Fake SMS Detection Chatbot
Milestone 03

Team Name: TextTrust Team
Laxminarayana Yadav Pakanati
Teja Kumar Muppala
Vinay Jyothi Gollapalli
Balanjani Kamasani
Naveen Venna
Hema Likhitha Adapa

Abstract

This project, titled *Fake SMS Detection Chatbot*, was developed by the Trust-Text Team to detect fake or spam SMS messages and analyze them in real-time using machine learning. The goal is to create a chatbot that can identify and classify SMS messages as legitimate or spam, helping users stay safe from fraudulent messages.

1 Project Idea

The project by TrustText Team aims to develop a chatbot capable of detecting fake SMS messages using machine learning techniques. The dataset used for this project is the SMS Spam Collection Dataset, which contains labeled SMS messages. These messages are categorized into two classes: **ham** (legitimate messages) and **spam** (fraudulent messages). The dataset includes the following attributes:

- **label**: The label (spam or ham)
- **text**: The actual text of the SMS message

The goal of this project is to classify SMS messages as spam or ham, thereby helping users stay safe from phishing and other fraudulent activities.

2 Dataset

The dataset contains SMS messages labeled as either spam or ham. The data will undergo the following preprocessing steps:

- Removing special characters.
- Tokenizing the text and removing stop words.
- Splitting the dataset into training and testing sets.

3 Tools and Technologies

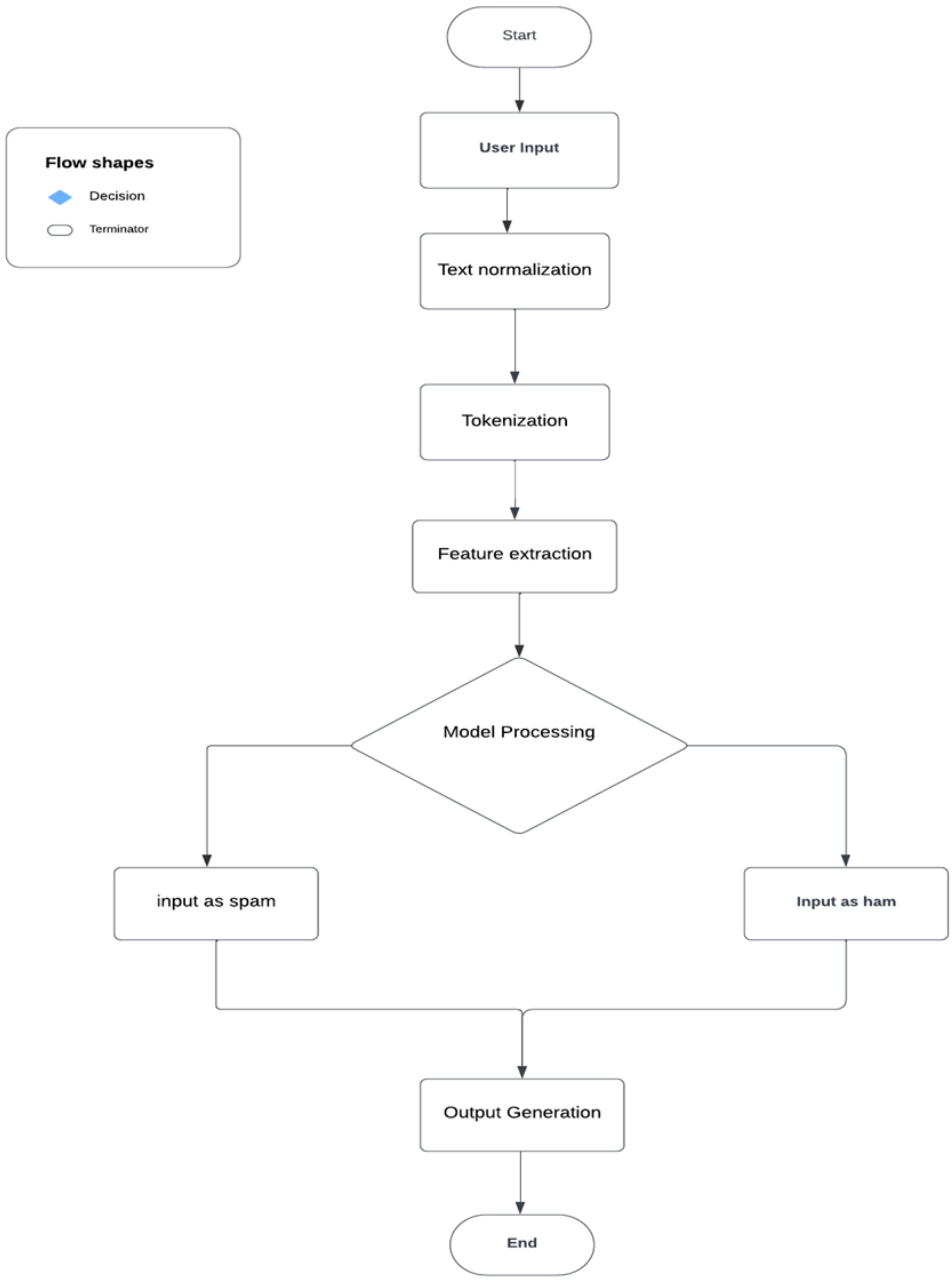
The following tools and technologies were utilized for the Fake SMS Detection Chatbot project:

