學號：B03502125 系級： 機械四 姓名：倪嘉宏

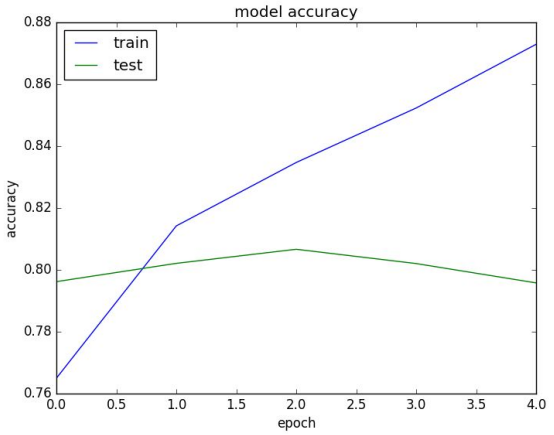
1. (1%) 請說明你實作的 RNN model，其模型架構、訓練過程和準確率為何？
   (Collaborators:)

   答：

	loss	acc	val_loss	val_acc
Epoch0	0.484	0.7642	0.4527	0.7916
Epoch1	0.4105	0.8153	0.4349	0.8054
Epoch2	0.3754	0.8341	0.4277	0.8073

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 32, 64)            524288
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 32, 128)           16512
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 32, 128)           32896
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 32, 128)           32896
_________________________________________________________________
dropout_1 (Dropout)          (None, 32, 128)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 48)                33984
_________________________________________________________________
dropout_2 (Dropout)          (None, 48)                0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               6272
_________________________________________________________________
dense_2 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_3 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 129
=================================================================
Total params: 680,001
Trainable params: 680,001
Non-trainable params: 0
_________________________________________________________________
```

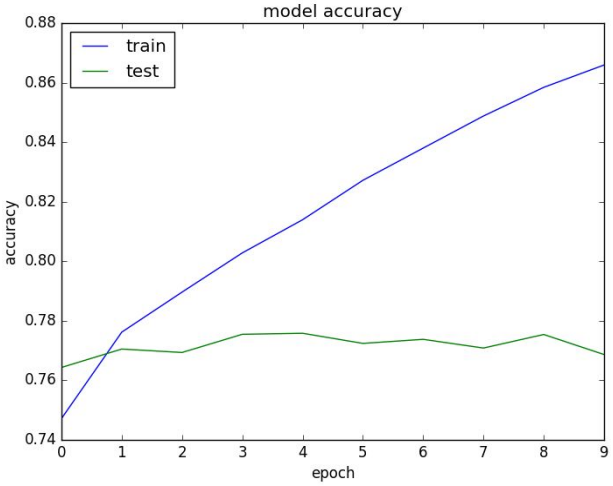
模型架構與訓練過程數據參考上面的圖片（注：兩層dropout皆設為0.2）。Epoch設定為3，因為第四個Epoch開始overfitting現象會變得太嚴重，使得真正的準確率反而下降。Batch Size設為512可在訓練效率與準確率之間取得最佳平衡。



每次train完之後都會用完全乾淨未曾使用過的部份 training data（大約切8%出來）再做一次validataion。通常這次的validation都會比keras內建的split validation還要再高0.2%左右，以上面表格的狀況來說，大約可以達到81.0的準確率。

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:)




```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 512)               512512
_________________________________________________________________
activation_1 (Activation)    (None, 512)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 1026
_________________________________________________________________
activation_2 (Activation)    (None, 2)                 0
=================================================================
Total params: 513,538
Trainable params: 513,538
Non-trainable params: 0
_________________________________________________________________
```

答：模型架構與訓練過程數據參考上面的圖片（注：dropout設為0.5）

準確率大約可以達到77%的水準。我想其實單純一靠一句話當中出現的字給出分數，就足以判斷大多數語句的情緒分數了。少數例外則是有比較複雜的轉折，或是用正面語句酸人等等情況會讓BOW判斷失準，而很可能這些語句正是造成判斷能力略遜於RNN的關鍵。

	loss	acc	val_loss	val_acc
Epoch0	0.5177	0.747	0.4885	0.7644
Epoch1	0.4693	0.7765	0.4777	0.7699
Epoch2	0.4486	0.7899	0.4763	0.7694
Epoch3	0.4281	0.803	0.4714	0.7754
Epoch4	0.4072	0.8144	0.4728	0.7755
Epoch5	0.3853	0.8267	0.476	0.7746
Epoch6	0.3639	0.8385	0.4812	0.7732
Epoch7	0.3435	0.8487	0.4915	0.7701
Epoch8	0.3256	0.8587	0.4989	0.7734
Epoch9	0.3088	0.8662	0.5101	0.7698

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:)

	RNN	BOW
today is a good day, but it is hot	0.3919	0.7026
today is hot, but it is a good day	0.9696	0.6981

答：RNN可以看出兩句話的得分有很顯著的差異，能夠導致輸出結果時第一句被判定為負面情緒，而第二句被判定為正面情緒。BOW則沒有這個現象，兩句話都被認為是還不錯的情緒。造成這個差異最主要的原因正是BOW沒有管輸入順序，只單存從出現的字去決定這句話的情緒基調。偏偏這個兩個句子包含的字恰好一模一樣，必須從出現順序當中揣摩作者真正想要表達的意思（but後面才是句子意思的重點）。而RNN當中的神經網路是有考慮進去單字出現的順序，故才能順利的判讀出正確的意思。

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:)

答：本提training細節請參考第一題。未移除標點符號與移除標點符號後分別進行五次的training並驗證成績。五次取平均後，不移除標點符號的tokenize方法準確率為80.91，而移除標點符號則為80.39，相差多達0.6個百分點。我認為相差這麼多其實也相當合理，再twitter當中使用的標點符號其實可以表達很豐富的情緒。譬如出現 ”...” 很有可能是無奈、無語的意思，整句的意思更有可能是負面的。諸如此類的標點符號對於判斷整體的情緒絕對很有幫助。

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:)

答：先以第一題方式訓練出一套model，再使用這套model去predict no label data，並算出每筆資料的情緒分數（以0至1的小數點表示）。若大於0.9分則標記為一，若小於0.1分則標記為零，並將這些資料納入training data，再送入model中訓練。上述流程可以重複一至兩次。可惜的是，試過數種不同的參數組合，semi-supervised 對於我的準確率都沒有正向的幫助。下表附上訓練過程的數據作為參考。

	loss	acc	val_loss	val_acc	
Epoch1/3	0.4847	0.7635	0.4453	0.7960	
Epoch2/3	0.4132	0.8136	0.4310	0.7997	
Epoch3/3	0.3784	0.8326	0.4258	0.8045	
Data append	共新增504948筆資料 合計633832				此時實際accuracy為80.90%
Epoch1/3	0.2313	0.8979	0.5192	0.7995	
Epoch1/3	0.1200	0.9585	0.4909	0.8048	
Epoch1/3	0.1092	0.9610	0.4701	0.8030	
Data append	共新增726747筆資料 合計726747				此時實際accuracy為79.59%
Epoch1/3	0.6833	0.5704	0.7018	0.5011	
Epoch1/3	0.6833	0.5704	0.7009	0.5011	
Epoch1/3	0.6833	0.5704	0.7051	0.5011	
					此時實際accuracy為50.31%

可以觀察到第一次的data append之後loss 變成極低。因為data append進去的資料就是這個model算出來的答案，因此會讓loss如此低也是正常的。如果在semi-supervise的過程當中混入兩三種不同的model可能會有更好的結果。