Machine learning Project file

Project name : Email Spam Classification Model

Pranshu kumar - 2021UCA1889
Harshit Bansal - 2021UCA1886
Arpit Agarwal - 2021UCA1934

1. Introduction

Email communication remains one of the most widely used forms of digital communication. However, spam emails, which can range from simple advertisements to phishing attacks, pose a significant risk to individuals and organizations. These spam emails waste storage space, consume bandwidth, and can contain malicious content aimed at stealing sensitive information. Traditional spam filters, based on manually defined rules, often fail to adapt to new spam strategies.

To address these challenges, this project leverages deep learning, particularly Long Short-Term Memory (LSTM) networks, to build an intelligent spam detection system that can automatically identify and filter out spam emails with high accuracy.

2. Objectives

- Develop a machine learning model to classify emails as spam or non-spam based on their content.
- Implement natural language processing (NLP) techniques to preprocess the email text data.
- Build and train a deep learning model using LSTM architecture to leverage the sequential nature of text.
- Fine-tune the model to improve performance using callbacks like EarlyStopping and ReduceLROnPlateau.
- Deploy the trained model to classify new email messages for real-time spam detection.

3. Problem Statement

Spam emails not only annoy users but also pose security threats. Current methods using manual filtering rules are static and ineffective against evolving spam tactics. The challenge is to develop a dynamic system that can efficiently detect spam emails using machine learning techniques. The focus of this project

is to use LSTM networks to exploit the sequential context of email content, making it robust against various spam strategies.

4. Literature Review

Spam email detection has been extensively studied over the years:

- **Rule-based Filters**: The earliest spam detection systems relied on manually defined rules (e.g., detecting specific keywords or phrases). However, these systems are inflexible and require frequent updates.
- **Machine Learning Algorithms**: Algorithms such as Naïve Bayes, SVM, and Decision Trees have been widely used to classify emails by extracting features such as word frequency, email length, and sender information.
- **Deep Learning Approaches**: More recently, neural networks, especially CNNs and LSTMs, have shown superior performance by learning patterns directly from the text without manual feature engineering. LSTMs are particularly effective for text classification due to their ability to capture long-term dependencies in sequential data.

5. Methodology

5.1 Data Collection

We used the publicly available `emails.csv` dataset, which consists of around 5,000 emails labeled as "spam" or "ham" (non-spam). The dataset includes two columns:

- `text`: The content of the email.
- `spam`: A binary label (0 = non-spam, 1 = spam).

5.2 Data Preprocessing

To prepare the data for training, the following steps were taken:

1. **Exploratory Data Analysis (EDA)**:

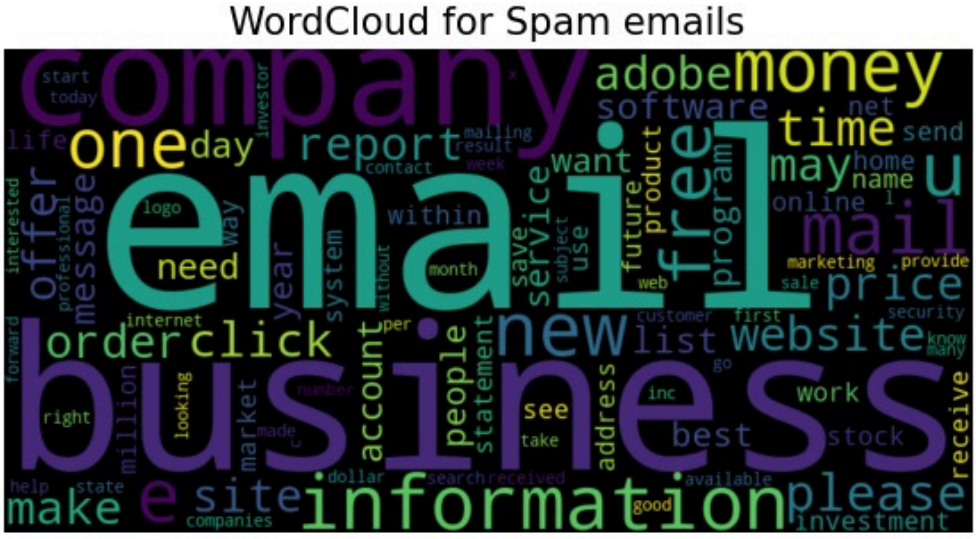
    - Analyzed the distribution of spam vs. non-spam emails using visualizations.
    - Observed that the dataset was imbalanced, with more non-spam emails than spam emails.
2. **Balancing the Dataset**:
    - Downsampled the non-spam (ham) emails to match the number of spam emails for a balanced dataset.
3. **Text Cleaning**:
    - Removed email subject prefixes (like "Subject:").
    - Removed punctuations using Python's `string.punctuation`.
    - Removed common English stopwords using the NLTK library to reduce noise in the text data.
4. **Tokenization and Padding**:
    - Tokenized the cleaned email text into sequences of words.
    - Used the `Tokenizer` class from TensorFlow/Keras to convert text into integer sequences.
    - Padded sequences to a maximum length of 100 tokens to ensure uniform input size for the LSTM model.