- **Programming Language:** Python
- **Natural Language Processing (NLP) Libraries:** NLTK, SpaCy
- **Machine Learning Frameworks:** TensorFlow, Scikit-learn
- **Data Preprocessing Tools:** Pandas, NumPy
- **Chatbot Framework:** Rasa
- **Development Environment:** Jupyter Notebook, PyCharm
- **Visualization Tools:** Matplotlib, Seaborn
- **Version Control System:** Git, GitHub
- **Dataset:** SMS Spam Collection Dataset (UCI Machine Learning Repository)
- **Deployment: A platform to deploy the chatbot (e.g., Flask, Django, or a chatbot framework like Dialogflow).**

4 High-Level Architecture

The high-level architecture for the Fake SMS Detection Chatbot is depicted in the block diagram below:

[h!]



Data Flow Diagram for Fake SMS Detection Chatbot

5 Explanation of the Diagram

The following steps outline the data flow and functionality of the chatbot:

- **Start:** The process begins when the user initiates the system.
- **User Input:** The user provides an SMS message to the system for evaluation.
- **Text Normalization:** The system cleans the text by:
  - Converting it to a standard format (e.g., lowercase).
  - Removing special characters.
- **Tokenization:** The cleaned text is split into smaller pieces (tokens), such as individual words.
- **Feature Extraction:** The tokens are transformed into numerical features (e.g., using TF-IDF) for further processing by the model.
- **Model Processing:** The extracted features are sent to the trained machine learning model (e.g., Naive Bayes) for classification. A decision is made to classify the input as either:
  - *Spam*: A suspicious or unwanted message.
  - *Ham*: A legitimate message.
- **Input as Spam:** If the model identifies the SMS as spam, it is labeled accordingly.
- **Input as Ham:** If the model determines the SMS is legitimate, it is labeled as ham.
- **Output Generation:** The system generates the final output, informing the user whether the input was spam or ham.
- **End:** The process ends after the classification result is provided to the user.

6 Implementation

This section details the step-by-step implementation of the Fake SMS Detection Chatbot.

6.1 Data Loading and Preprocessing

The dataset is loaded and preprocessed to clean the text and normalize it for better performance. The preprocessing steps include converting text to lowercase and removing special characters.

Listing 1: Data Loading and Preprocessing

```
import pandas as pd
import re
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
```

```python
from sklearn.metrics import accuracy_score

# Loading the dataset
file_path = "C:\Users\S565725\Downloads\spam.csv"
df = pd.read_csv(file_path, encoding="ISO-8859-1")
# Assuming the dataset has 'text' and 'label' columns.
texts = df['text'].values
labels = df['label'].values

# Normalizing the text

def normalize_text(text):
    text = text.lower()
    text = re.sub(r"[^a-z\s]", "", text)  # Remove special characters
    return text

normalized_texts = [normalize_text(text) for text in texts]
```

6.2 Feature Extraction

The text data is converted into numerical features using TF-IDF vectorization to capture the importance of words.

