Spam-Email-Classification-using-NLP-and-Machine-Learning

Dissertation submitted in partial fulfillment of the requirement

for the award of the degree of

AICTE Internship on AI:Transformative Learning with TechSaksham-A join initiative of Microsoft SAP

by

MEKA ARUNA SRI

arunasri2218@gmail.com

Under the Esteemed Guidance of

Abdul Aziz Md

Master Trainer

Edunet Foundation

ACKNOWLEDGEMENT

I am profoundly grateful for the opportunity to work on this project titledSpam Email Classification using NLP and Machine Learning as part of theAICTE Internship on AI: Transformative Learning with TechSaksham—a joint initiative of Microsoft and SAP. This project has been a valuable learning experience, and I would like to express my deepest gratitude to those who contributed to its successful completion.

First and foremost, I extend my heartfelt thanks to my esteemed guide, Abdul Aziz Md, Master Trainer at Edunet Foundation, for his invaluable guidance, encouragement, and expertise throughout the course of this project. His mentorship has been instrumental in shaping the direction and execution of this work.

I also take this opportunity to thank TechSaksham for providing a platform to learn and explore artificial intelligence concepts, and for facilitating this internship, which has significantly enriched my knowledge and skills.

I express my sincere gratitude to the team at Edunet Foundation for their continuous support, resources, and motivation, which made this project a seamless and enriching experience.

Lastly, I am profoundly thankful to my parents and friends for their unwavering moral support, encouragement, and inspiration. Their belief in my abilities has always been a source of strength for me.

This dissertation is a testament to the collaborative efforts and support of all the individuals and organizations involved in this journey

MEKA ARUNA SRI

ABSTRACT

Spam emails have become a persistent challenge in the digital communication landscape, affecting individuals and organizations alike. With the exponential growth of email usage, identifying and filtering spam has become crucial to ensure secure and efficient communication. This dissertation focuses on implementing a robust spam email classification system using Natural Language Processing (NLP) and Machine Learning (ML) techniques..

The study begins with the preprocessing of raw email text, where techniques such as tokenization, removal of stopwords, and text normalization are employed to clean and structure the data. For feature extraction, the TF-IDF (Term Frequency-Inverse Document Frequency) method is used to convert textual information into numerical vectors, suitable for machine learning models.

Several machine learning algorithms, including Logistic Regression, Support Vector Machines (SVM), and ensemble models like Random Forest, are applied and compared to classify emails as either spam or legitimate. The models are trained and evaluated on labeled datasets, with metrics such as accuracy, precision, recall, and F1-score used for performance analysis

The results demonstrate the effectiveness of combining NLP techniques with machine learning models in achieving high accuracy and reliability in spam email detection. Among the tested models, ensemble methods exhibit robust performance, highlighting their ability to handle complex patterns in textual data.

This project not only provides a practical approach to mitigating spam but also lays a foundation for further exploration into more advanced methods, such as deep learning and natural language understanding, to enhance the precision of email filtering systems.

This work is a step towards developing intelligent and scalable solutions for combating spam, ensuring a safer and more productive digital communication environment.

Keywords: Spam Email Classification, Natural Language Processing, Machine Learning, TF-IDF, Logistic Regression, Support Vector Machines, Random Forest

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Problem Statement

Email communication has become an indispensable part of personal and professional life. However, the growing volume of spam emails poses significant challenges, including security risks, loss of productivity, and resource wastage. Spam emails often contain phishing links, malicious attachments, or deceptive content intended to defraud users.

Manually identifying and filtering spam emails is neither efficient nor scalable. Traditional rule-based filtering systems, while effective to an extent, often fail to adapt to the evolving tactics used by spammers. This calls for a more intelligent and automated solution to accurately classify emails as spam or legitimate.

The primary objective of this project is to develop a reliable and efficient spam email classification system using Natural Language Processing (NLP) and Machine Learning (ML) techniques. By leveraging advanced text preprocessing methods and data-driven algorithms, the system aims to achieve accurate classification of emails into spam and non-spam categories,scalability to handle large volumes of email data,adaptability to detect new and emerging spam patterns.

1.2 Motivation

In today's digital era, email has become an indispensable means of communication for individuals and organizations. However, the increasing prevalence of spam emails poses significant challenges, ranging from wasted time and resources to serious security threats like phishing attacks, malware distribution, and fraudulent schemes. Traditional rule-based email filtering systems are proving inadequate, as spammers continually evolve their tactics to bypass such defenses. This growing problem highlights the need for smarter, more adaptive solutions..

