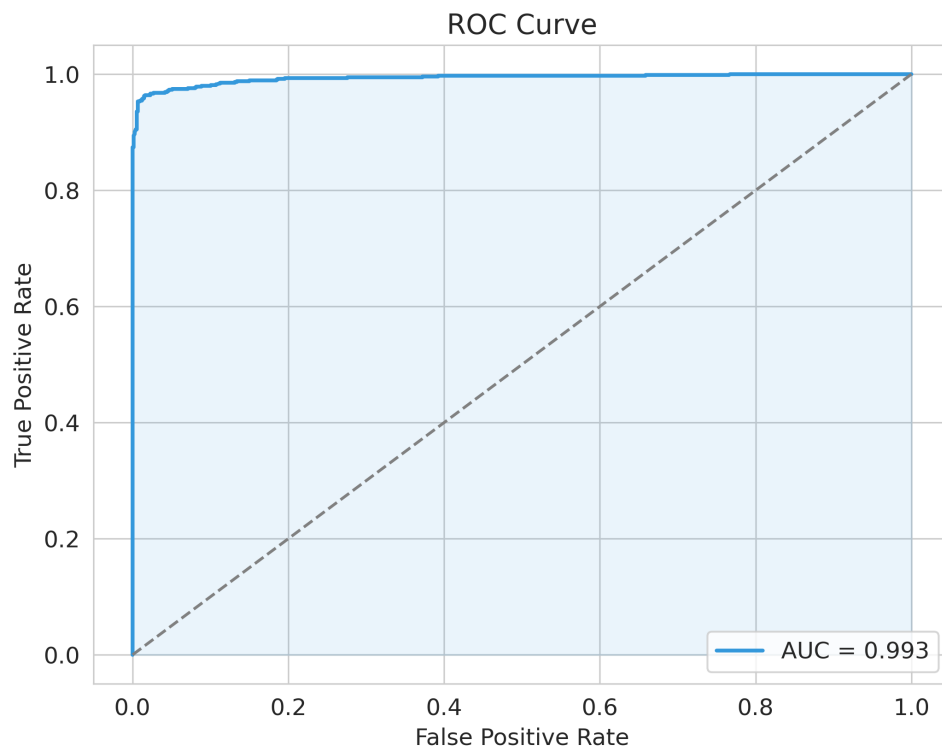


Figure 7: ROC Curve for SMS Model (UCI Dataset)

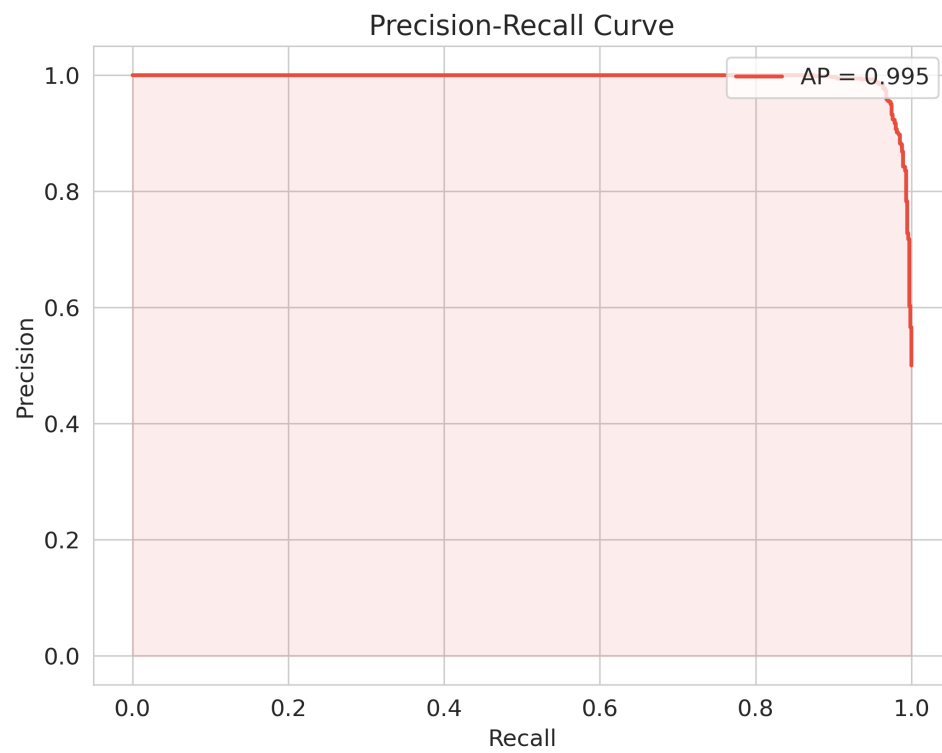


Figure 8: Precision-Recall Curve for SMS Model (UCI Dataset)

