Spam\_Email Detection

Taha Alaa : N-grams feature extraction

Supervisor: Dr. Wesam Ahmed

Group Number: 6

Submission date: May 12th, 2025

**Introduction**

Spam emails are a significant challenge in digital communication, often containing unsolicited advertisements, phishing attempts, and malicious content. This project aims to develop a spam classification system by leveraging **N-grams** for feature extraction and using a combination of traditional machine learning models and a deep learning approach. The models employed include **Naive Bayes**, **Logistic Regression**, **Support Vector Machines (SVM)**, and **Long Short-Term Memory (LSTM)** networks.

**Chapter 2: Preprocessing**

Preprocessing is a crucial step in preparing text data for machine learning and deep learning models. The preprocessing steps are as follows:

* **Lowercasing**: Converts all characters in the text to lowercase to ensure uniformity.
* **Punctuation Removal**: Removes non-alphabetic characters, reducing unnecessary noise.
* **Stopword Removal**: Removes common English words (e.g., "and", "the") that carry little meaning for classification tasks.
* **Stemming**: Reduces words to their base/root form (e.g., "running" → "run") using the **Porter Stemmer**.

These preprocessing steps ensure that the text data is clean and ready for the next stages of feature extraction and model training.

**Feature Extraction: N-grams**

In this project, **N-grams** (unigrams and bigrams) are used for feature extraction:

* **N-grams**: A set of **n** consecutive words that occur together in a given text. For example, for the sentence "This is spam," the unigrams would be ["This", "is", "spam"], and the bigrams would be ["This is", "is spam"].
* **TF-IDF Vectorization**: The **TF-IDF (Term Frequency-Inverse Document Frequency)** method is used to convert text into numerical features, capturing the importance of words or n-grams in the document relative to the entire corpus. We use both **unigrams** and **bigrams** to capture the contextual relationships between words in the text.
* **Max Features**: The feature set is capped at **5000 features** to avoid overfitting and keep computational requirements reasonable.

**Model Training and Evaluation**

Once the text data has been converted into features using **N-grams**, the dataset is split into training and testing sets using an 80/20 ratio. The models used are:

1. **Naive Bayes (MultinomialNB)**
   * Trained on the **N-grams** features using **TF-IDF**. The Naive Bayes model is efficient and works well for text classification tasks.
2. **Logistic Regression**
   * A **Logistic Regression** classifier is trained on the **N-grams** extracted features. It is a common model for binary classification tasks.
3. **Support Vector Machine (SVM)**
   * An **SVM** model with a **linear kernel** is trained to classify the data based on the extracted features. SVMs work well for high-dimensional data and are suitable for text classification.
4. **Long Short-Term Memory (LSTM)**
   * The **LSTM** model is designed to capture sequential dependencies within text. It is trained using padded tokenized sequences derived from the cleaned messages.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Models | Training Accuracy | Testing Accuracy | Accuracy | Recall | Precision | F1-measure | AUC Value |
| Logistic Regression | 95% | 96% | |  | | --- | |  |  |  | | --- | |  |   95% | 70% | 97% | 81% | |  | | --- | |  |  |  | | --- | | 98.9% | |
| Naïve Bayes | 97% | 97% | 97% | 80% | 100% | 89% | 98% |
| SVM | |  | | --- | |  |   98% | 98.5% | 98% | 88% | 98.5% | 93% | 99% |
| LSTM | |  | | --- | |  |  |  | | --- | | 99% | | 97% | 98% | 88% | 98.5% | 91% | 99% |

**Visualization**

**Confusing Matrixes**

**A comparison of blue and white squares

AI-generated content may be incorrect.A blue squares with black numbers

AI-generated content may be incorrect.**

**Deep Learning**

An LSTM-based model was implemented using a N-grams.  
Architecture:  
- E N-grams  
- LSTM (64 units)  
- Dropout (0.5)  
- Dense (1 neuron, sigmoid)