Machine Learning HW6 Report

學號：B05901068 系級：電機三 姓名：蕭如芸

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：

使用 gensim 套件中的 Word2Vec 實作。訓練資料使用 training data 和 testing data，詞向量的維度為 250，訓練的回數為 10，以 Skip-gram 來訓練。

```
model = Word2Vec(seg_list, size=250, iter=10, sg=1)
```

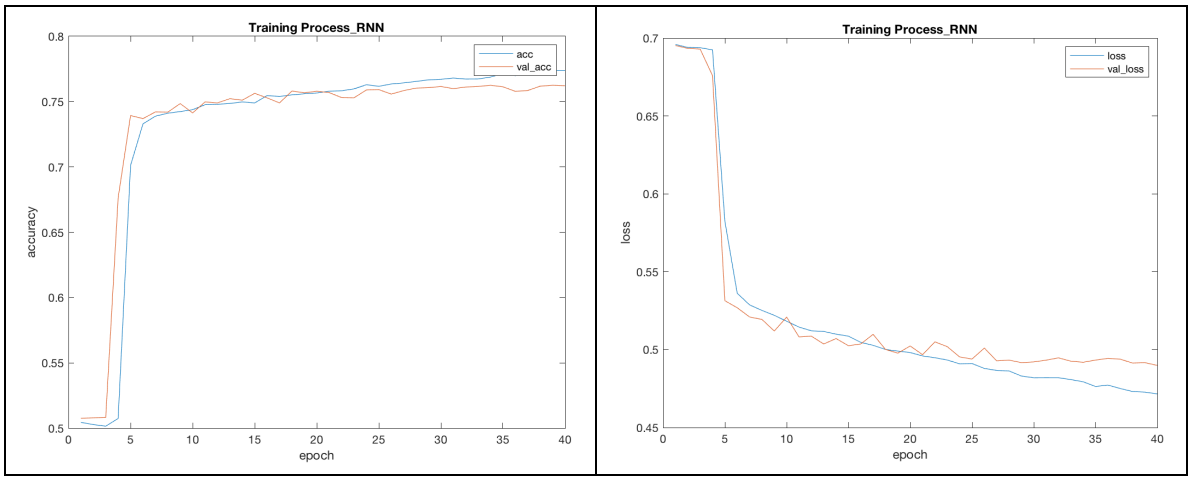
RNN 模型架構：

```
Embedding
Dropout(0.25)
LSTM(units=256, dropout=0.25, recurrent_dropout=0.25,
     activation='sigmoid', inner_activation='hard_sigmoid',
     implementation=2)
Dense(2, activation='softmax')
```

正確率：Public 0.76110, Private 0.75960

Kaggle 上最高的分數是使用三個 model 做 majority vote，三個 model 的架構大致相同，只是改變 dropout rate 和 LSTM 的 units 數目。

訓練曲線：



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*。

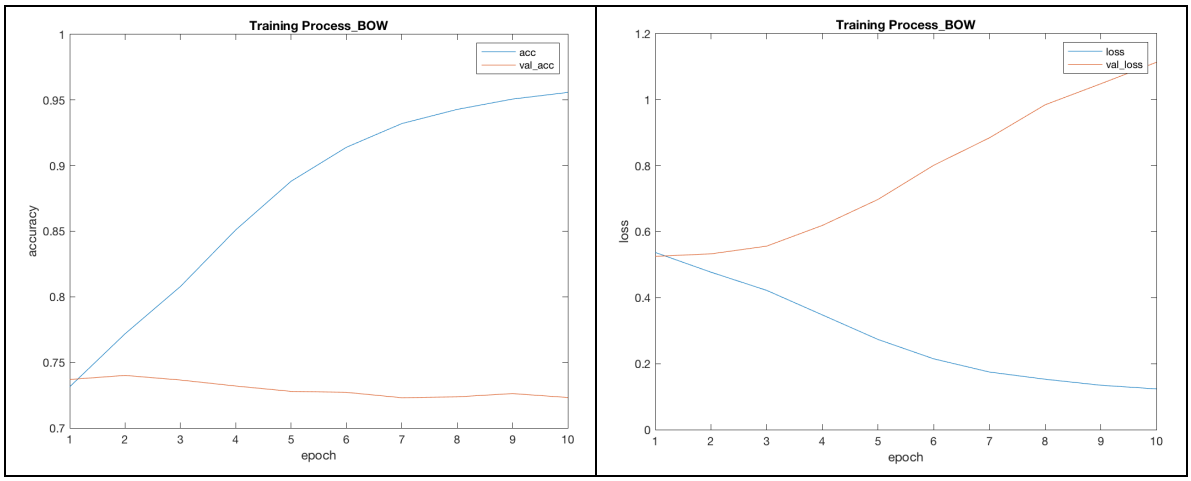
BOW：使用 Keras 的 Tokenizer 實作，取出現次數最多的 5000 個詞作為 bag of words 的 dictionary 中的詞。

DNN 模型架構：

```
Dense(512)
Activation('relu')
Dropout(0.5)
Dense(2)
Activation('softmax')
```

正確率：Public 0.74270, Private 0.74180. 使用 RNN 模型的正確率較高。

訓練曲線：



Validation accuracy 在 epoch 2 最高（0.74008），之後就往下降，而 training accuracy 則持續上升到超過 0.95，很明顯是 overfit，因此取訓練完 2 epoch 為最終的 model。

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。

(1) 訓練 Word2Vec model 時，適當調整參數。使用 size=250 (default: 100)可以更有效的表達詞與詞之間的關係；使用 iter=10 (default: 5)，因為增加詞向量的維度可能需要訓練更多次，而且增加訓練次數可以得到更精準的結果。

(2) 每個句子長短不一，計算後發現，訓練資料斷詞後，最長的句子長度超過一萬，但平均長度只有 33，表示大部分句子並不長，應當設定一個 threshold，大於這

個長度就切掉，小於則補 0。我選擇 threshold = 200，如此訓練資料和測試資料都只有約 1.6%會被切掉。這樣的設計可以增加運算效率，也不會因為太多句子被切掉而影響是否為惡意留言的判斷。

(3) 在 model 中加入 dropout 可以提升正確率，避免 overfitting。

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。

Accuracy	有做斷詞	不做斷詞
RNN	Public: 0.76110, Private: 0.75960	Public: 0.74330, Private: 0.73330
	Average: 0.76035	Average: 0.73830
BOW	Public: 0.74270, Private: 0.74180	Public: 0.74900, Private: 0.74880
	Average: 0.74225	Average: 0.74890

在 RNN，做斷詞對提高正確率幫助很大，因為斷詞有助於語意判斷。而 BOW 不做斷詞的正確率反而還高一些，可能是因為 BOW 只判斷句子的組成有哪些字詞，和語意較無關係。

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。

句子	"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"
斷詞後	['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	[0.42720005, 0.5727999]	[0.40960985, 0.59039015]
BOW	[0.35776016, 0.64223987]	[0.35776016, 0.64223987]

RNN 會考慮詞語的先後順序，根據 model output 的結果，第二句話是惡意留言的機率比較高一些，這個結果是合理的。

BOW 只考慮句子中有哪些詞，不考慮先後順序，這兩個句子斷詞後包含的詞語完全相同，因此 model output 的結果相同。BOW 認為句子的惡意程度高於 RNN，原因可能是 model 看到「白痴」就認為是惡意留言，不考慮前後文。