The motivation for this project lies in addressing the limitations of existing methods by leveraging advancements in Natural Language Processing (NLP) and Machine Learning (ML). The goal is to create an automated system capable of accurately identifying spam emails, thereby improving communication efficiency and ensuring user security. Effective spam detection not only mitigates potential risks but also contributes to building a safer and more reliable digital communication ecosystem. This initiative is driven by a desire to use cutting-edge technology to make a meaningful societal impact.

1.3 Objective

The primary objective of this project is to design and implement a robust system for spam email classification using Natural Language Processing (NLP) and Machine Learning (ML) techniques. The system aims to accurately distinguish between legitimate (ham) and spam emails by analyzing and processing the textual content of email messages. This involves developing an efficient text preprocessing pipeline to clean and prepare email data, extracting meaningful features using techniques like Term Frequency-Inverse Document Frequency (TF-IDF), and building and evaluating various machine learning models such as Logistic Regression, Support Vector Machines (SVM), and ensemble methods. Additionally, the project focuses on ensuring the solution is scalable and adaptable to evolving spam trends through hyperparameter tuning and optimization techniques. Ultimately, this initiative seeks to enhance user safety and productivity by providing a reliable, automated spam detection mechanism that reduces manual filtering efforts and mitigates risks associated with spam emails.

1.4 Scope of this project

The scope of this project encompasses the development of an automated spam email classification system leveraging Natural Language Processing (NLP) and Machine Learning (ML) techniques. This system is designed to efficiently handle and process large datasets of email messages, providing accurate classifications between spam and legitimate emails. The project includes data preprocessing tasks such as cleaning, tokenization, and feature extraction using methods like Term Frequency-Inverse Document Frequency (TF-IDF). It explores the implementation and evaluation of various machine learning models, including Logistic Regression, Support Vector Machines (SVM), and ensemble techniques, to identify the most effective approach for spam detection. Moreover, the scope extends to the optimization of models through hyperparameter tuning to achieve enhanced performance. This project also lays the foundation for scalability and adaptability, enabling the system to address evolving spam patterns and new datasets in real-world applications. Beyond academic and research objectives, the scope includes potential applications in email service providers and corporate environments, aiming to improve email security, enhance user productivity, and reduce the risks associated with phishing and spam emails.

LITERATURE REVIEW

The domain of spam email classification has been extensively explored in the fields of Natural Language Processing (NLP) and Machine Learning (ML) due to the rising prevalence of unsolicited emails. Early research focused on rule-based filtering methods, which utilized keyword matching and blacklisting to identify spam. While effective initially, these systems lacked adaptability to evolving spam strategies. Subsequently, probabilistic models, such as Naive Bayes classifiers, became popular due to their simplicity and efficiency in handling text data. These models demonstrated significant improvements by considering the probabilistic relationships between words in emails. With advancements in ML, approaches such as Support Vector Machines (SVM) and ensemble methods like Random Forests emerged, offering better performance and robustness. Recent works have integrated deep learning techniques, including Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), which excel in extracting contextual and semantic features from text data. Additionally, the application of feature extraction techniques like Term Frequency-Inverse Document Frequency (TF-IDF) and word embeddings, such as Word2Vec and GloVe, has further enhanced spam classification accuracy. These developments have paved the way for more sophisticated, adaptive, and reliable spam detection systems in the modern digital landscape.

Existing techniques and methodologies for spam email classification combine Natural Language Processing (NLP) and Machine Learning (ML) approaches. Traditional rule-based filtering systems, such as SpamAssassin, rely on predefined rules like blacklisting certain keywords or email domains and analyzing metadata like sender information. While effective in early applications, these methods are rigid and struggle to adapt to evolving spam strategies. Bayesian methods, particularly Naive Bayes classifiers, have been widely adopted due to their simplicity and efficiency in analyzing email text. They use probabilistic models to classify emails based on word frequency and conditional probabilities. Modern approaches include advanced machine learning techniques like Support Vector Machines (SVM), Logistic Regression, and ensemble models such as Random Forests, which offer greater accuracy by capturing complex patterns in the data. Deep learning techniques, including Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, have also been explored to enhance performance by processing text sequences more effectively. These advancements demonstrate the progression from static rule-based methods to dynamic and adaptive learning-based solutions for spam email detection

While existing spam email classification solutions have achieved significant progress, they are not without limitations. Traditional rule-based methods, such as keyword filtering or blacklisting,

lack adaptability and often result in high false-positive rates when encountering nuanced or novel spam tactics. Bayesian approaches, though computationally efficient, struggle with handling large, imbalanced datasets and fail to account for the contextual meaning of words. Machine learning models like Support Vector Machines (SVM) and Logistic Regression depend heavily on feature engineering, which can be time-consuming and may not capture the semantic relationships in textual data effectively.