6. Implementation

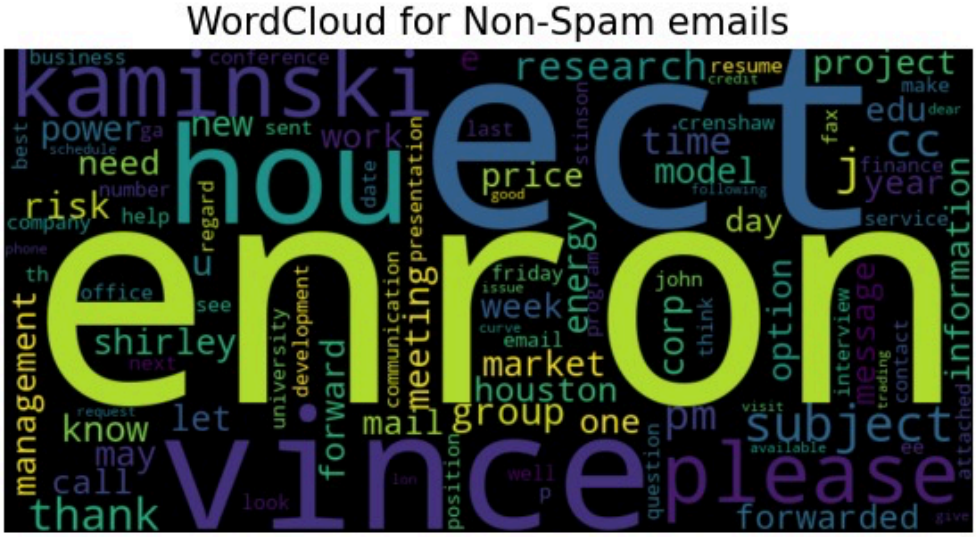
6.1 WordCloud Visualization

To gain insights into the most frequently occurring words in spam vs. non-spam emails:

- **Spam Emails**: Words like "free", "offer", "win", and "click" appeared frequently, indicating the typical nature of spam emails.



- **Non-Spam Emails**: The WordCloud for non-spam emails contained conversational words, reflecting normal communication.



6.2 Model Architecture

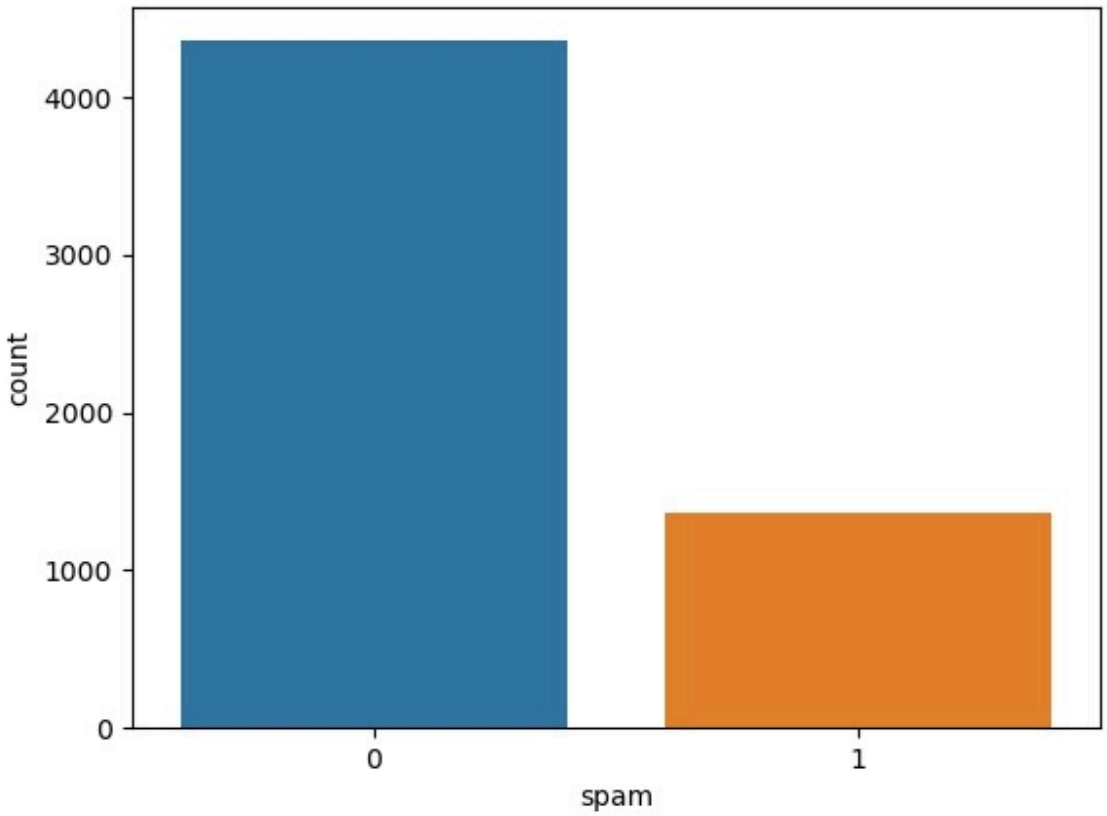
The LSTM-based model was designed as follows:

