**ChatGPT Sentiment Analysis using RNN**

**Overview**

This project aims to perform sentiment analysis on a dataset of tweets about ChatGPT using a Recurrent Neural Network (RNN). The model classifies the text into three categories: **good**, **neutral**, or **bad**. We used Keras on Google Colab to build the model, and the final model achieved an accuracy of **89.12%** on the test set.

**Objectives**

* **Preprocess and clean** a large dataset of tweets.
* **Tokenize and pad** the tweet sequences for input into deep learning models.
* **Build and train** an RNN with an LSTM layer to perform sentiment analysis.
* **Evaluate and visualize** the model's performance.
* **Predict the sentiment** of new input text.

**Dataset**

* **Name**: ChatGPT Sentiment Analysis
* **Source**: Kaggle Dataset by **charunisa**
* **Size**: 219,294 tweets
* **Labels**:
  + *good* (1)
  + *neutral* (0)
  + *bad* (2)

**Tools & Libraries**

* **Google Colab**
* **Python 3.x**
* **Pandas**
* **TensorFlow / Keras**
* **Scikit-learn**
* **Matplotlib**
* **Kaggle API**

**Preprocessing Steps**

1. Convert all text to **lowercase**.
2. **Remove URLs**, hashtags, mentions, punctuation, and numbers.
3. Normalize **whitespace** between words.
4. Use **Keras Tokenizer** to tokenize the text.
5. **Pad sequences** to ensure uniform input length.

**Model Architecture**

| **Layer** | **Details** |
| --- | --- |
| **Embedding Layer** | Input dim: 5000, Output dim: 64 |
| **LSTM Layer** | 128 units, Dropout: 0.2 |
| **Dense Layer** | 64 units, Activation: ReLU + Dropout 0.5 |
| **Output Layer** | Dense(3), Activation: Softmax |
| **Optimizer** | Adam |
| **Loss Function** | sparse\_categorical\_crossentropy |
| **Epochs** | 10 |
| **Batch Size** | 64 |

**Results**

* **Training Accuracy**: 91.13%
* **Validation Accuracy**: 89.27%
* **Test Accuracy**: 89.12%

**Epoch-wise Performance:**

* Training vs Validation Loss
* Training vs Validation Accuracy

**Visualizations**

* **Loss Curve**: A graph showing the loss during training.
* **Accuracy Curve**: A graph showing the accuracy during training.

**Example of Generated Plots:**

* Loss curve plot (loss\_plot.png)
* Accuracy curve plot (accuracy\_plot.png)

**Sample Predictions**

Here are some examples of predictions using the model:

python

CopyEdit

predict\_sentiment("I love using ChatGPT!") ➝ 'good'

predict\_sentiment("ChatGPT is okay, I guess.") ➝ 'neutral'

predict\_sentiment("ChatGPT ruined my day.") ➝ 'bad'

**Future Improvements**

* Use **Bidirectional LSTM** to capture better context from both directions.
* Implement an **attention mechanism** to enhance the model’s focus on relevant parts of the text.
* Add support for **sarcasm detection** to improve sentiment prediction.
* Convert the model into a **web-based sentiment analyzer** using Flask or Streamlit for interactive use.

**How to Run**

1. **Clone or Download** the notebook from GitHub.
2. **Upload** your kaggle.json file to authenticate and download the dataset.
3. **Run all cells** in the notebook to start the analysis.
4. Use the predict\_sentiment("your text here") function to test new input.

**Project Structure**

pgsql

CopyEdit

ChatGPT-Sentiment-RNN/

├── ChatGpt Sentiment Analysis using RNN.ipynb

├── kaggle.json

├── file.csv

├── accuracy\_plot.png

├── loss\_plot.png

└── README.md

**Author**

**Mohammed (Mikha)**  
*Beginner AI Engineer, Data Science Enthusiast*  
Contact: essamelsayed256@gmail.com