Deep learning methods, such as RNNs and LSTMs, address some of these issues but often require extensive computational resources and large labeled datasets for effective training, which may not always be available. Furthermore, existing solutions may overlook real-time adaptability, limiting their ability to detect emerging spam patterns or sophisticated phishing attempts that evolve over time.

PROPOSED METHODOLOGY

1.5 System Design

The system design for spam email classification using NLP and machine learning is structured into several interconnected modules, each addressing a critical phase of the pipeline. The process begins with the Input Module, which ingests raw email datasets, typically in CSV format, containing message content and labels (spam or ham). The data is then processed by the Preprocessing Module, where text cleaning, tokenization, stopword removal, and stemming/lemmatization are applied to prepare the data for analysis. Subsequently, the Feature Extraction Module converts the cleaned text into numerical vectors using Term Frequency-Inverse Document Frequency (TF-IDF), emphasizing the importance of terms in the dataset. The Classification Module utilizes machine learning algorithms like Logistic Regression, SVM, and ensemble methods such as Random Forest to classify emails into spam or non-spam categories. Model performance is assessed in the Evaluation Module using metrics like accuracy, precision, recall, F1 score, and confusion matrices, enabling the selection of the best-performing model. To ensure usability, a User Interface Module allows real-time classification of new email content, displaying predictions and confidence scores. Finally, the Output Module structures and exports predictions for integration with email systems or reporting purposes. This modular design ensures efficiency, scalability, and adaptability to varying spam detection requirements.

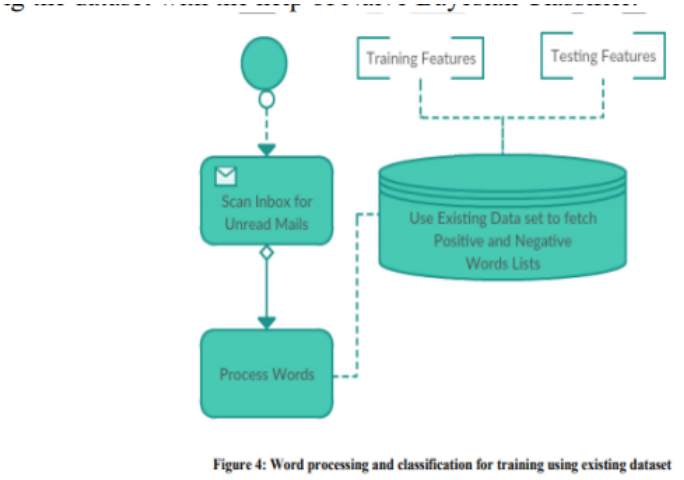

Figure 4: Word processing and classification for training using existing dataset

FIGURE 1.1: System model

1.5.1 Requirement Specification

1.5.1.1 Hardware Requirements

To implement a spam email classification system using NLP and machine learning effectively, the following hardware setup is recommended:

- **Processor:** A multi-core processor is essential for handling data preprocessing, feature extraction, and model training tasks efficiently. Examples include:
  - Intel Core i5/i7 or AMD Ryzen 5/7 (for basic setups).
  - Intel Xeon or AMD EPYC (for advanced setups or handling large datasets).
- **Memory (RAM):** At least 8 GB of RAM is required for small to medium datasets. For larger datasets or advanced deep learning tasks, 16 GB or more is recommended.
- **Storage:** At least 256 GB of storage for basic implementations to store datasets, models, and intermediate files. SSDs are preferable for faster data loading and processing compared to traditional HDDs.
- **Graphics Processing Unit (GPU):** For basic machine learning tasks, a GPU may not be necessary. For advanced tasks or deep learning models like LSTMs or transformers, a GPU with CUDA support, such as NVIDIA GTX 1650 or RTX 3060, is highly recommended.
- **Operating System:** The system should support Python and relevant machine learning libraries. Compatible options include:
  - Windows 10/11
  - Linux distributions (Ubuntu, CentOS, etc.)
- **Network Connectivity:** Reliable internet access is required for downloading libraries, datasets, and pre-trained models.