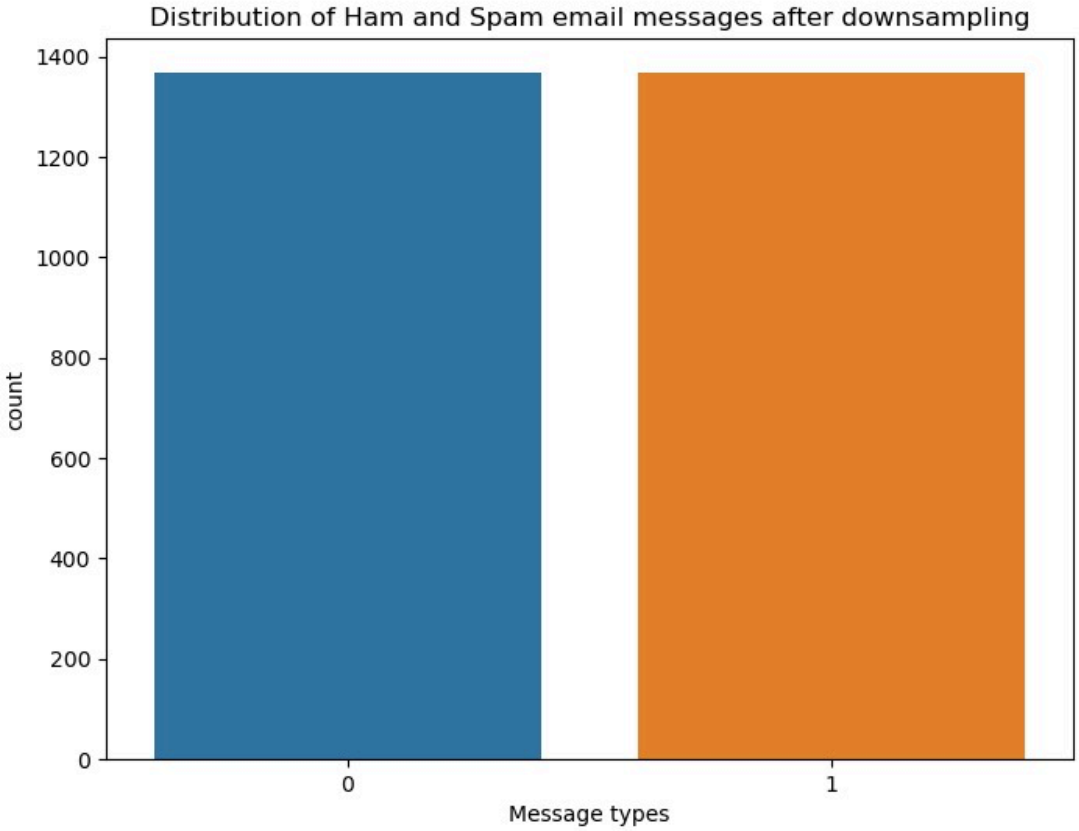
- **Embedding Layer**: Converts words into dense vectors of fixed size (32 dimensions).
- **LSTM Layer**: Captures sequential patterns and dependencies within the text.
- **Dense Layer (ReLU Activation)**: Adds non-linearity to the model.
- **Output Layer (Sigmoid Activation)**: Outputs a probability score to classify an email as spam or non-spam.