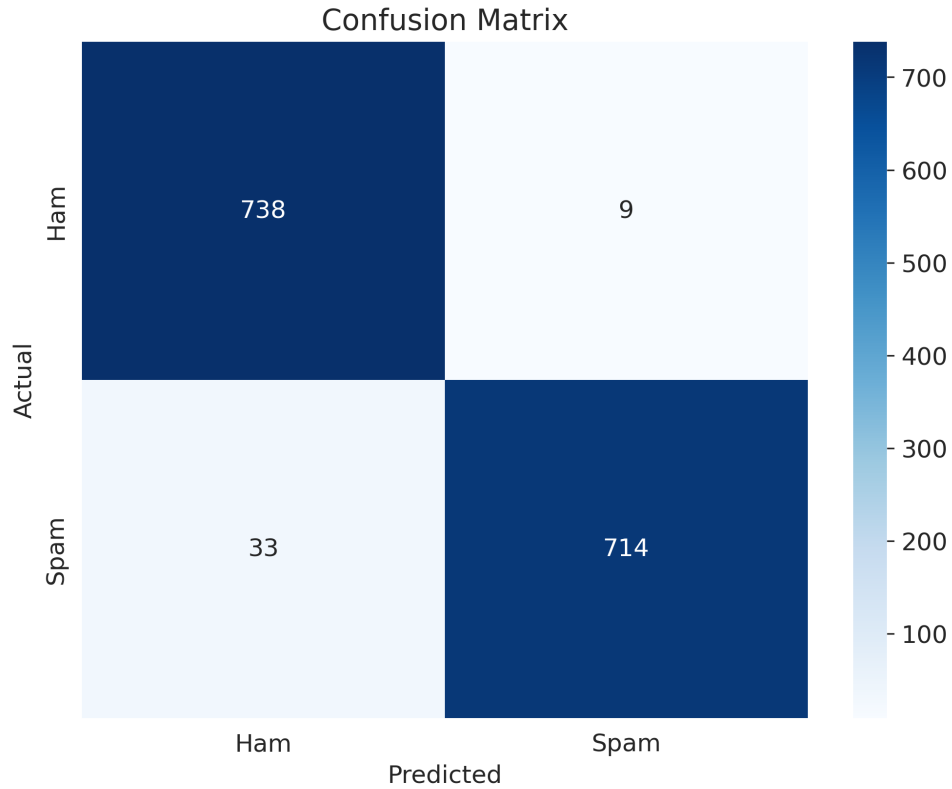


Figure 9: Confusion Matrix for SMS Model (UCI Dataset)

### 3.2.3 Email Model

#### Feature Extraction

We applied TF-IDF vectorization separately to both the email subject and body, allowing the model to leverage textual patterns in each section.

Text vectorization settings:

- Minimum document frequency: 3
- Maximum document frequency: 0.90
- 5000 max features

Subject vectorization settings:

- Minimum document frequency: 2
- Maximum document frequency: 0.95
- 1000 max features

#### Training

We trained an **XGBoost classifier** (No Stratified K-fold), with the same hardware but with a **70/30 train-test split**:

- 500 estimators
- Max depth of 10
- Learning rate of 0.05

- Subsampling of 0.8
- Early stopping at 30 rounds if no improvement

The model achieved the following results after training 23000 rows, taking 249 secs:

- Accuracy: **96.90%**
- AUC: **99.59%**
- Average Precision: **99.55%**

Table 7: Email Model Evaluation Report

	<b>Precision</b>	<b>Recall</b>	<b>F1-Score</b>	<b>Support</b>
<b>Legitimate</b>	<b>0.99</b>	<b>0.95</b>	<b>0.97</b>	2649
<b>Spam</b>	<b>0.95</b>	<b>0.99</b>	<b>0.97</b>	2649
<b>Accuracy</b>			<b>0.97</b>	5297
Macro Avg	0.97	0.97	0.97	5297
Weighted Avg	0.97	0.97	0.97	5297

### 3.2.4 YouTube Comments Model

#### Feature Extraction

Here, we once again utilize a TF-IDF vectorizer for text, as well as one for author.

