

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：先用 Word2Vec 得到每個字的向量，再把 tokenize 的編號轉成 Word2Vec 的 index，用 keras 預設的 embedding 層。

模型架構：

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 128)	2560000
lstm_1 (LSTM)	(None, 128)	131584
dense_1 (Dense)	(None, 1)	129
Total params: 2,691,713		
Trainable params: 2,691,713		
Non-trainable params: 0		

訓練過程和準確率：

```
200000/200000 [=====] - 743s 4ms/step - loss: 0.4589 - acc: 0.7829
Epoch 2/3 an not read it...
200000/200000 [=====] - 732s 4ms/step - loss: 0.4073 - acc: 0.8133
Epoch 3/3 你=107學年...
200000/200000 [=====] - 713s 4ms/step - loss: 0.3834 - acc: 0.8265
```

Kaggle: public: 0.81260

private: 0.81113

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：與上個模型架構差不多，但把 tokenize 的 num_words 設為 100，不然 input 會太大。用 texts_to_matrix 轉成 BOW 的 input。

模型架構：

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 128)	2560000
lstm_1 (LSTM)	(None, 128)	131584
dense_1 (Dense)	(None, 1)	129
Total params: 2,691,713		
Trainable params: 2,691,713		
Non-trainable params: 0		

訓練過程和準確率：

```

200000/200000 [=====] - 506s 3ms/step - loss: 0.6880 - acc: 0.5188
Epoch 2/3
200000/200000 [=====] - 501s 3ms/step - loss: 0.6817 - acc: 0.5380
Epoch 3/3
200000/200000 [=====] - 498s 2ms/step - loss: 0.6798 - acc: 0.5413

```

Kaggle: public: 0.53132

Private: 0.53166

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"
BOW	0.49039897	0.48992708
RNN	0.31660247	0.94093776

因為 **RNN** 有考慮每個句子中的單字順序，**BOW** 只考慮出現的字，沒有順序的差別，所以幾乎同樣的字但順序不同、意思不同的話，**RNN** 比較分辨得出來。

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：

有標點符號：token = Tokenizer(num_words = 5000, filters='\\t\\n')

無標點符號：token = Tokenizer(num_words = 5000)

因為 **filters** 預設為 '!"#\$%&()*+,-./:;<=>?@[\\]^_`{|}~\\t\\n'，所以不設 **filter** 的話就會把大部分的標點符號去掉了。

無標點符號的準確率在 Kaggle 上為：

Public: 0.80464

Private: 0.80539

較有標點符號的 **model** 差，表示句子裡的標點符號也跟情緒有一定的關係。

5. (1%) 請描述在你的 **semi-supervised** 方法是如何標記 **label**，並比較有無 **semi-supervised training** 對準確率的影響。

答：

先 **train** 完第一題的 **RNN** 後，用這個 **model** **predict** "training_nolabel.txt"，得到每個句子的預測值，若預測值大於 0.5，則標記為 1，若小於 0.5，則標記為 0，再用 "training_nolabel.txt" 和新增的標記再丟到同個 **model** **train** 一次。

訓練過程如下：

```
Epoch 1/3  
200000/200000 [=====] - 698s - loss: 0.3541 - acc: 0.8427  
Epoch 2/3  
200000/200000 [=====] - 523s - loss: 0.3417 - acc: 0.8489  
Epoch 3/3  
200000/200000 [=====] - 514s - loss: 0.3278 - acc: 0.8552
```

丟到 Kaggle 上的準確率為：

Public: 0.80882 Private: 0.80905

較第一題 RNN 的結果差，可能是 semi-supervised training 讓原本就 train 不好的情況更嚴重。