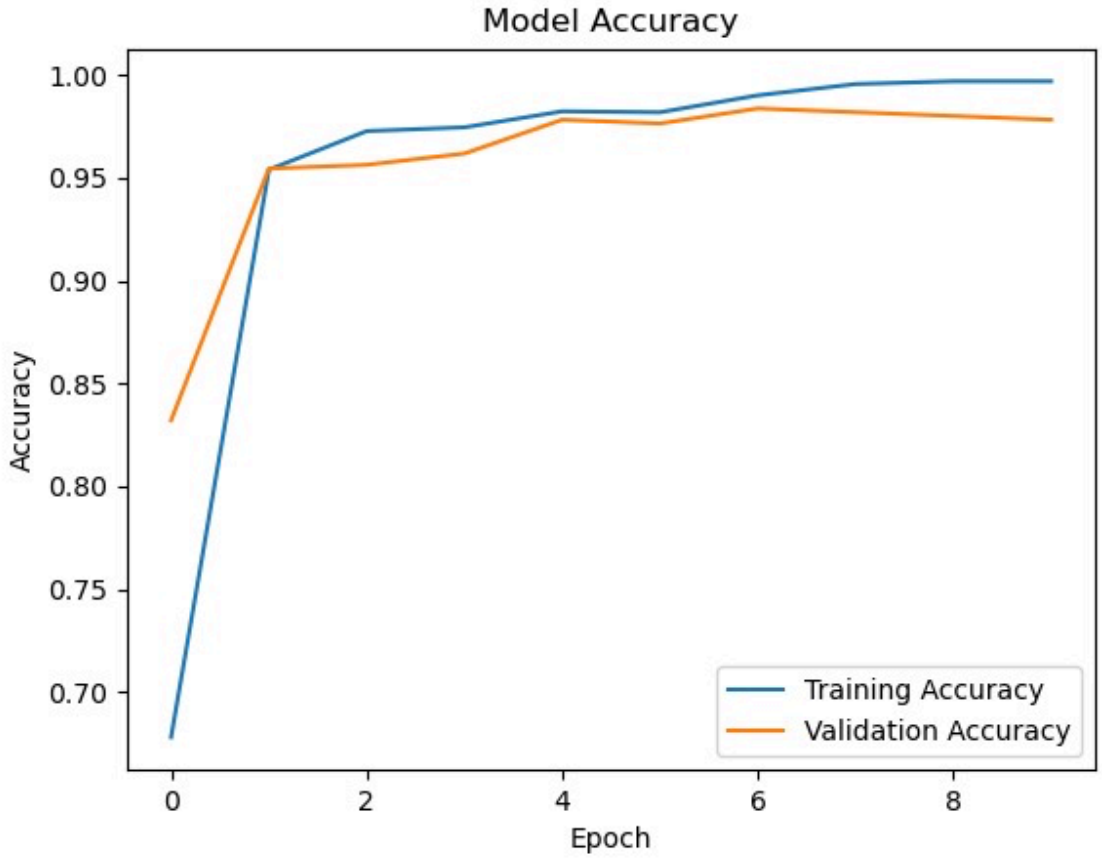
6.3 Model Training

- **Loss Function**: Binary Crossentropy was used since this is a binary classification problem.
- **Optimizer**: Adam optimizer was used for efficient training.
- **Callbacks**:
    - **EarlyStopping**: Stops training if validation accuracy doesn't improve after 3 epochs.
    - **ReduceLROnPlateau**: Reduces the learning rate if validation loss plateaus.

7. Results and Evaluation




Distribution of Ham and Spam email messages after downsampling
1400
1200
1000
800
600
400
200
0
count
0
1
Message types



8. Conclusion

The LSTM-based model achieved high accuracy in classifying emails as spam or non-spam. By leveraging sequential data and capturing long-term dependencies, the model outperformed traditional machine learning classifiers. This project demonstrates the potential of deep learning for spam detection and can be further extended to include more sophisticated features.

---

9. Future Work

- **Hyperparameter Optimization**: Experiment with different configurations to improve performance.

- **Data Augmentation**: Use techniques like synthetic data generation to enhance model robustness.
- **Real-Time Deployment**: Develop a web interface or API to classify incoming emails in real-time.
- **Multi-language Support**: Extend the model to handle emails in languages other than English.

---

10. References

1. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
2. Chollet, F. (2017). *Deep Learning with Python*. Manning Publications.
3. Brownlee, J. (2021). *Deep Learning for Natural Language Processing*. Machine Learning Mastery.