Sentiment Analyzer Project

Sebastián Escobar Marín - A00374994
Juan Camilo Gonzalez - A00378625
Gabriel Restrepo - A00377741

Introduction and objectives

Supervised Neural Networks are useful for study and predict some behaviors, in this case we have reviews taken from three web pages: product reviews from Amazon, movies reviews from IMDB and Restaurants reviews from Yelp, all classified in positive or negative, so we want to train a Supervised Neural Network to predict if the review received is positive or negative.

Objectives:

General:

- Implement Neural Networks to detect patterns and predict output values given a set of entries.
- Take advantage of Natural Language Processing benefits to analyze sentiments present in some text.
- Implement Neural Network model tuning methods to get better results.

Specific:

- Analyze correctly the data given for plain preprocessing and the processing.
- Clean the data if the analysis determines it necessary.
- Implements a RNN and LSTM to train it and predict if the review is positive or not.

The Dataset

The dataset received is composed of three .txt files, where each line has a review written by a user and, at the end, a tab (\t) and a number 1 or 0; 1 if positive 0 for negative. First, we put headers to each file, review and positive, to make it easier to transform it to a dataset in Pandas, before making the datasets we clean the reviews by tokenizing the data and deleting the **stopwords** by using NLTK.
Once the data is clean, we can import it into our Jupyter notebook and train our Neural Networks.

Architecture and training process

- **Dummy Classifier as Baseline:** To establish a benchmark, a Dummy Classifier that randomly assigns classes was used. Metrics such as precision, recall, F1 Score, and accuracy were evaluated on the test set.

- **Recurrent Neural Network (RNN) Model:** An RNN model was designed and trained with an Embedding layer, two SimpleRNN layers and a dense layer with a sigmoid activation function. The model was compiled with the 'adam' optimizer and binary cross loss. Training was initially carried out with 25 epochs and a batch size of 128.

- **Long Short-Term Memory (LSTM) Model:** An LSTM model was implemented with an Embedding layer, LSTM layer, Dropout layer and a dense layer with sigmoid activation. At first, the model was compiled and trained in a similar way to the RNN model, with 25 epochs and a batch size of 32.

- **Model Evaluation:** Both models (RNN and LSTM) were evaluated on the test set, and loss and accuracy metrics were calculated. These metrics offer a detailed view of the models' performance in the sentiment analysis task. Performance in terms of accuracy, precision, recall, F1-score and kappa was also evaluated and compared.

Performance evaluations

- **Performance Evaluation: RNN:** The Recurrent Neural Network (RNN) model shows remarkable performance in the sentiment analysis task, as evidenced in the metrics during the 25 training epochs:
    - *Loss:* The loss value decreases consistently from 0.6788 to 0.0258.
    - *Accuracy:* Increases from 0.56 to 0.9867 on the training set and from 0.6733 to 0.6950 on the test set.

- **Performance Evaluation:** LSTM: The Long Short-Term Memory (LSTM) model also shows solid performance in the sentiment analysis task, with the following observations in the metrics:
    - *Loss:* The loss decreases progressively from 0.6820 to 0.0587 during the 25 epochs.
    - *Accuracy:* Increases from 0.5858 to 0.9733 on the training set and from 0.7233 to 0.7533 on the test set.

- **Performance Comparison: RNN vs LSTM:** Both models show outstanding performance, but there are notable differences:

- *Final Accuracy:* The final accuracy on the test set is slightly higher for the LSTM model (0.7533) compared to the RNN model (0.6950).

- *Convergence Speed:* The RNN model seems to converge more quickly in the first epochs, achieving high precision. However, LSTM also achieves good accuracy and maintains consistent performance throughout training.
- *Stability on the Test Set:* Both models show some variability in test set accuracy, but the LSTM appears to be more consistent across epochs.

Appreciations:

- Both RNN and LSTM models offer solid performance in the sentiment analysis task.
- The LSTM tends to slightly outperform the RNN in terms of final accuracy on the test set.
- The choice between RNN and LSTM may depend on factors such as the complexity of the problem, the amount of data, and the computational resources available.

In general, both RNN and LSTM are viable options, and the choice between them will depend on the specific needs of the problem and the available resources. In this case, the LSTM seems to have a slightly better advantage in terms of final accuracy on the test set.

Comparative

A common practice while implementing Neural Networks is finding which are the best hyperparameters. This can be done by trial and error with each combination. Fortunately, we have a tool to optimize this process. After using GridSearchCV function to permute these parameters, the result was an increment on most of the performance criteria (accuracy, precision, recall, F1-score and kappa) for both RNN and LSTM. The output below shows this.

- **Recurrent Neural Network (RNN) Model**

  Without tuning:

```
history = rnn_model.fit(X_train_padded, y_train, epochs=25, batch_size=128, validation_data=(X_test_padded, y_test))
```

```
Accuracy: 0.67
Precision: 0.6794871794871795
Recall: 0.6838709677419355
F1 Score: 0.6816720257234727
```

```
Kappa Score: 0.3391188251001336
```

After tuning:

```
Best parameters: {'batch_size': 16, 'epochs': 5}
Best score: 0.72
```

```
best_rnn_model = create_rnn_model()
tunned_model = best_rnn_model.fit(X_train_padded, y_train, epochs=5, batch_size=16, validation_data=(X_test_padded, y_test))
```

```
Accuracy: 0.7066666666666667
Precision: 0.768
Recall: 0.6193548387096774
F1 Score: 0.6857142857142857
Kappa Score: 0.4165745856353592
```

As we can appreciate. There was an increase in accuracy, precision and kappa score. Training Neural Networks is not a deterministic process so different results are expected to be shown in another execution, but the incremental behavior will tend to remain.

- **Long Short-Term Memory (LSTM) Model:**

Without tuning:

```
lstm_history = lstm_model.fit(X_train_padded, y_train, epochs=25, batch_size=32, validation_data=(X_test_padded, y_test))
```

```
Accuracy: 0.7366666666666667
Precision: 0.7516556291390728
Recall: 0.7322580645161291
F1 Score: 0.741830065359477
Kappa Score: 0.47321627028228497
```

After tuning:

```
Best parameters: {'batch_size': 16, 'epochs': 5}
Best score: 0.7175
```

```
best_lstm_model = create_lstm_model()
tunned_model = best_lstm_model.fit(X_train_padded, y_train, epochs=5, batch_size=16, validation_data=(X_test_padded, y_test))
```

```
Accuracy: 0.7916666666666666
Precision: 0.7846153846153846
Recall: 0.8225806451612904
F1 Score: 0.8031496062992126
Kappa Score: 0.5821727019498608
```

As we can see. There was an increase in all criteria. Again, training Neural Networks is not a deterministic process so different results are expected to be shown in another execution, but the incremental behavior will tend to remain.

Conclusions

We could evidence that the usage of Neural Networks in addition to Natural Language Processing can have a huge impact on how the information given by users of common virtual platforms can be used to detect and predict behavioral patterns. Furthermore, it was shown the importance of a good data preprocessing to capture relevant variables and improve patterns recognition performance. In conclusion, a successful neural network for sentiment analysis implementation involves a combination of data preprocessing, model selection, hyperparameter tuning and the evaluation of multiple indicators and metrics to lead implementation decisions.