ML2017FALL HW4 Report

學號：B04902090 系級： 資工三 姓名：施長元

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: b04902089 林政豪，b04902105 戴培倫，兩人提供訓練意見)

答：

- ✓ 所有的 data 都前置處理好(分開 label 以及 data，或是把資料讀近來)
- ✓ 把資料轉成 acsii 去除奇怪的字元
- ✓ keras preprocess 中的 text to word sequence，
  - ➢ 把句子轉成 word sequence、去除標點符號以及數字、全部字母轉為小寫
- ✓ itertools 中的 groupby，去除重複字元(ex. Wooooow -> wow)
- ✓ 將這些 (labeled、unlabeled、testing data) 都丟到 gensim 中 train
  - ✧ 100 維度、window 為 3、字數小於 3 的不放入 train。
- ✓ 利用適才 train 好的 gensim model，得到 word2idx 以及 idx2vec
- ✓ Sequential()
- ✓

```
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 40, 100)           6855900
_________________________________________________________________
bidirectional_1 (Bidirection (None, 40, 512)           731136
_________________________________________________________________
bidirectional_2 (Bidirection (None, 256)               656384
_________________________________________________________________
dense_1 (Dense)              (None, 2048)              526336
_________________________________________________________________
dropout_1 (Dropout)          (None, 2048)              0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 2049
=================================================================
Total params: 8,771,805
Trainable params: 8,771,805
Non-trainable params: 0
_________________________________________________________________
```

- ✓ Embedding 使用剛剛得到的 idx2vec，並設定為 Trainable，bidirectional 中放入 LSTM 分別為 256、128，activation 使用 tanh，drpoout 以及 recurrent dropout 皆為 0.4
- ✓ 接入 Dense 層 2048，activation 使用 relu
- ✓ Activation 為 sigmoid 的輸出層
- ✓ model.compile(loss='binary_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
- ✓ 訓練 6 個 epoch，batch_size 為 256
- ✓ 最終上傳 kaggle 的準確率為 Private : 0.82317, Public : 0.82455
- ✓ 最後忘記改之前勾選的，所以成績比這個爛 QQ

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: b04902099 黃嵩仁 提供方向，B04902021 陳弘梵 建議字典大小)
答：
   - ✓ 讀入資料
   - ✓ 使用 tokenizer，字典數量為 10000，filter 是標點符號
     - ■ (電腦只能接受這個大小 11000 就會 memory error)
   - ✓ Fit_on_text (labeled train data 以及 testing data)
   - ✓ 取出所有資料的空行
   - ✓ texts_to_matrix(mode='count')
   - ✓ sequential()

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 800)               8000800
_________________________________________________________________
dropout_1 (Dropout)          (None, 800)               0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               410112
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 513
=================================================================
Total params: 8,411,425
Trainable params: 8,411,425
Non-trainable params: 0
_________________________________________________________________
```

   - ✓ 沒有 embedding，也沒有 RNN，直接使用 DNN，並把 params 湊齊
   - ✓ 第二個 epoch 就開始 overfitting 了
   - ✓ 最後上傳 kaggle 的成績為 Private : 0.79852, Public : 0.79852

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: Myself)
答：

	Bag of words	RNN
today is a good day, but it is hot	0.57281137	0.42311069
today is hot, but it is a good day	0.57281137	0.86828786

因為 Bag of word 並沒有考慮語序性，單就句子裡有的單字去進行 Training；而 RNN 有考慮語序性，LSTM 具有記憶性，懂得考慮相同單字不同語序造成的情緒變化，所以很明顯的上面那句是稍微負面情緒，下面那句是幾乎正向。

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：
   without 標點符號 : Private : 0.82317, Public : 0.82455
   with 標點符號 :Private : 0.82107 , Public : 0.82318
   標點符號似乎給予些許阻礙，推測是許多標點可能不具有情緒意義，但 word2vec 將其視為一個元素，造成訓練上的負擔以及誤判。

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: 助教 PPT)
   答：為了降低偏誤，我設定 pos_threshold 為 0.7，使用原先訓練完成的 model 對 unlabeled data 進行 predict，之後將兩筆資料以及 label 合成，再次進行訓練。
   semi-supervised : Private : 0.80842, Public : 0.80998
   Only labeled data : Private : 0.80839, Public : 0.80905
   //(這並不是使用最好一次的 model，因為最好一次 train 是 deadline 之前出來的，但 semi 是在之前 train 出來的，因為手邊資源不足沒辦法在 github deadline 之前再 train 一次 semi，懇請助教同情沒有資源的同學 QQ)
   可以發現到，semi-supervised 提升了一點點，但就只有那麼一點點
   推測是 0.7 還是太寬鬆了，可以考慮提高 barrier。