- Textual Features: TF-IDF vectorization applied to comment text
- Author Based Features: TF-IDF vectorization applied to usernames to detect repeated spam patterns from specific users, as well as recognize patterns in usernames that look suspicious

#### Training

Here, we train using a **Gradient Boosting Classifier** mainly so we can capture hierarchical relationships within the text data, since they are comments. This is all done using the same hardware and with a 70/30 train-test split. The hyperparameters include:

- 200 estimators
- Max depth of 6
- Learning rate of 0.05
- Subsampling of 0.7

The model achieved the following results after training 1962 rows, taking 0.73 secs:

- Accuracy: **94.22%**
- AUC: **98.65%**
- Average Precision: **98.78%**

Table 8: YouTube Comments Model Evaluation Report

	<b>Precision</b>	<b>Recall</b>	<b>F1-Score</b>	Support
<b>Legitimate</b>	<b>0.91</b>	<b>0.98</b>	<b>0.94</b>	286
<b>Spam</b>	<b>0.98</b>	<b>0.91</b>	<b>0.94</b>	285
<b>Accuracy</b>			<b>0.94</b>	571
Macro Avg	0.94	0.94	0.94	571
Weighted Avg	0.94	0.94	0.94	571

## 4 Inference

The inference process across SMS, email, and comments, generally follows the same execution steps. While the specifics vary slightly for each type of message, the overall approach remains the same.

### 4.1 Preprocessing and Feature Extraction

1. URL Cleaning: Any URLs present in the text are extracted and normalized.
2. Vectorization: The cleaned text is transformed using pre-trained vectorizers specific to each message type (SMS, email, comments). Email subjects and comment author names are also vectorized where applicable.
3. URL Feature Extraction: If URLs are found in the text, a feature extraction function analyzes them based on characteristics such as length, presence of suspicious keywords, and domain reputation, as done within the training scripts feature extractor.

### 4.2 Probability Estimation

1. Text Classification: The vectorized text is passed through a trained classification model that returns a spam probability score.
2. URL Spam Detection: If URLs are present, they are sent into the URL model to predict the likelihood of them being malicious or spam-related.
3. Weighted Combination: The probability scores from text classification and URL analysis are combined using a weighted sum approach. The weight distribution varies depending on the message type, with URL spam probability often given higher significance if URLs are detected.

### 4.3 Final Decision

The final spam probability score is then compared against a predefined threshold (typically 0.5). If the score meets or exceeds this threshold, the message is classified as spam; otherwise, it is classified as non-spam.

## 5 Results and Discussion

### 5.1 Comparison to Traditional Methods

As mentioned previously, the study by Gawai and Salunke [8] aimed to compare the performance of various machine learning algorithms for SMS spam detection. The results of their study are shown within the table below:

Table 9: Performance of different algorithms by Gawai and Salunke [8]

Algorithms	Precision		Recall		F1-Score		Accuracy
	Ham	Spam	Ham	Spam	Ham	Spam	
SVM	0.96	0.99	0.99	0.70	0.98	0.82	0.97
Decision Trees	0.98	0.91	0.99	0.81	0.98	0.86	0.96
<b>Random Forest</b>	<b>0.97</b>	<b>0.99</b>	<b>0.99</b>	<b>0.79</b>	<b>0.99</b>	<b>0.88</b>	<b>0.97</b>
KNN	0.94	0.99	0.99	0.54	0.97	0.70	0.95
<b>Naïve Bayes</b>	<b>0.98</b>	<b>0.99</b>	<b>0.99</b>	<b>0.85</b>	<b>0.99</b>	<b>0.92</b>	<b>0.98</b>

When utilizing the same dataset and training algorithms, we get the following results:

Table 10: Performance of different algorithms, Our Results

Algorithms	Precision		Recall		F1-Score		Accuracy
	Ham	Spam	Ham	Spam	Ham	Spam	
SVM	0.96	0.99	0.99	0.96	0.97	0.97	0.97
Decision Trees	0.94	0.94	0.94	0.94	0.94	0.94	0.96
<b>Random Forest</b>	<b>0.96</b>	<b>0.99</b>	<b>0.99</b>	<b>0.96</b>	<b>0.97</b>	<b>0.97</b>	<b>0.97</b>
KNN	0.95	0.95	0.95	0.95	0.95	0.95	0.95
<b>Naïve Bayes</b>	<b>0.97</b>	<b>0.98</b>	<b>0.98</b>	<b>0.97</b>	<b>0.98</b>	<b>0.98</b>	<b>0.98</b>

