Machine Learning HW6 Report

學號：r07943150　　系級：電子碩一　　姓名：吳辰鋐

1. (1%) 請說明你實作之 RNN 模型架構及使用的 word embedding 方法，回報模型的正確率並繪出訓練曲線*

word embedding 方式:將 jeiba 切好的詞放入 word2vec 訓練，根據 word2vec 的字彙數量 M 和維度大小 N 建立(M+1) x N 矩陣，再把那 M 個詞在 N 維空間中的向量一一放入矩陣中，+1 因為留一列可供判斷為"others"

ex: 詞彙有 3 個 在 2 維空間中的 word embedding

李弘毅 [0.1 0.99]

級棒 [0.4 0.9]

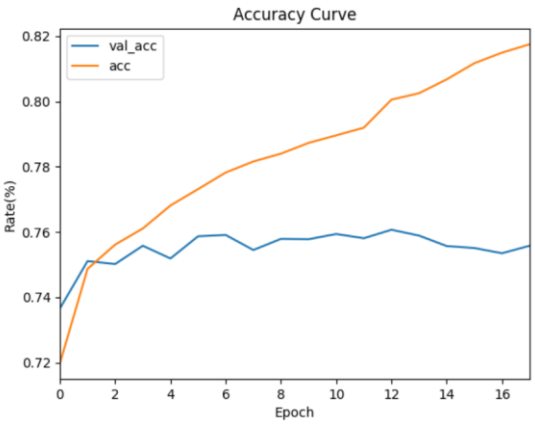
大金 [0.6 0.8]

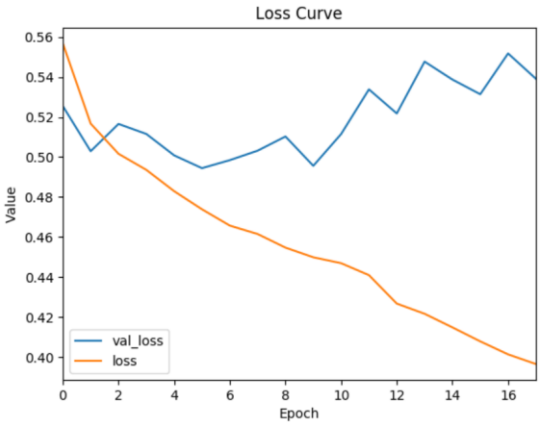
RNN 模型架構:四層雙向 128 neuron 的 LSTM 後接上兩層 256 neuron 的 MLP

Activation function 分別為 tanh 和 LeakyReLU

```
model = Sequential()
model.add(embedding_layer)
model.add(Bidirectional(LSTM(128, activation='tanh', kernel_initializer='Orthogonal',
model.add(Bidirectional(LSTM(128, activation='tanh', kernel_initializer='Orthogonal',
model.add(Bidirectional(LSTM(128, activation='tanh', kernel_initializer='Orthogonal',
model.add(Bidirectional(LSTM(128, activation='tanh', kernel_initializer='Orthogonal',
model.add(Dense(256))
model.add(Dropout(drop_rate))
model.add(LeakyReLU(0.2))
model.add(Dense(256))
model.add(Dropout(drop_rate))
model.add(LeakyReLU(0.2))
model.add(Dense(1))
model.add(Activation('sigmoid'))
adam = optimizers.Adam(lr=0.002, clipvalue=1, beta_1=0.9, beta_2=0.999, epsilon=None,
model.compile(optimizer=adam, loss='binary_crossentropy', metrics=['accuracy'])
model.summary()
```

