Universidad del Valle de Guatemala
Data Science

Luis Santos: 20226
Diego Cordova: 20212

Laboratorio #4

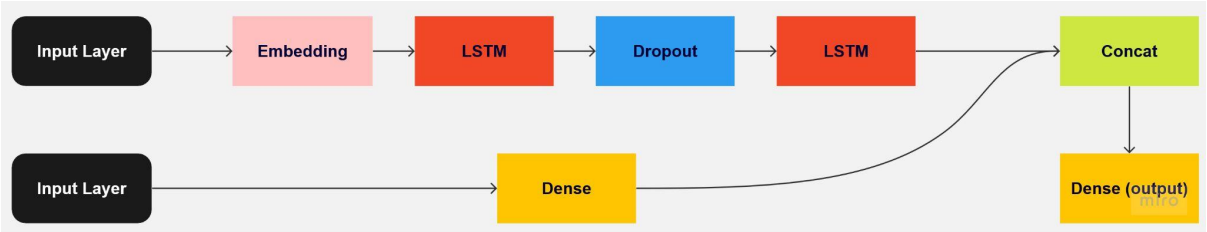
Características del Diseño

En este caso se implementó un modelo utilizando el API funcional que provee Keras. En este caso se utilizó para modularizar el modelo creando 2 pipelines que se encargan de partes diferentes del entrenamiento para luego juntarlos y obtener los insights de ambos.

Para dicha implementación se utilizó la siguiente documentación.
https://www.tensorflow.org/guide/keras/functional_api

Arquitectura del Modelo

Layer (type)	Output Shape	Param #	Connected to
input_18 (InputLayer)	[(None, 128)]	0	[]
embedding_8 (Embedding)	(None, 128, 128)	6400000	['input_18[0][0]']
lstm_18 (LSTM)	(None, 128, 64)	49408	['embedding_8[0][0]']
dropout_12 (Dropout)	(None, 128, 64)	0	['lstm_18[0][0]']
input_17 (InputLayer)	[(None, 1)]	0	[]
lstm_19 (LSTM)	(None, 64)	33024	['dropout_12[0][0]']
dense_14 (Dense)	(None, 1)	2	['input_17[0][0]']
concatenate_6 (Concatenate)	(None, 65)	0	['lstm_19[0][0]', 'dense_14[0][0]']
dense_15 (Dense)	(None, 1)	66	['concatenate_6[0][0]']



Se utilizaron 2 pipelines en donde uno maneja el análisis de texto utilizando una combinación de capas Embedding, LSTM y una capa Dropout (para evitar overfitting) y la otra maneja la proporción de las palabras positivas utilizando una fully connected. Por último los outputs se juntan y se pasan por otra capa fully connected para el output de la red.

Se utilizó esta arquitectura para lograr un entrenamiento que tenga en cuenta tanto la incidencia de las palabras que demuestran una reseña positiva como el orden y la "gramática" del texto (para esto se utilizaron las capas LSTM).

Resultados obtenidos

```
Epoch 1/5
625/625 [==============================] - 56s 82ms/step - loss: 2.8611 - accuracy: 0.7755 - val_loss: 0.4590 - val_accuracy: 0.8497
Epoch 2/5
625/625 [==============================] - 17s 27ms/step - loss: 0.3664 - accuracy: 0.8969 - val_loss: 0.4300 - val_accuracy: 0.8582
Epoch 3/5
625/625 [==============================] - 12s 19ms/step - loss: 0.2864 - accuracy: 0.9313 - val_loss: 0.5658 - val_accuracy: 0.8484
Epoch 4/5
625/625 [==============================] - 12s 19ms/step - loss: 0.2382 - accuracy: 0.9491 - val_loss: 0.4616 - val_accuracy: 0.8500
Epoch 5/5
625/625 [==============================] - 11s 17ms/step - loss: 0.2012 - accuracy: 0.9628 - val_loss: 0.5137 - val_accuracy: 0.8498
<keras.src.callbacks.History at 0x7882fcf2fb50>
```

```
313/313 [==============================] - 2s 7ms/step - loss: 0.5137 - accuracy: 0.8498
Test Accuracy: 84.979999
```

El modelo realmente tenía muy buenos resultados al momento de entrenar llegando a tener un accuracy de más del 90%. Sin embargo al realizar la evaluación no se logró superar el 85%, llegando a obtener un 84.97% específicamente. De cualquier manera esto fue un incremento del 4.5% con respecto al modelo inicial visto en clase. Cabe mencionar que el modelo implementado en este laboratorio cuenta con mucha más optimización al momento de realizar el entrenamiento ya que tarda aproximadamente 3 minutos.