Additionally, as stated before, they trained their models on the UCI SMS Spam Collection Dataset [2]. When using the same dataset but following our normalization technique by balancing the dataset with equal number of spam and non spam results, we achieved the results shown in table 6. The datasets in the end are quite different in size, as the original UCI dataset contains **5574 rows**, while our dataset only contains **1494 rows** after normalization. However, we can see that our results are quite similar to theirs. For example, for all algorithms, we receive the same accuracy scores. However, on average we achieved much **better results** when it comes to **Spam** detection, specifically when taking a look at the **Recall** and **F1-Score**, where when we look at **Random Forest** for example, **they received 0.79 and 0.88** respectively, while **we received 0.96 and 0.97** respectively. Coupled with the fact that we also trained our model using Stratified K-fold, we can see that our approach yields much better results when it comes to spam detection.

Furthermore, when we compare the results of our **Naive Bayes** to their results using the Naive Bayes algorithm, we see both similar and slightly improved results. For Spam for example, **they received a precision of 0.99 and a recall of 0.85**, while **we received a precision of 0.98 and a recall of 0.97**. We also both have the **same accuracy of 98%**. This shows that our model is able to achieve similar and if not better results compared to traditional methods, while also being able to outperform them in some cases. This highlights the importance of proper data normalization and preprocessing, as well as using multiple training methods to achieve the best results possible; as even with the best training algorithms, if the data is not properly preprocessed, the results will not be as good as they could be.

## 5.2 Analysis of Hybrid Approach

The hybrid approach that we implemented, by training a separate model for URL spam detection, and then combining the results with the text model, also showed very promising results. In order to evaluate its effectiveness, we decided to compare the results of the hybrid approach, to that of the traditional approach. To begin with, we trained the SMS model using a balanced combination of UCI [2] with the SMS Spam Dataset [4]. This model was then used for both the traditional and hybrid approach, with the only difference being that the hybrid approach also used the URL model to classify the URLs present in the messages. We then, using the UCI dataset [2], append malicious and benign URLs to the end of the messages, to individually test each models ability to detect spam.

Table 11: Sample of combined dataset with SMS and URL

is_spam	text
1	You have 1 new voicemail. Please call 08719181513. <a href="http://aslong.googlecode.com/svn/Soft.exe">http://aslong.googlecode.com/svn/Soft.exe</a>
1	You are a winner U have been specially selected 2 receive 1000 cash ... <a href="http://tech-keem.pw/local/">http://tech-keem.pw/local/</a>
0	Ok going to sleep. Hope i can meet her. <a href="https://www.tomgibsoncommunications.com/">https://www.tomgibsoncommunications.com/</a>

For the traditional approach, we use the same model to classify the input messages as Legitimate or Spam. For the hybrid approach, we first classify the URLs using the URL model, and then combine the results with the SMS model to classify the input messages as Legitimate or Spam. If both models classify the input as Legitimate, then the input is classified as Legitimate. If either model classifies the input as Spam, then the input is classified as Spam. This way, we can see how well the hybrid approach performs compared to the traditional approach. Although this is not the most fair comparison, as the hybrid approach utilizes another URL model trained on the URL dataset, it is still a good way to see how well the hybrid approach performs compared to the traditional approach, in terms of real world applications of a detection system. The results post prediction for both approaches, are shown below:

Table 12: Traditional model Approach prediction Results, using balanced UCI SMS Spam Collection Dataset [2]

	Precision	Recall	F1-Score	Support
0	1.00	0.26	0.41	374
1	0.57	1.00	0.73	374
<b>Accuracy</b>			<b>0.63</b>	748
Macro Avg	0.79	0.63	0.57	748
Weighted Avg	0.79	0.63	0.57	748

Table 13: Hybrid Model Approach prediction Results, trained using balanced UCI SMS Spam Collection Dataset [2]

	Precision	Recall	F1-Score	Support
0	1.00	0.99	0.99	374
1	0.99	1.00	0.99	374
<b>Accuracy</b>			<b>0.99</b>	748
Macro Avg	0.99	0.99	0.99	748
Weighted Avg	0.99	0.99	0.99	748