Listing 2: Feature Extraction

```python

# Tokenization & Feature Extraction
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(normalized_texts)
y = [1 if label == "spam" else 0 for label in labels]

# Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

6.3 Model Training

A Naive Bayes classifier is trained on the dataset to classify an SMS messages as spam or ham.

Listing 3: Model Training

```python
# Train the Model
model = MultinomialNB()
model.fit(X_train, y_train)
```

6.4 Evaluation

The trained model is evaluated using accuracy metrics, and a real-time classification function is implemented.

Listing 4: Evaluation

```python
#Evaluate the Model
y_pred = model.predict(X_test)
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}")
```

6.5 Real-Time Classification

Listing 5: Classification

```
# Step 6: Classify New Input
def classify_text(input_text):
    normalized_input = normalize_text(input_text)
    input_features = vectorizer.transform([normalized_input])
    prediction = model.predict(input_features)[0]
    return "spam" if prediction == 1 else "ham"
```

6.6 Input message from user

```
user_input = input("Enter your message: ")
classification = classify_text(user_input)
print(f"The input is classified as: {classification}")
```

Github repo: Fake$_SMS_Detection$

References

[1] Kayode Sakariyah Adewole, Nor Badrul Anuar, Amirrudin Kamsin, and Arun Kumar Sangaiah. Smsad: a framework for spam message and spam account detection. *Multimedia Tools and Applications*, 78:3925–3960, 2019.

[2] Ashish Bajaj and Dinesh Kumar Vishwakarma. Deceiving deep learning-based fraud sms detection models through adversarial attacks. In *2023 17th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 327–332, 2023.

[3] Hongyu Gao, Jun Hu, Christo Wilson, Zhichun Li, Yan Chen, and Ben Y Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 35–47, 2010.

[4] Abdallah Ghourabi. Sm-detector: A security model based on bert to detect smishing messages in mobile environments. *Concurrency and Computation: Practice and Experience*, 33(24):e6452, 2021.

[5] Mehul Gupta, Aditya Bakliwal, Shubhangi Agarwal, and Pulkit Mehndiratta. A comparative study of spam sms detection using machine learning classifiers. In *2018 eleventh international conference on contemporary computing (IC3)*, pages 1–7. IEEE, 2018.

[6] Saharsh Gupta, Ayush Goyal, Mehul Kumar, Saurav Kumar, and Nishi Jain. Unified detection: Enhancing information reliability through machine learning classification of fake, spam, and legitimate content.

[7] SL Jany Shabu, V Bose, Venkatesh Bandaru, Sardar Maran, and J Refonaa. Spam and fake spam message detection framework using machine learning algorithm. *Journal of Computational and Theoretical Nanoscience*, 17(8):3444–3448, 2020.

[8] Asif Karim, Sami Azam, Bharanidharan Shanmugam, Krishnan Kannoorpatti, and Mamoun Alazab. A comprehensive survey for intelligent spam email detection. *Ieee Access*, 7:168261–168295, 2019.

[9] Mingxuan Liu, Yiming Zhang, Baojun Liu, Zhou Li, Haixin Duan, and Donghong Sun. Detecting and characterizing sms spearphishing attacks. In *Proceedings of the 37th Annual Computer Security Applications Conference*, pages 930–943, 2021.

[10] Maria Soledad Pera and Yiu-Kai Ng. Spamed: A spam e-mail detection approach based on phrase similarity. *Journal of the American Society for Information Science and Technology*, 60(2):393–409, 2009.

[11] Anirudh Ramachandran, Anirban Dasgupta, Nick Feamster, and Kilian Weinberger. Spam or ham? characterizing and detecting fraudulent" not spam" reports in web mail systems. In *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, pages 210–219, 2011.

[12] Minoru Sasaki and Hiroyuki Shinnou. Spam detection using text clustering. In *2005 International Conference on Cyberworlds (CW'05)*, pages 4–pp. IEEE, 2005.

[13] Nilam Nur Amir Sjarif, Nurulhuda Firdaus Mohd Azmi, Suriayati Chuprat, Haslina Md Sarkan, Yazriwati Yahya, and Suriani Mohd Sam. Sms spam message detection using term frequency-inverse document frequency and random forest algorithm. *Procedia Computer Science*, 161:509–515, 2019.

[2] [1] [4] [6] [7] [9] [13] [12] [10] [3] [11] [5] [8]