**Sentiment Analysis Report**

**Introduction**

Sentiment analysis, also known as opinion mining, is the process of determining the sentiment or emotional tone behind a piece of text. This report presents the implementation and evaluation of sentiment analysis using both Machine Learning (ML) and Recurrent Neural Network (RNN) approaches. The dataset used for this analysis is sourced from Kaggle, specifically the "Sentiment Analysis Dataset" available at [this link](https://www.kaggle.com/datasets/abhi8923shriv/sentiment-analysis-dataset?resource=download&select=training.1600000.processed.noemoticon.csv).

The primary goal of this analysis is to classify text data into three sentiment classes: positive, negative, and neutral. This report covers data collection, preprocessing, exploratory data analysis (EDA), feature engineering, model selection and evaluation, challenges faced during the analysis, and results obtained from both ML and RNN models.

**Data Collection**

The dataset consists of text data labeled with sentiment polarity values. The following columns are included in the dataset:

- `polarity`: Sentiment polarity values (0 = negative, 2 = neutral, 4 = positive).

- `id`: Tweet ID.

- `date`: Date and time of the tweet.

- `query`: Query term used in the tweet (or "NO\_QUERY" if not applicable).

- `user`: User who tweeted.

- `text`: Text content of the tweet.

**Preprocessing**

**Handling Missing Values**

The dataset was initially in CSV format. The first step was to load the data and perform preprocessing. This included handling missing values, which were not present in the dataset.

**Label Encoding**

The `polarity` column was encoded as follows:

- 0: Negative

- 1: Neutral

- 2: Positive

**Text Preprocessing**

Text preprocessing involved the following steps:

1. Tokenization: Breaking text into individual words or tokens.

2. Lowercasing: Converting all text to lowercase to ensure consistency.

3. Stopword Removal: Removing common English stopwords.

4. Punctuation Removal: Eliminating special characters and punctuation marks.

5. Numerical Removal: Removing numbers and numerical characters.

6. Lemmatization or Stemming (not applied): Reducing words to their base form.

**Exploratory Data Analysis (EDA)**

Exploratory data analysis was performed to gain insights into the dataset. Key observations include:

- Class distribution: The dataset was imbalanced, with varying counts of positive, negative, and neutral sentiment samples. This imbalance was addressed during ML model training.

- Word cloud: Word clouds were generated to visualize frequently occurring words in each sentiment class.

**Feature Engineering**

Feature engineering was limited in this analysis due to the nature of the problem. The primary features were the preprocessed text data itself, represented as numerical sequences after tokenization.

**Machine Learning (ML) Models**

A variety of ML classifiers were trained and evaluated using the preprocessed text data. The models included:

- AdaBoostClassifier

- BaggingClassifier

- BernoulliNB

- CalibratedClassifierCV

- ComplementNB

- DecisionTreeClassifier

- DummyClassifier

- ExtraTreeClassifier

- ExtraTreesClassifier

- GradientBoostingClassifier

- KNeighborsClassifier

- LinearSVC

- LogisticRegression

- LogisticRegressionCV

- MLPClassifier

- MultinomialNB

- NearestCentroid

- NuSVC

- PassiveAggressiveClassifier

- Perceptron

- RandomForestClassifier

- RidgeClassifier

- RidgeClassifierCV

- SGDClassifier

- SVC

All models were evaluated based on accuracy, precision, recall, and F1-score. The best-performing model based on accuracy was the Support Vector Machine (SVC) classifier with an accuracy of 0.7088.

**Recurrent Neural Network (RNN) Model**

An RNN model was implemented for sentiment analysis using TensorFlow and Keras. The model architecture included:

- Embedding layer (max\_words=5000, input\_length=max\_len)

- LSTM layers (128 units, return\_sequences=True; 64 units)

- Dense layer with softmax activation (output layer for 3 classes)

The model was trained using the preprocessed text data. During training, the following results were obtained:

- Training accuracy reached approximately 94%.

- Validation accuracy stabilized at around 65%.

The evaluation of the RNN model on the test dataset yielded the following results:

- Accuracy: 0.67

- Classification Report (precision, recall, F1-score) for each sentiment class.

**Challenges Faced**

During this analysis, several challenges were encountered, including:

- Imbalanced dataset: The dataset contained an unequal distribution of sentiment classes, which required class balancing techniques.

- Insufficient RAM: Handling large datasets and models in Google Colab led to memory issues, which were resolved by utilizing Google Colab's features and TPU (Tensor Processing Unit).

- Session collapse: Session crashes occurred due to resource limitations, resolved by using TPU acceleration.

- Code issues and solutions: Various coding issues were addressed, including data preprocessing, model training, and evaluation.

**Conclusion**

This report provides a comprehensive overview of sentiment analysis using both ML and RNN approaches. The analysis included data preprocessing, EDA, feature engineering, and model selection and evaluation. While ML models achieved competitive accuracy, the RNN model demonstrated potential for further improvement.

For future work, the following enhancements are recommended:

- Fine-tuning the RNN model for improved accuracy.

- Experimenting with advanced deep learning architectures.

- Handling class imbalance more effectively.

- Conducting more extensive feature engineering.

This analysis serves as a foundation for understanding and implementing sentiment analysis on text data, with potential applications in various domains, including social media sentiment tracking and customer feedback analysis.