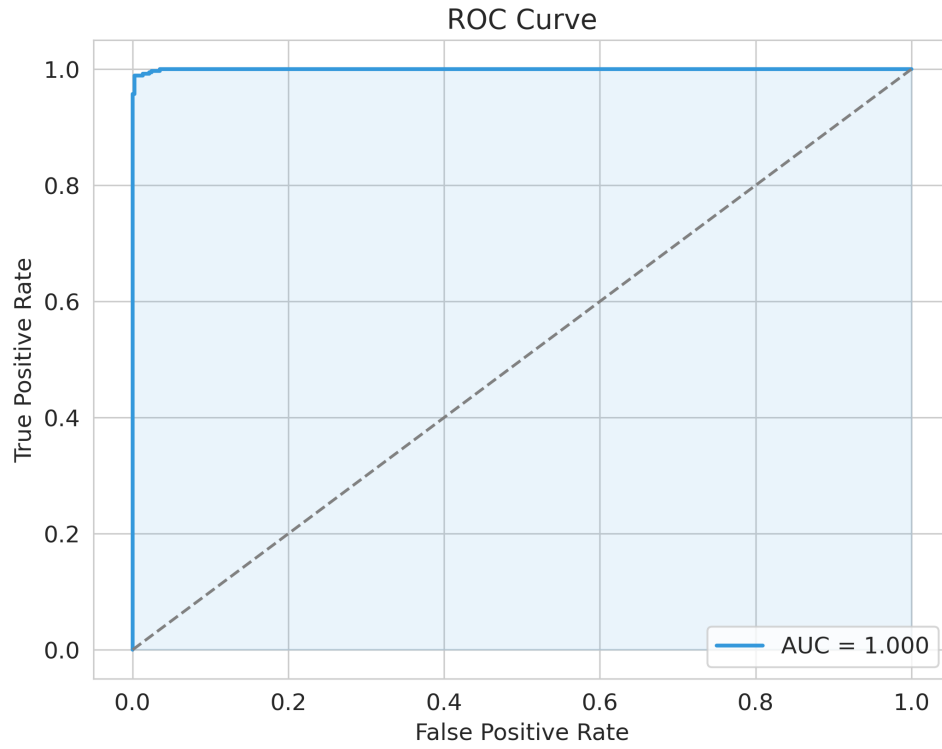


Figure 10: ROC Curve for SMS Traditional Model, trained using both UCI SMS Spam Collection Dataset [2] and SMS Spam Dataset [4]

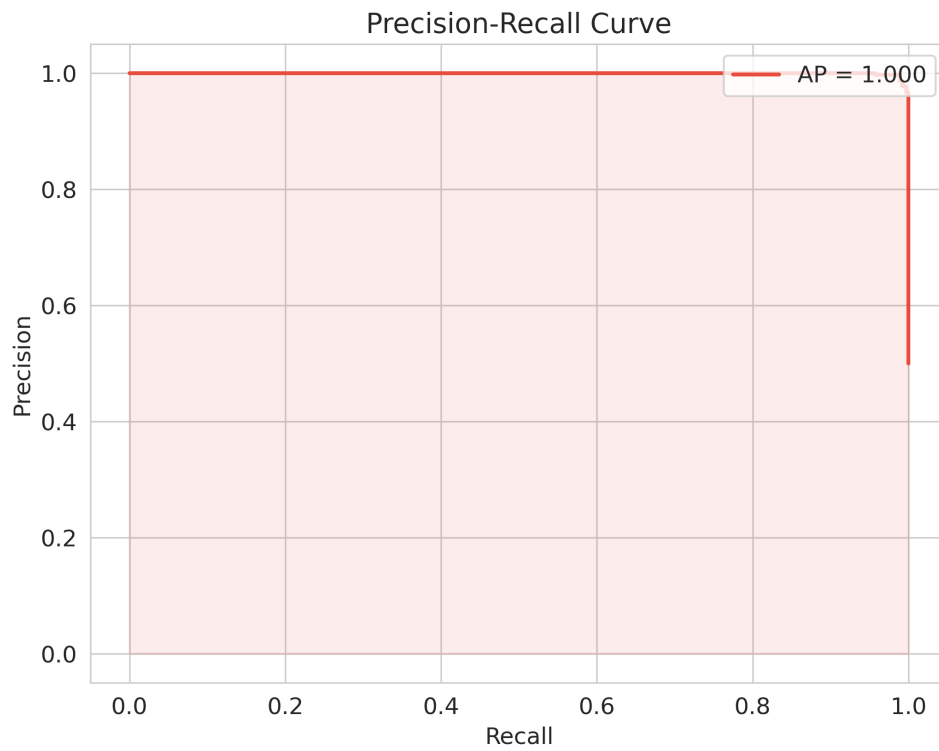


Figure 11: Precision-Recall Curve for SMS Traditional Model, trained using both UCI SMS Spam Collection Dataset [2] and SMS Spam Dataset [4]

