學號：R01209002 系級： 大氣碩二　姓名：羅章碩

1. (1%) 請說明你實作的 RNN model，其模型架構、訓練過程和準確率為何？(Collaborators: r05229014 鄒適文)

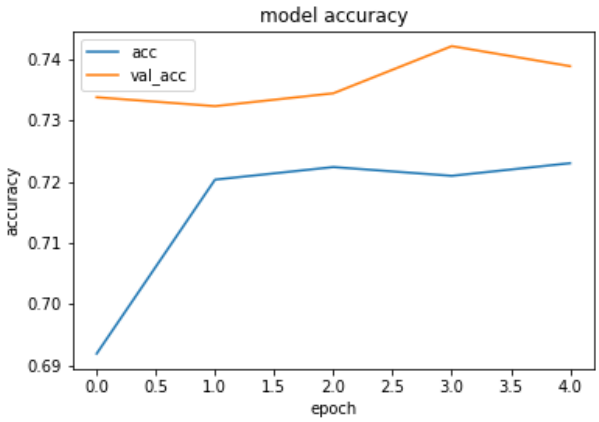
我使用 gensim 的 package 先使用 training data, nolabel_data 和 testing data 中的詞 train 成詞向量，之後將維度設為 128 維當成 input，使用兩層 LSTM 和一層的 DNN，加一層 dropout。其在 kaggle 上的準確度為 public 為 0.814，private 為 0.812，而 kaggle 上最好的分數則是使用到 ensemble 的方式，用了四個在 val_acc 為 0.80~0.81 左右的 model 做預測，得到 kaggle 上 private 0.8187 及 0.819 的分數，可以看到使用 ensemble 的方式的確會使準確度上升。

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_7 (Embedding)      (None, 36, 128)           2800512
_________________________________________________________________
lstm_10 (LSTM)               (None, 36, 256)           394240
_________________________________________________________________
lstm_11 (LSTM)               (None, 256)               525312
_________________________________________________________________
dense_17 (Dense)             (None, 128)               32896
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_18 (Dense)             (None, 1)                 129
=================================================================
Total params: 3,753,089
Trainable params: 952,577
Non-trainable params: 2,800,512
_________________________________________________________________
```

2. (1%) 請說明你實作的 BOW model，其模型架構、訓練過程和準確率為何？(Collaborators: r05229014 鄒適文)

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 64)                128064
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               8320
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
batch_normalization_1 (Batch (None, 128)               512
_________________________________________________________________
dense_3 (Dense)              (None, 256)               33024
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_4 (Dense)              (None, 512)               131584
_________________________________________________________________
dropout_4 (Dropout)          (None, 512)               0
_________________________________________________________________
batch_normalization_2 (Batch (None, 512)               2048
_________________________________________________________________
dense_5 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_5 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_6 (Dense)              (None, 1024)              525312
_________________________________________________________________
dropout_6 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 1025
=================================================================
Total params: 1,092,545
Trainable params: 1,091,265
Non-trainable params: 1,280
_________________________________________________________________
```

上述截圖為我使用的 BOW model 的 summary，字典數量為 2000，總共使用了大約 1 百萬個神經元，並在 keras 的 tokenizer.texts_to_matrix 使用 count 模式而 batch_size 為 32。在 kaggle 上的準確率沒有很高，private、public 分別只有得到 0.70866、0.70856 的分數而已，可見 bag of model 雖然神經元數遠大於 RNN 但是表現並不會比 RNN 來的好。而以下則是我在 training 過程中的 acc、val_acc 的訓練過程。因為我有使用 EarlyStopping(patience=3)，所以在 train 沒多久就結束了。



3. **(1%) 請比較 bag of word 與 RNN 兩種不同 model 對於"today is a good day, but it is hot"與"today is hot, but it is a good day"這兩句的情緒分數，並討論造成差異的原因。**

   (Collaborators: r05229014 鄒適文)

   答：對於"today is a good day, but it is hot"而言 RNN 預測出來的分數 0.27569708(情緒不佳)，而 BOW model 則為 0.44574633，在另外一句"today is hot, but it is a good day"時 RNN 的預測為 0.95018309(情緒佳)，而 BOW model 同樣也是 0.44574633。

   RNN 因為會考慮”詞”出現的順序，所以在判斷語意的情緒上時就算是同樣的詞組成的句子但是因為順序不同則可以出現不同的分數，而對於 BOW model 而言，它不會考慮到詞出現的順序只考慮句子中詞出現的次數，所以兩個不同的句子但是詞都相同的時候，它就會出現相同的分數。因此我覺得在語意情緒分析上，RNN 可以有較佳的表現。

4. **(1%) 請比較"有無"包含標點符號兩種不同 tokenize 的方式，並討論兩者對準確率的影響。**

   (Collaborators: r05229014 鄒適文)

   答：有標點符號時在 kaggle 上得到的分數 private 、public 分別為 0.78477、0.78376，而沒有標點符號時的分數則為 0.78834、0.78679。

從上述可知，把標點符號去掉之後可以微微的使正確率上升，但是標點符號的存在對於 RNN 判斷語意情緒時並沒有很大的影響，因為有無標點符號的正確率差別並不大。就我自己感覺大部分標點符號對於 RNN 而言是沒有甚麼感覺的，因此拿掉這些符號會使得分數有所上升。除了一些可能具有情緒的符號像是驚嘆號或是…或是問號等等。

5. (1%) **請描述在你的 semi-supervised 方法是如何標記 label，並比較有無 semi-surpervised training 對準確率的影響。**
(Collaborators: r05229014 鄒適文)
答：我使用 semi-supervised 中的 self-training。先 train 好一個 model 之後預測 no_label data 的值，當預測出來的值大於 0.7 就標記成 1 而小於 0.3 則標記成 0，並將這些 data 加到 training data 中再 train 一次 model。最後比較沒有使用 semi-supervised 時 model 在 kaggle private、public 上的分數為 0.78834 和 0.78679（在此使用之 RNN model 和第一題不一樣），而使用 self-training 之後可得到 0.79994、0.79930 的分數，亦即我使用 semi-supervised 的方法後，有提升 model 的準確率。