學號：B03902016 系級： 資工四 姓名：周聖筌

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？

答：

首先先利用沒有label的data做word2vec，得到一個總共43723個字，每個字對應到一個200維的一維陣列的字典。

再來利用keras內建的tokenizer中的texts_to_sequences將每個詞對應為一個數字，同樣的詞會有同樣的數字。並利用keras內建的pad_sequences將所有的句子pad到和最長的句子一樣長。

再來建立一個matrix，對應的列數即代表經過texts_to_sequences後的數字，而一列對應的vector即為word2vec中該字的vector。

model的layer一開始先利用embedding將每個數字從轉為200為vector。

再來就一層LSTM（activation為tanh）和一層dense（activation為softmax）。

訓練過程總共做10個epochs

kaggle public score : 0.81929

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 37, 200)           16425000
_________________________________________________________________
lstm_1 (LSTM)                (None, 400)               961600
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 802
_________________________________________________________________
activation_1 (Activation)    (None, 2)                 0
=================================================================
Total params: 17,387,402
Trainable params: 962,402
Non-trainable params: 16,425,000
```

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？

答：

利用keras內建的tokenizer，先設定tokenizer的num_words為20000，讓最後矩陣不要太大。再來利用texts_to_matrix，mode設定為'count'，就可以算出每一句中對應20000維的字數。

最後丟進如下的layer，最後一層的activation是softmax，其他都是relu，總共跑10個epochs。跑出來的結果叫LSTM差，但也有快八成。
kaggle public score : 0.79514

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 500)               10000500
_________________________________________________________________
dropout_1 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_2 (Dense)              (None, 500)               250500
_________________________________________________________________
dropout_2 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_3 (Dense)              (None, 500)               250500
_________________________________________________________________
dropout_3 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_4 (Dense)              (None, 500)               250500
_________________________________________________________________
dropout_4 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 1002
_________________________________________________________________
activation_1 (Activation)    (None, 2)                 0
=================================================================
Total params: 10,753,002
Trainable params: 10,753,002
Non-trainable params: 0
```

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。

答：

	"today is a good day, but it is hot"		"today is hot, but it is a good day"	
RNN	0 : 0.03992813	1 : 0.96007192	0 : 0.08580543	1 : 0.91419425
BOW	0 : 0.39878783	1 : 0.6012122	0 : 0.39878783	1 : 0.6012122

BOW對於兩句話的預測是一模一樣的，因為BOW的只看有的字的字數的緣故，兩句話中包含字是一模一樣的，因此預測一模一樣，都是預測為正向評論。
而RNN因為會看句子中資的先後順序，因此預測結果不太一樣，但也都是較大機率預測為正向評論。

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。

答：

我一開始時做的是沒有標點符號，就如第一題所描述，kaggle上public結果是0.81929。

把tokenizer中的filter改為只有'\n'（換行符號），並對重新對有標點符號的unlabel data做word2vec，做出一個總共有46352個詞的字典。再來就和第一題架構一一樣，這樣做出來的結果有比較好，kaggle public變為0.82166，就是我最佳的結果。

我認為這樣的原因是因為有些句子因為標點符號的不同就會改變語氣，因此這樣判斷會比較準。

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。

答：對於semi-supervised，我是將threshold設為0.7。每次fit完兩個epochs後，我就會將unlabel的data丟去model預測，一個一個看所有預測結果，若兩維其中一個值超過0.7，就代表他對這句話判斷是比較有信心的，就把他加入訓練資料中，再下去fit兩個epochs，然後反反覆覆重複5次。

這樣出來的kaggle public是 0.81938，並沒有顯著進步。