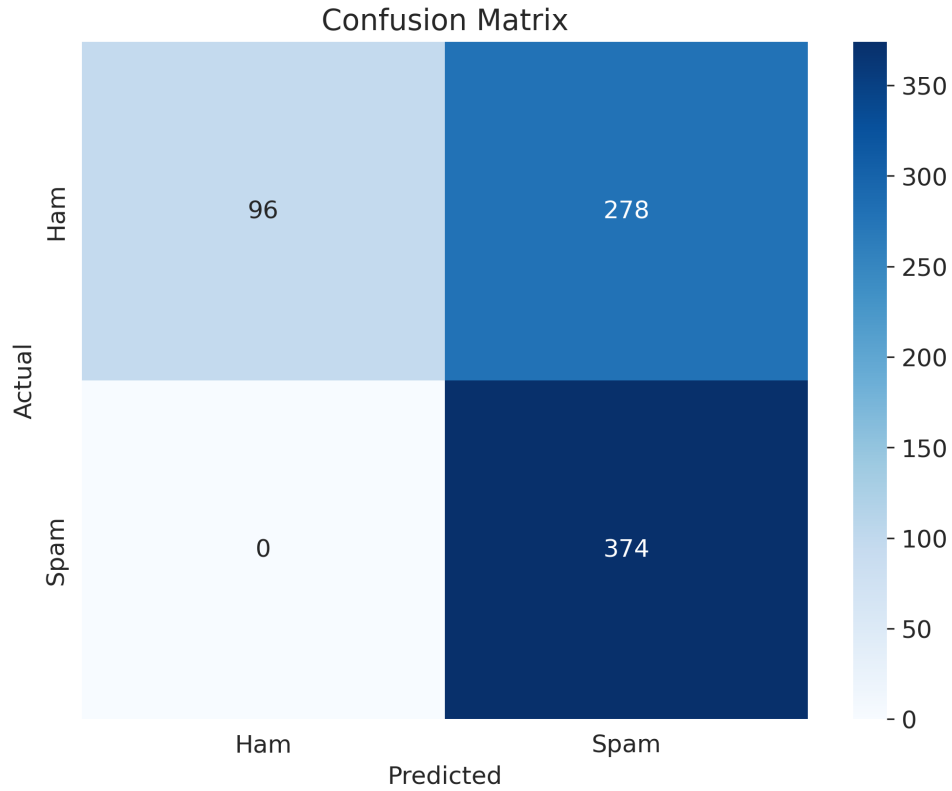


Figure 12: Confusion Matrix for SMS Traditional Model, trained using both UCI SMS Spam Collection Dataset [2] and SMS Spam Dataset [4]