| | Public Score | Private Score |
|---|---|---|
| RNN | 0.76180 | 0.75660 |



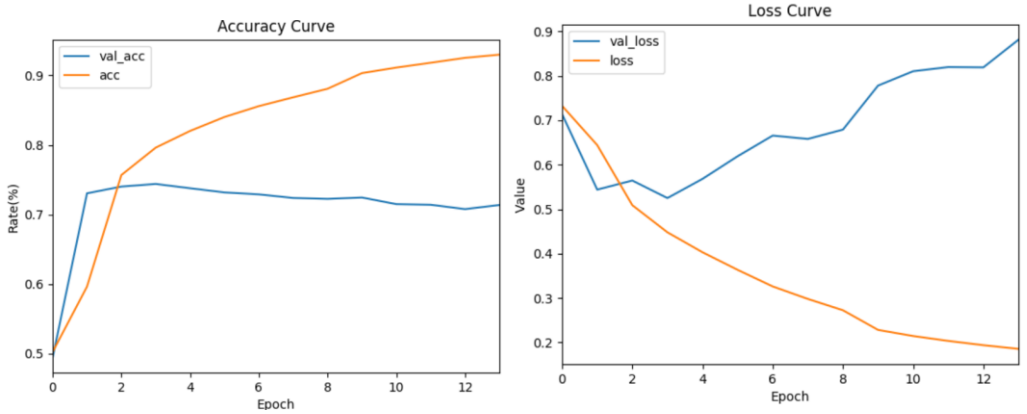



2. (1%) 請實作 BOW+DNN 模型，敘述你的模型架構，回報模型的正確率並繪出訓練曲線*。

模型架構: :將 jeiba 切好的詞放入 word2vec 訓練，根據 word2vec 的字彙數量 M 和 training data 大小 N 建立 N x (M+1)矩陣，若每一列的 data 有出現過這個字或詞則在該對應行+1，形成超大陣列後再放入 MLP 進行訓練

```
model = Sequential()
model.add(Dense(2048, input_shape=X_train[0].shape, activation='sigmoid'))
model.add(Dense(2048, activation='sigmoid'))
model.add(Dense(1024, activation='sigmoid'))
model.add(Dense(1024, activation='sigmoid'))
model.add(Dense(2, activation='softmax'))
adam = optimizers.Adam(lr=0.002, clipvalue=1, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0)
model.compile(optimizer=adam, loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

| | Public Score | Private Score |
|---|---|---|
| BOW+DNN | 0.74860 | 0.74620 |



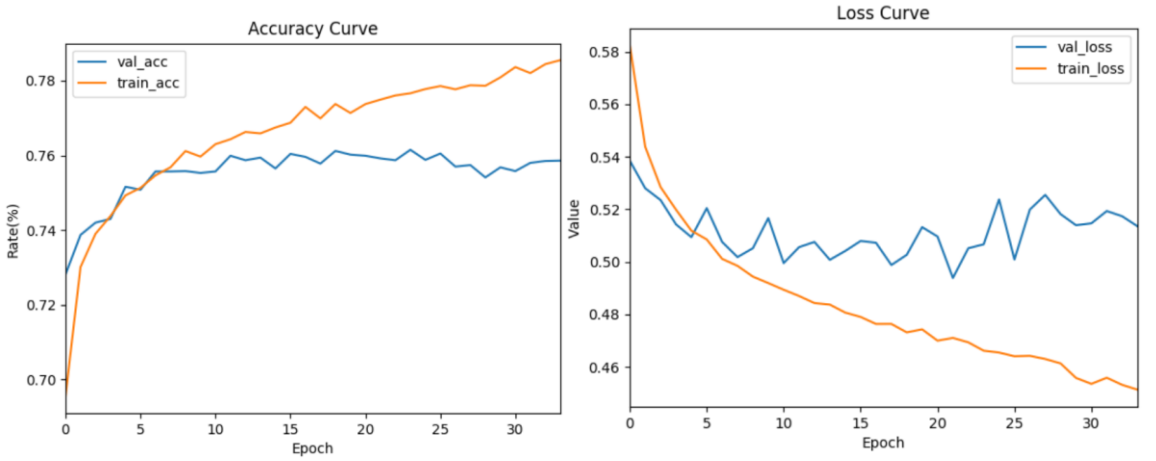

3. (1%) 請敘述你如何 improve performance（preprocess, embedding, 架構等），並解釋為何這些做法可以使模型進步。

試過很多 RNN 架構，換成 GRU、改變層數及神經元數、進行文字處理(如刪除標點符號、語助詞等等) 、加入 BatchNormalization 層，可是效果都微乎其微，比較明顯的改變只有替換 activate function、替換 optimizer、ensemble、加入 dropout。Ensemble 比較容易理解，畢竟每個 model 都有自己的 bias，若進行 ensemble 可將其抵銷，dropout 可避免太快 overfitting，而 activate function 和 optimize 方式(Adam)則是參考網路上推薦的使用方式。

4. (1%) 請比較不做斷詞 (e.g., 以字為單位) 與有做斷詞，兩種方法實作出來的效果差異，並解釋為何有此差別。

以下為不經過 jeiba 直接將全部句子以字為單位切開，用 word2vec 訓練字的關係後再以 Embedding Layer 方式放入 RNN 訓練

| | Public Score | Private Score |
|---|---|---|
| RNN | 0.75920 | 0.75510 |




理論上斷詞後效果會比較好，因為一個字在不同詞中會有不同意義，但從數據上來看著實差異不大，推測因為即使沒有斷詞，RNN 也可以根據前後字判斷語意而不影響準確率。

5. (1%) 請比較 RNN 與 BOW 兩種不同 model 對於 "在說別人白痴之前，先想想自己" 與"在說別人之前先想想自己，白痴" 這兩句話的分數（model output），並討論造成差異的原因。

以下數字為 ”是惡意留言的機率”

| | 在說別人白痴之前，先想想自己 | 在說別人之前先想想自己，白痴 |
|---|---|---|
| RNN | 0.29649514 | 0.58528394 |
| BOW | 0.78856625 | 0.78856625 |

RNN 和 BOW 最關鍵差異在於 RNN 是有順序性的，BOW 只記錄有沒有這個詞彙。

因為順序不同，RNN 可以判斷出第一句話是沒有惡意的，而第二句話是有惡意的，故分數不同。

但這兩句話對 BOW 是相同的，因為出現的辭彙都一樣，故分數會相同。