Cairo University
Faculty of Computers & Artificial Intelligence

NLP Experiment Report

Team Members

Fayza Ahmed Sayed Ahmed Abo Zaid **20190376**

Asmaa Refaat Abd Elmaboud Habib **20190101**

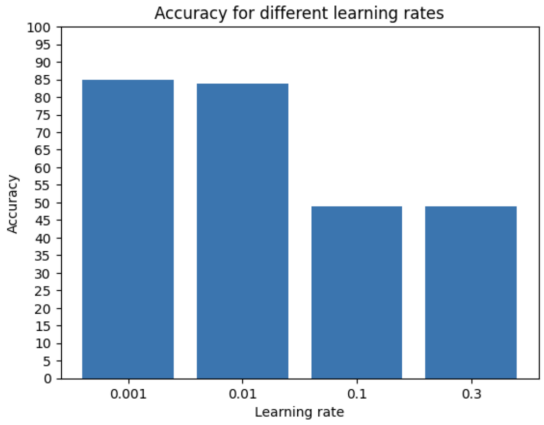
The models that we used:

- CNN
- LSTM

CNN Model:

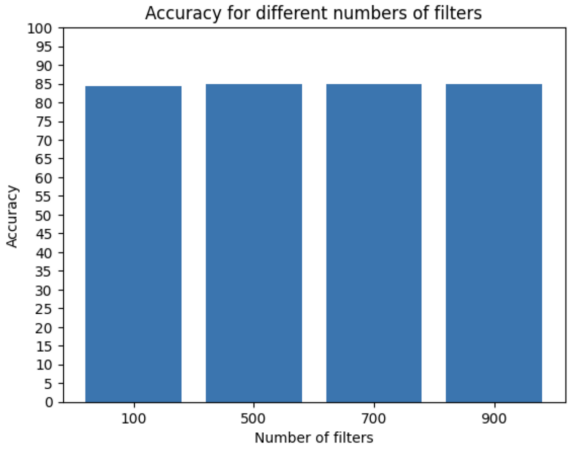
Tuning the CNN model with the hyperparameter (4 trials for the learning rate):

- lr1 = 0.001 → acc = 85.0041232
- lr2 = 0.01 → acc = 84.2453123
- lr3 = 0.1 → acc = 49.6342121
- lr4 = 0.3 → acc = 49.6342122



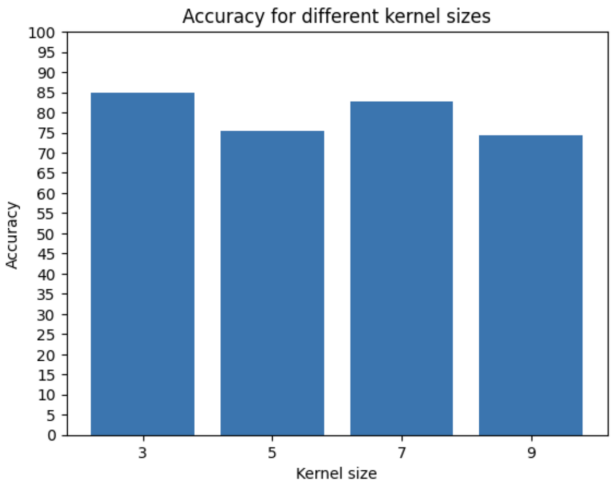
Tuning the CNN model with the hyperparameter (4 trials for the number of filters):

- filter1 = 100 → acc = 84.50012
- filter2 = 500 → acc = 85.35001
- filter3 = 700 → acc = 85.10231
- filter4 = 900 → acc = 85.11012



Tuning the CNN model with the hyperparameter (4 trials for the kernel_size):

- Kernal_size1 = 3 → acc = 85.01203
- Kernal_size2 = 5 → acc = 76.10020
- Kernal_size3 = 7 → acc = 83.72001
- Kernal_size4 = 9 → acc = 75.22132



Best hyperparameters for the CNN model:

Learning rate = 0.001
Num of filters = 500
Kernel size = 3

That gives us an accuracy of `85.05437970161438 %`

```
[19] loss, acc = cnn_model.evaluate(testX, testY)
     print('Test accuracy:', (acc*100))

     1371/1371 [==============================] - 6s 4ms/step - loss: 0.4803 - accuracy: 0.8505
     Test accuracy: 85.05437970161438
```

Each time we run the model gives us a different accuracy in this range [84.5 : 85.3] %

Because of the possibility of getting accuracy less than 85% we tried to make the parameter *`trainable=True`* and we got a better result

Now the accuracy of our model is above 88%

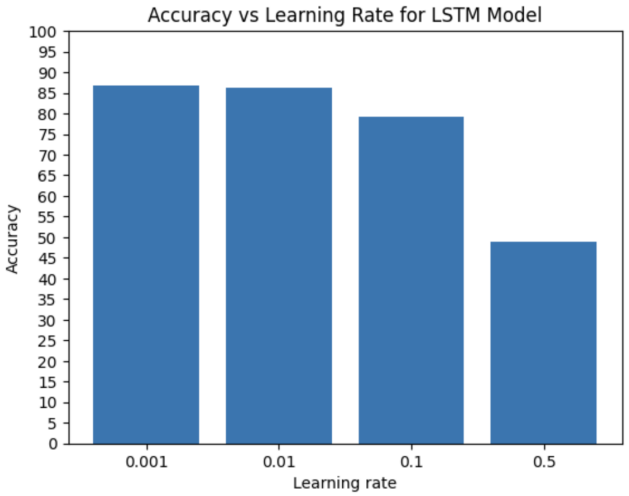
```
[15] loss, acc = cnn_model.evaluate(testX, testY)
     print('Test accuracy:', (acc*100))

     1371/1371 [==============================] - 6s 4ms/step - loss: 0.4895 - accuracy: 0.8836
     Test accuracy: 88.35814595222473
```