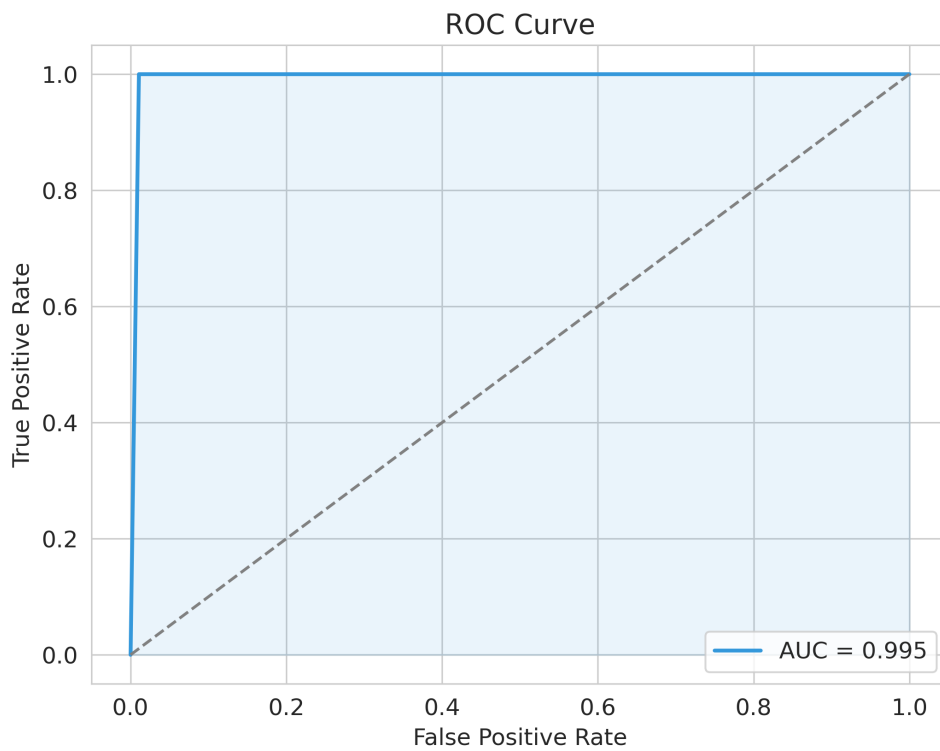


Figure 13: ROC Curve for SMS Hybrid Model, trained using both UCI SMS Spam Collection Dataset [2] and SMS Spam Dataset [4]

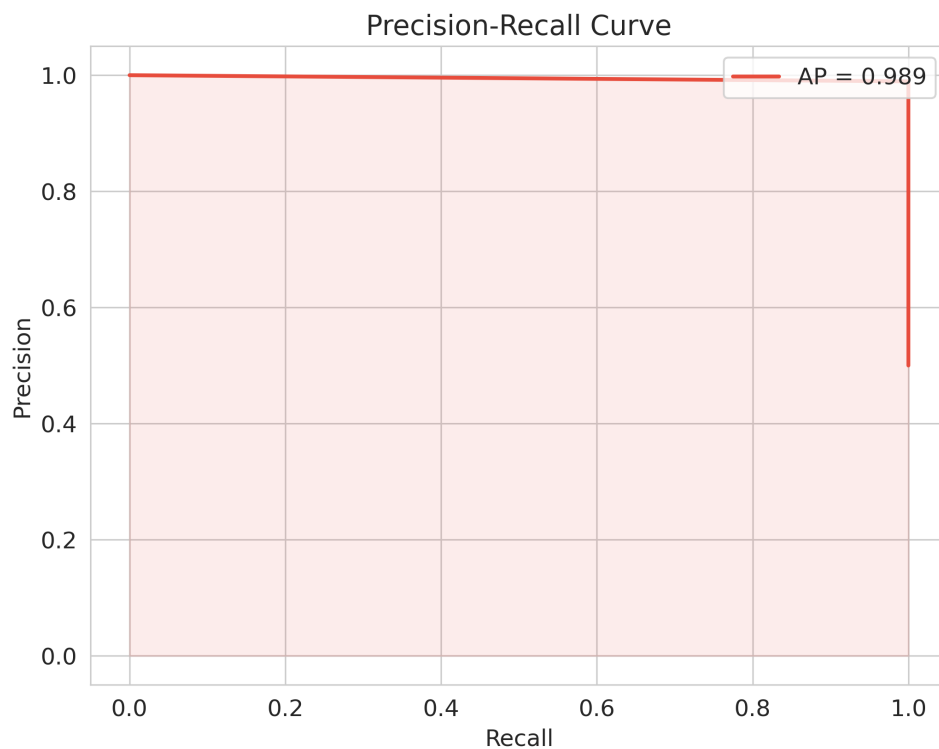


Figure 14: Precision-Recall Curve for SMS Hybrid Model, trained using both UCI SMS Spam Collection Dataset [2] and SMS Spam Dataset [4]

