Members: Cristian Cardona, David Kim, Sebastián Paz

Introduction

Based on the data collected from Amazon.com, yelp.com and imdb.com on sentiment labeled sentences, we plan to evaluate the behavior of the different classification models (Vanilla RNN, LSTM and DummyClasifier as a base) or their resulting predictions from the aforementioned data to determine which model performs best considering the datasets provided. The Dummy Classifier will only be used as a base model to know the behavior of the data.

The data are divided into two variables, sentence and sentiment, where sentence is a character string and sentiment takes two possible values 1 (for positive) or 0 (for negative).

Objectives

General Objective

Find the best performing model and the best hyperparameters with respect to the data set provided,

Specific Objectives

- Analyze the vanilla RNN model from the data provided.
- Analyze the LSTM model from the data provided above.
- Identify the best hyperparameters for each of the above models.

LSTM

The sentiment analysis model based on a Recurrent Neural Network with layers of long-term memory (LSTM). This type of model is widely used in natural language processing (NLP) due to its effectiveness in handling sequences of data, such as texts.

Model Architecture:

1. **Embedding Layer**:
    - **Input Dimension**: 10000. This indicates that the model expects a vocabulary of up to 10000 unique words.
    - **Output Dimension**: 128. Each word in the vocabulary is represented by a 128-dimensional vector in the embedding space.
    - **Input Length**: 100. The length of the input sequences is 100, meaning that the model processes text sequences of up to 100 tokens.
2. **First LSTM Layer**:
    - **Units**: 64. The first LSTM layer has 64 units (or LSTM cells), meaning that the hidden state in this layer will have a dimension of 64.
    - **Return Sequences**: True. This makes the LSTM layer return the full sequence of hidden states for each time step, which is necessary for stacking another LSTM layer on top.
3. **Dropout Layer**:
    - **Rate**: 0.2. This layer randomly drops 20% of the features (elements in the hidden state) to prevent overfitting.

4. **Second LSTM Layer**:
    - **Units**: 32. The second LSTM layer has 32 units. This layer only returns the final hidden state.
5. **Dense Output Layer**:
    - **Unit**: 1. The output layer has a single neuron with a **sigmoid** activation function, suitable for binary classification (e.g., positive or negative sentiment).

Training Procedure:

- **Model Compilation**: The model is compiled with the **adam** optimizer and **binary_crossentropy** loss function, which are standard choices for binary classification problems.
- **Training**:
    - The model is trained with the training data (**train_padded**, **y_train**) for 10 epochs, meaning it passes through the entire dataset 10 times.
    - The batch size is 128, indicating that the model updates its weights after processing 128 examples.
    - A **validation_split** of 0.2 is used, meaning 20% of the training data is reserved to validate the model after each epoch. This helps monitor and prevent overfitting.

Vanilla RNN

The Vanilla RNN is a simple recurrent neural network (RNN) consisting of a hidden layer of recurrent neurons. The recurrent neurons have a connection between each neuron in the hidden layer, allowing information to flow through the layer over time.

For this model we added 3 types layers:

- **Embedding Layer**:
  In this layer we embed the input sequences into a vector space. In the input_dim, that is the size of the vocabulary, we specified 10,000 in this case. The output_dim parameter that specifies the dimensionality of the embedding vectors, we chose 32 in this case. The input_length parameter that specifies the maximum length of the input sequences,we chose 100 in this parameter

- **SimpleRNN**::
  In this layer we process the embedded sequences, where we specified 64 in the parameter which means the number of units in the recurrent neural network.

- **Dense**:
  In this layer we output a single probability value. In the parameter with the number 1 we specify the number of units in the dense layer. The sigmoid activation function is used to ensure that the output value is between 0 and 1.

Performance evaluation

For LSTM we got these metrics:

```
Accuracy Score: 0.4818181818181818
Precision Score: 0.4818181818181818
Recall Score: 1.0
F1 Score: 0.6503067484662576
Kappa Score: 0.0
```

For Vanilla we got these metrics:

```
Accuracy: 0.7072727272727273
Precision: 0.7363636363636363
Recall: 0.6113207547169811
F1 Score: 0.6680412371134021
Kappa Score: 0.41025641025641024
```

Analysis of the models

- **LSTM:** Based on the provided results, the model seems to be performing reasonably well for binary classification, with an accuracy score of 0.48 and a precision score of 0.48. However, the recall score of 1.0 and the F1 score of 0.65 indicate that the model is heavily biased towards predicting the positive class. This is further supported by the kappa score of 0.0, which suggests that the model's performance is no better than random guessing.

- **Vanilla RNN:** Based on the provided results, the Vanilla model seems to be performing better than the first model, with an accuracy score of 0.71, a precision score of 0.74, and an F1 score of 0.67. This suggests that the SimpleRNN layer is more effective for this particular task than the LSTM layer. However, the recall score of 0.61 indicates that the model is still not perfect at predicting the positive class.

Conclusion

Taking into account the values obtained by each model, that is, the metrics obtained when predicting the classes, we can say that, although LSTM processes sequential data such as text, audio or video and solves the problem of the leakage gradient, a problem that other traditional models have, it does not fit the given task in the best way, possibly because the hyperparameters with which the model was designed are not the best to make predictions in this particular case, and we can say this because of the powerful model that LSTM is.

Future Work

For future occasions, we hope to spend a little more time, through trial and error, and the results of the SearchGrid, to test other combinations of hyperparameters to improve the accuracy of the model LSTM in the classification of sentiment.