LSTM Model:

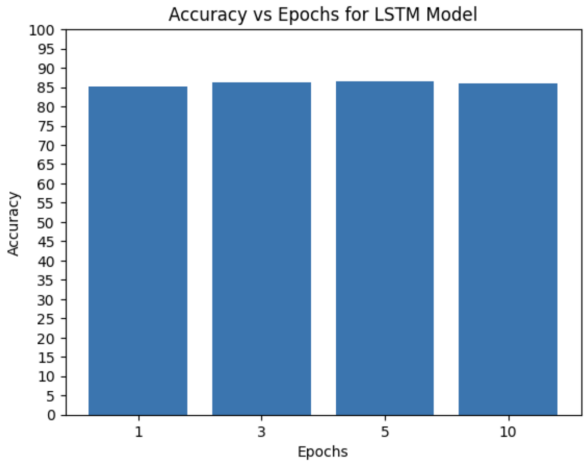
Tuning the LSTM model with the hyperparameter (4 trials for the Learning rate):

- lr1 = 0.001 → acc = 87.1020211
- lr2 = 0.01 → acc = 86.1204351
- lr3 = 0.1 → acc = 79.1322000
- lr4 = 0.5 → acc = 48.9123034



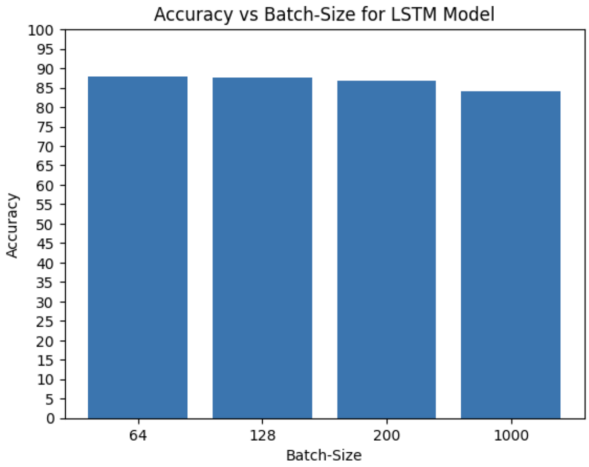
Tuning the LSTM model with the hyperparameter (4 trials for the Epochs):

- epoch1 = 1 → acc = 85.8010211
- epoch2 = 3 → acc = 86.9201351
- epoch3 = 5 → acc = 87.1001100
- epoch4 = 10 → acc = 86.3133134



Tuning the LSTM model with the hyperparameter (4 trials for the Batch-size):

- size1 = 64 → acc = 88.0010201
- size2 = 128 → acc = 87.3200351
- size3 = 200 → acc = 86.1201130
- size4 = 1000 → acc = 84.3141174



Best hyperparameters for the LSTM model:

Learning rate = 0.001
Epochs = 5
Batch_size = 64

That gives us an accuracy of `87.54645586013794 %`

```
[22] lstm_model = LSTM_model(lr=0.001, epochs=5, batch_size=64)
     loss, acc = lstm_model.evaluate(testX, testY, verbose=0)
     print('Test accuracy:', (acc*100))

     Test accuracy: 87.54645586013794
```

Each time we run the model gives us a different accuracy in this range [86.7 : 87.8] %

But when the parameter *`trainable=True`* the accuracy of our model is above 89%

```
[28] lstm_model = LSTM_model(lr=0.001, epochs=5, batch_size=64)
     loss, acc = lstm_model.evaluate(testX, testY, verbose=0)
     print('Test accuracy:', (acc*100))

     Test accuracy: 89.2268419265747
```

The Sentiment Analysis result of the user input:

Positive input:

```
[69] # Taking the input
     new_tweet = input('Enter a new tweet: ')

     Enter a new tweet: ChatGPT is so useful and helpful

[70] # CNN Prediction
     predict_tweet(new_tweet, cnn_model, tokenizer)

     1/1 [==============================] - 0s 24ms/step
     The Sentiment Analysis result of the tweet is positive

[71] # LSTM Prediction
     predict_tweet(new_tweet, lstm_model, tokenizer)

     1/1 [==============================] - 0s 29ms/step
     The Sentiment Analysis result of the tweet is positive
```

Negative input:

```
[75] # Taking the input
     new_tweet = input('Enter a new tweet: ')

     Enter a new tweet: chatgpt is useless

[76] # CNN Prediction
     predict_tweet(new_tweet, cnn_model, tokenizer)

     1/1 [==============================] - 0s 28ms/step
     The Sentiment Analysis result of the tweet is negative

[77] # LSTM Prediction
     predict_tweet(new_tweet, lstm_model, tokenizer)

     1/1 [==============================] - 0s 31ms/step
     The Sentiment Analysis result of the tweet is negative
```

Neutral input:

```
[151] # Taking the input
      new_tweet = input('Enter a new tweet: ')

      Enter a new tweet: openai launched chatgpt
```

```
[152] # CNN Prediction
      predict_tweet(new_tweet, cnn_model, tokenizer)

      1/1 [==============================] - 0s 23ms/step
      The Sentiment Analysis result of the tweet is neutral
```

```
[153] # LSTM Prediction
      predict_tweet(new_tweet, lstm_model, tokenizer)

      1/1 [==============================] - 0s 22ms/step
      The Sentiment Analysis result of the tweet is neutral
```