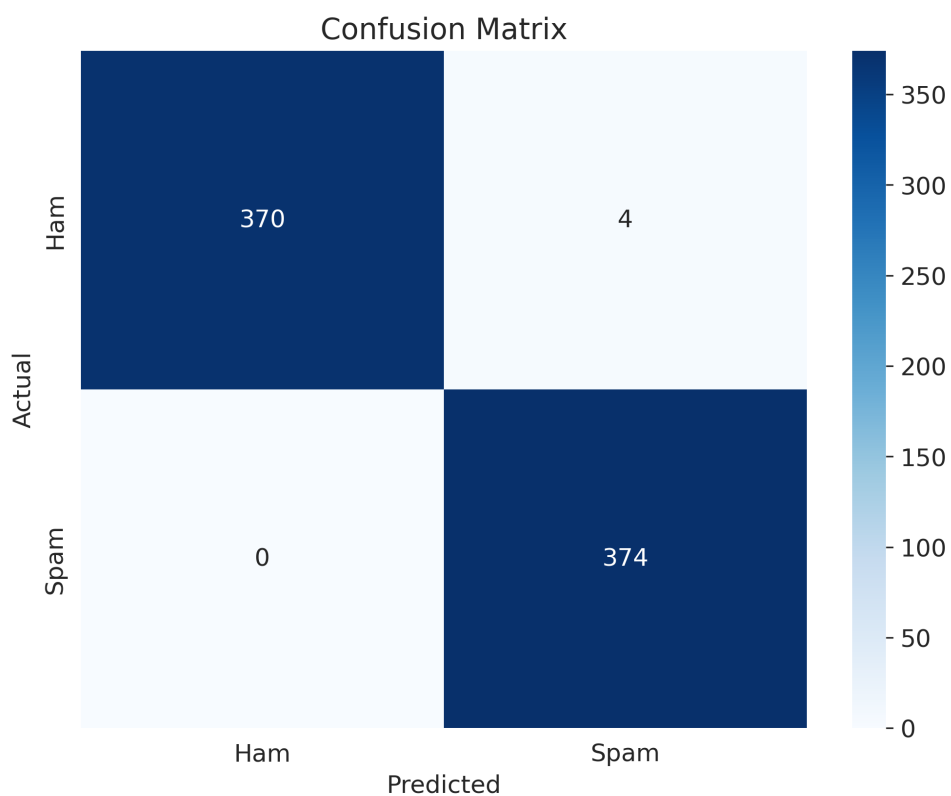


Figure 15: Confusion Matrix for SMS Hybrid Model, trained using both UCI SMS Spam Collection Dataset [2] and SMS Spam Dataset [4]

As shown in the tables above, we can see that the hybrid approach was able to achieve much better results than the traditional approach. The **hybrid approach** was able to achieve an **overall accuracy of 99%**, while the **traditional approach** was only able to achieve an **accuracy of 63%**. This shows that the hybrid approach is much more effective in detecting spam messages, especially when it comes to messages containing URLs. Furthermore, we can also see that the **hybrid approach** was able to achieve a much better **F1-Score**, with a score of **0.99**, while the **traditional approach** only achieved a score of **0.57**.

## Conclusion

In conclusion, our study demonstrates the effectiveness of a hybrid approach for spam detection by combining traditional text-based models with a dedicated URL classification model. By addressing the limitations of existing datasets and employing advanced feature extraction techniques, we achieved high accuracy and robustness across SMS, email, and YouTube comments. The results indicate that our hybrid model significantly outperforms traditional methods, particularly in scenarios involving URLs. Additionally, training datasets being adequately balanced is an absolute must when it comes to reliable spam detection, as well as reliable training techniques such as Stratified K-fold. Overall, this research highlights the importance of continuous innovation in spam detection methodologies to adapt to evolving threats in digital communication.

## References

- [1] S. D. Gupta, S. Saha, and S. K. Das, "SMS Spam Detection Using Machine Learning," *\*J. Phys. Conf. Ser.\**, vol. 1797, no. 1, p. 012017, Feb. 2021, doi: 10.1088/1742-6596/1797/1/012017.
- [2] UCI Machine Learning Repository, "SMS Spam Collection Dataset," [Online]. Available: <https://www.kaggle.com/datasets/uci/sms-spam-collection-dataset>. [Accessed: 2025-03-24].
- [3] S. Kumar, "Malicious and Benign URLs," [Online]. Available: <https://www.kaggle.com/datasets/siddharthkumar25/malicious-and-benign-urls/data>. [Accessed: Mar. 24, 2025].
- [4] T. Kumar, "SMS Spam Dataset," [Online]. Available: <https://www.kaggle.com/datasets/tinu10kumar/sms-spam-dataset>. [Accessed: Mar. 24, 2025].
- [5] N. Bharathi, "Email Spam Dataset," [Online]. Available: <https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset?select=enronSpamSubset.csv>. [Accessed: Mar. 24, 2025].
- [6] V. Venky, "Spam Mails Dataset," [Online]. Available: <https://www.kaggle.com/datasets/venky73/spam-mails-dataset>. [Accessed: Mar. 24, 2025].
- [7] A. Waheed, "Youtube Comments Spam Dataset," [Online]. Available: <https://www.kaggle.com/datasets/ahsenwaheed/youtube-comments-spam-dataset>. [Accessed: Mar. 24, 2025].
- [8] S. Gawai and S. S. Salunke, "Classifying SMS as Spam or Ham Leveraging NLP and Machine Learning Techniques," in *\*2024 International Conference on Communication, Computing and Internet of Things (IC3IoT)\**, 2024, pp. 1-6.