Machine Learning HW5 Report

學號：B03901015 系級： 電機四 姓名：梅希聖

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: None)

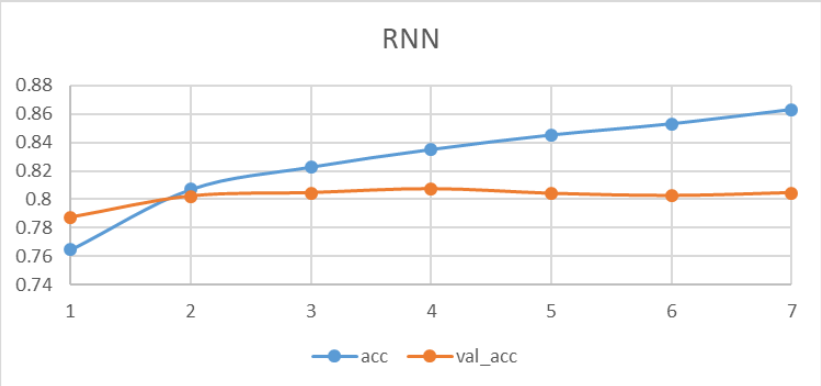
答：

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 40, 50)	9597750
lstm_1 (LSTM)	(None, 40, 256)	314368
lstm_2 (LSTM)	(None, 256)	525312
dense_1 (Dense)	(None, 64)	16448
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 1)	65

Total params: 10,453,943
Trainable params: 10,453,943
Non-trainable params: 0

以 tokenizer 將字串轉為長度為 40 之向量後，通過一層 embedding layer，再通過 2 層 LSTM，再通過一層 fully connected layer 後接一個 sigmoid 的 output。

訓練過程如下，橫軸為 epoch，縱軸為準確率。



在 kaggle 獲得之分數為：0.80909 (private), 0.80573 (public)

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: None)

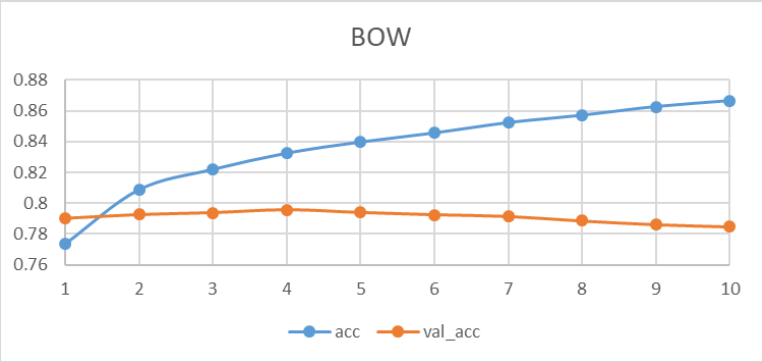
答：

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 64)	640064
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 256)	16640
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 256)	65792
dropout_3 (Dropout)	(None, 256)	0
dense_4 (Dense)	(None, 1)	257

Total params: 722,753
Trainable params: 722,753
Non-trainable params: 0

因記憶體之限制，設定 tokenizer 之 num_words=10000，將字串轉為長度=10000 之向量後進入 DNN 模型訓練，模型由 3 層 relu 的 fully connected layer + dropout 組成，最後通過一個 sigmoid 的 output。

訓練過程如下，橫軸為 epoch，縱軸為準確率。



在 kaggle 獲得之分數為：0.79510 (private), 0.79916 (public)

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: None)
答：

	BOW	RNN
"today is a good day, but it is hot"	0.69430	0.47312
"today is hot, but it is a good day"	0.69430	0.92300

兩句話所包含的字詞均相同，因此在 BOW 中得分也會相同，因 BOW 不考慮字詞之順序，均會把這兩句話分類成 1 之原因，可能是這句話含有 ”good” 的關係。

在 RNN 中，兩句話的分類並不同。從英文文法來解讀，”but”後面代表句子本身之語意，第一句之語氣的確有負面之轉折，而第二句有較正面的語氣，字詞的順序會改變整句話之語意，這應是 RNN model 如此分類的原因。

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: None)
   答：

```
from gensim.parsing.preprocessing import *
def preprocess_docs(docs):
    filters = [lambda x: x.lower(), stem_text, strip_multiple_whitespaces, strip_non_alphanum]
    docs = trim_list(docs)
    tmp = [' '.join(preprocess_string(s, filters=filters)) for s in docs]
    docs = tmp.copy()
    del tmp
    return docs
```

   使用 gensim.parsing.preprocessing 的 strip_non_alphanum，將字母與數字外的符號去除，使用 RNN model 進行訓練。
   在 kaggle 之分數結果如下：

標點符號	Private	Public	Average
有	0.80881	0.80909	0.80895
無	0.80471	0.80573	0.80522

   兩者之準確率差不多，但有標點符號之準確率高一點點，可能因為標點符號本身也會夾帶語意的成分，如問號、驚嘆號等等，進而提高了準確率。

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: None)
   答：
   使用 self-training 的方法，model 經過 labeled data 訓練完成後，預測 unlabeled data 所屬之類別，若預測結果<0.2 則將資料標記為 0，>0.8 則標記為 1，再將標記完成之資料加入 labeled data 一起訓練，步驟重複 5 次。
   在 kaggle 之分數結果如下：

	Private	Public	Average
Only labeled data	0.80881	0.80909	0.80895
Semi-supervised	0.81558	0.81655	0.81608

經過 semi-supervised 的訓練後，準確率有所提高。好的 data 愈多，的確會讓訓練的結果變好。