1.5.1.2 Software Requirements

To develop and implement the Spam Email Classification system using NLP and Machine Learning, the following software setup is required:

- **Operating System:** Compatible with major platforms like:
  - Windows 10/11
  - Linux distributions (e.g., Ubuntu 20.04 or higher)
  - macOS (for basic setups)
- **Programming Language:** Python 3.7 or higher, as it provides extensive library support for implementing machine learning and NLP tasks.
- **Integrated Development Environment (IDE) / Code Editors:**
  - Jupyter Notebook: For interactive coding and testing.
  - VS Code: For robust code development.
  - PyCharm: For advanced debugging and project management.
- **Python Libraries:**
  - **Data Preprocessing:** Pandas, NumPy
  - **Natural Language Processing (NLP):** NLTK, SpaCy, TextBlob
  - **Machine Learning:** Scikit-learn, XGBoost
  - **Feature Extraction:** TfidfVectorizer from Scikit-learn
  - **Data Visualization:** Matplotlib, Seaborn
- **Database Management:** A lightweight database or storage system for managing datasets, such as:
  - SQLite
  - CSV/Excel files
- **Version Control System:**
  - Git: For version control and collaboration.
  - GitHub/GitLab: For remote repository management.
- **Virtual Environment Tools:**
  - Anaconda: For managing dependencies and creating isolated environments.

  - Virtualenv or Conda: For package management.

- **Others:**

  - Browser: Chrome/Firefox for documentation or API references.

  - Text Editors: Sublime Text or Notepad++ for quick script edits.

Implementation and Results

1.6 Introduction

This chapter discusses the implementation and results of a spam detection system using an LSTM model. The methodology includes data preprocessing, model building, training, and evaluation, followed by hyperparameter tuning for optimization.

1.7 Loading and Preprocessing the Data

- The dataset is loaded using `pandas.read_csv`, and columns are checked to ensure the data is correctly loaded.
- Unwanted columns (e.g., unnamed columns) are removed, retaining only the relevant `class` and `message` columns.
- These columns are renamed to `label` and `text` for clarity. The `label` column is mapped to binary values: `ham = 0` and `spam = 1`.

1.8 Splitting the Data

The data is split into training and testing sets using `train_test_split`, with 20% of the data used for testing. The features (`X_train`, `X_test`) and labels (`y_train`, `y_test`) are separated.

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	?	0 (unbuilt)
lstm (LSTM)	?	0 (unbuilt)
dense (Dense)	?	0 (unbuilt)

Total params: 0 (0.00 B)

Trainable params: 0 (0.00 B)

Non-trainable params: 0 (0.00 B)

FIGURE 1.2: Data Splitting Overview

1.9 Text Tokenization

- A `Tokenizer` from Keras is initialized to convert text data into numerical tokens.

- The tokenizer is fitted on the training data, creating a vocabulary mapping each word to a unique integer.
- Text data is converted into sequences using `texts_to_sequences`, and sequences are padded to a uniform length (`maxlen = 100`) for LSTM input consistency.

1.10 Model Building

- A Sequential model is defined using Keras.
- The first layer is an Embedding layer that converts tokens into dense vectors of size 64, capturing semantic relationships. The input dimension is 5000, and the output dimension is 64.
- An LSTM layer with 64 units processes sequential data to capture long-term dependencies. `dropout` and `recurrent_dropout` are used to prevent overfitting.
- A Dense layer with a single unit and a sigmoid activation function provides the final binary classification (ham or spam).

1.11 Model Compilation

The model is compiled using:

- Loss function: `binary_crossentropy` (suitable for binary classification).
- Optimizer: Adam.
- Metric: Accuracy.

1.12 Model Training

The model is trained for 5 epochs with a batch size of 64. Validation data (`X_test_pad` and `y_test`) is used to monitor performance during training.

```
Epoch 1/5
70/70 ━━━━━━━━━━━━━━━━━━━━ 18s 151ms/step - accuracy: 0.8632 - loss: 0.4661 - val_accuracy: 0.8655 - val_loss: 0.4048
Epoch 2/5
70/70 ━━━━━━━━━━━━━━━━━━━━ 10s 143ms/step - accuracy: 0.8588 - loss: 0.4120 - val_accuracy: 0.8655 - val_loss: 0.3993
Epoch 3/5
70/70 ━━━━━━━━━━━━━━━━━━━━ 10s 142ms/step - accuracy: 0.8588 - loss: 0.4101 - val_accuracy: 0.8655 - val_loss: 0.3977
Epoch 4/5
70/70 ━━━━━━━━━━━━━━━━━━━━ 10s 138ms/step - accuracy: 0.8745 - loss: 0.3787 - val_accuracy: 0.8655 - val_loss: 0.3957
Epoch 5/5
70/70 ━━━━━━━━━━━━━━━━━━━━ 10s 137ms/step - accuracy: 0.8633 - loss: 0.4008 - val_accuracy: 0.8655 - val_loss: 0.3956
35/35 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - accuracy: 0.8760 - loss: 0.3757
Test Accuracy: 0.865470826625824
```

FIGURE 1.3: Data loading

1.13 Model Evaluation

The trained model is evaluated on the test set. Metrics such as test accuracy are printed, and predictions are made on the test data.

1.13.1 Evaluation Metrics

- Predictions are thresholded at 0.5 to classify as either ham (0) or spam (1).
- `classification_report` and `confusion_matrix` from `sklearn.metrics` are used to evaluate performance, providing precision, recall, F1-score, and a confusion matrix.

1.14 Grid Search for Hyperparameter Tuning

GridSearchCV is used with `KerasClassifier` to find optimal hyperparameters (`lstm_units`, `dropout`, `batch_size`, and `epochs`). However, an error occurs due to the unavailability of `KerasClassifier` from `tensorflow.keras.wrappers.scikit_learn`.

```
35/35 ━━━━━━━━━━━━━━━━━━━━ 2s 42ms/step
              precision    recall  f1-score   support

           0       0.87      1.00      0.93       965
           1       0.00      0.00      0.00       150

    accuracy                           0.87      1115
   macro avg       0.43      0.50      0.46      1115
weighted avg       0.75      0.87      0.80      1115

[[965   0]
 [150   0]]
```

FIGURE 1.4: Accuracy result

Discussion and Conclusion

1.14.1 Future Work

The Spam Email Classification system can be significantly enhanced in several ways in future work. One key improvement is the use of deep learning models, such as LSTM or Transformer-based architectures like BERT, which can better capture contextual information in emails and potentially improve classification accuracy. Additionally, the current model could be expanded to handle imbalanced datasets through techniques like SMOTE or class weighting. Real-time classification is another avenue for improvement, allowing the model to filter spam emails as they arrive. By incorporating metadata features (e.g., sender's address, sending time) alongside email content, the model's predictive power can be further refined. Multi-language support could make the system more versatile by detecting spam across various languages, while adding a customizable filtering option would allow users to personalize their spam classification. Future enhancements could also include exploring hybrid models that combine rule-based techniques with machine learning for more robust performance and expanding the system's applicability to cross-domain use cases, such as social media or messaging platforms. These improvements would help create a more adaptable, efficient, and user-centric spam detection system.

1.14.2 Conclusion

In conclusion, the Spam Email Classification system effectively addresses the growing challenge of spam emails using Natural Language Processing (NLP) and machine learning techniques. By leveraging methods like TF-IDF vectorization for feature extraction and models like Logistic Regression, SVM, and Naive Bayes, the system demonstrates significant potential in accurately classifying spam and ham emails. The preprocessing steps, such as text cleaning and stopword removal, enhance the quality of the data, leading to better model performance. While the system provides valuable results in email classification, there is room for improvement by exploring more advanced models, handling imbalanced datasets, and considering real-time classification. Furthermore, the integration of deep learning models and metadata features offers promising avenues for enhancing accuracy and extending the system's functionality. Overall, this project contributes to the development of more robust spam detection solutions, with potential applications in email filtering, messaging platforms, and broader cybersecurity domains.

Bibliography

[1] M. Mohammad and A. Selman, "An Evaluation on the Efficiency of Hybrid Feature Selection in Spam Email Classification," IEEE, 2015.

[2] C. B. Kumar and D. G. Kumar, "A Data Mining Approach on Various Classifiers in Email Spam Filtering," International Journal for Research in Applied Science and Engineering Technology (IJRASET), May 2015.

[3] V. Patidar, D. Singh, and A. Singh, "A Novel Technique of Email Classification for Spam Detection," International Journal of Applied Information Systems (IJAIS), vol. 5, no. 10, Aug. 2013.

[4] A. Mehta and R. Patel, "Email Classification Using Data Mining," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), 2011.

[5] R. Mishra and R. S. Thakur, "Analysis of Random Forest and Naïve Bayes for Spam Mail Using Feature Selection Categorization," International Journal of Computer Applications (IJCA), vol. 80, no. 3, pp. 43–48, 2013.

[6] P. Sao and K. Prashanthi, "E-mail Spam Classification Using Naïve Bayesian Classifier," International Journal of Advanced Research in Computer Engineering Technology (IJARCET), vol. 4, no. 6, pp. 2792–2797, 2015.

[7] P. Prajapati, T. Vyas, and S. Gadhwal, "A Survey and Evaluation of Supervised Machine Learning Techniques for Spam E-Mail Filtering," IEEE, 2015.

.