HW4 Report

學號：R05945012　系級：生醫電資碩二　姓名：張凱崴

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？

模型架構：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 128)         3132800
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               131584
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 3,281,025
Trainable params: 3,281,025
Non-trainable params: 0
```

由上圖所示，我的 RNN model 非常單純，由一層 embedding layer、一層 LSTM 以及兩層 DNN 所組成。Optimizer 為 adam，loss function 為 binary crossentropy。

訓練過程：

(1) Embedding layer

我使用 gensim 的 Word2Vec 將 training 和 testing data 建構成一個 24474 種單字的字典 (出現頻率大於 5 才列入)，再藉由這個字典將 training data 轉成 index sequence (未包含在字典內的單字為零)，pad sequence(補 0，maxlen 為 40)後拿去訓練，gensim 產生的 embedding matrix 則作為 embedding layer 的初始 weights (24475 x 128)。此外，實驗後發現 mask_zero=False、trainable=True 會有較好的結果(不把 index 0 消去，且僅將 gensim 訓練出的 embedding 作為初始值而非固定)。

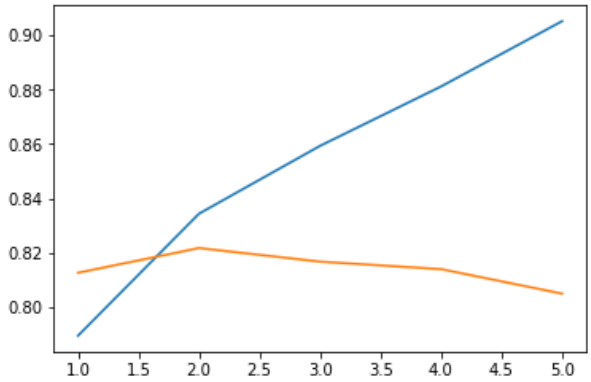
(2) LSTM

LSTM unit 的個數與 word vector dimension 相同為 128，dropout 與 recurrent dropout 經實驗後發現都設為 0，會產生最佳的模型。雖然此舉會使 training accuracy 過度上升造成 overfitting，但 valid accuracy 曾經到達的最高值卻會是最好的 (如圖一)，且即便調高 dropout rate 會減少 overfitting 的情況，但實際上也無法讓 valid accuracy 增加。

(3) DNN

實驗後發現無論是增加層數或 cell 的個數，都無法讓 valid accuracy 有所上升，最簡單的 model (一層接 LSTM，一層接 binary output)即可有最好的結果。Dropout rate 與 LSTM 層有相同的結論，因此設為 0。

準確率： (橫軸：epoch、縱軸：accuracy、藍：training、橘：valid)



At epoch 2: Training accuracy = 0.8344, Valid accuracy = 0.8218

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？

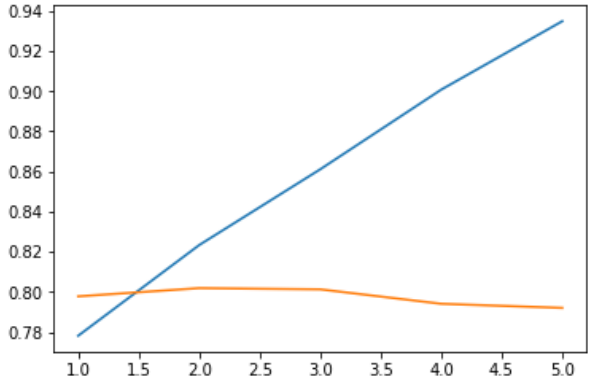
模型架構： (為了與 RNN model 做比較，DNN 的設定與其相同。)

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 128)               3132928
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 3,133,057
Trainable params: 3,133,057
Non-trainable params: 0
```

訓練過程：

與 Q1 使用相同的字典，並將 training data 轉成 24475 維的 BOW 形式，為避免 Memory Error 的問題，我將 data type 設為 uint8 (在一句話中，同個單字不太可能出現 256 次以上)。與 Q1 相同，增加層數與 cell 個數，以及提高 dropout rate 都沒有太好的效果。

準確率： (橫軸：epoch、縱軸：accuracy、藍：training、橘：valid)



At epoch 2: Training accuracy = 0.8233, Valid accuracy = 0.8018

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。

	RNN (Q1)	RNN (Q5, best)	BOW
第一句	0.522654	0.132625	0.642455
第二句	0.990871	0.621687	0.642455

如上表所示，RNN model 對於兩句有不同的評分，但 BOW 卻相同，因為對 BOW 而言，兩句是完全一樣的。另外，我也比較了 Q1 (無 semi)與 Q5 (有 semi) 兩種 RNN model，對於沒使用 semi-supervised 的模型，兩句都是正面的，但第一句很接近負面，而有用 semi 的模型則可以分出第一句是負面且第二句是正面的，準確率較高。

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。

助教的範例 code 中使用 Keras sequence processing 的 filter 將標點符號去除，然而我發現在 data 中有些除了標點符號之外的亂碼存在，例如 ð ´ ñ ‹ ðº ð½ðµ ð ±ðµñ€ñƒñ，應該也必須濾掉。因此與其設定那些字元該濾除，我使用 re 將我要訓練的字元「包含」在內。如下：

(1) 有標點符號：

```
re.findall(r'[A-Za-z0-9,.:;?!\'"()\[\]{}<>*/+-=~_#$%&]+',text[1])
```

(2) 無標點符號：

```
re.findall(r'[A-Za-z0-9]+',text[1])
```

準確率：

	有標點符號	無標點符號
Valid acc.	0.827	0.814
Public score	0.82115	0.81402
Private score	0.81926	0.81381

有標點符號的準確率較高，因為標點符號可能包含了情緒的資訊。

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。

我將 training、testing 和 unlabeled data 共同由 gensim 建出一個新的字典，並用與 Q1 相同的方法做訓練，先用 training data 訓練出一個暫時的模型 (下表的 no semi)，接著再用此模型對 unlabeled data 做預測，得到的結果根據 threshold 賦予 0 和 1 (value > threshold → 1, value < 1-threshold → 0)，最後將被賦予 label 的部分 unlabeled data 加入 training data，訓練出新的模型。

對於不同 threshold 而產生的模型的準確率，列於下表：

	No semi	thr=0.9	thr=0.8	thr=0.7	thr=0.6
Valid acc.	0.822	0.956	0.965	0.967	0.961
Public score	0.82333	0.82411	0.82637	0.82561	0.82522
Private score	0.82091	0.82173	0.82319	0.82257	0.82193

可以很明顯地看出來，有用 semi 的模型準確率都比較高，且 threshold 的設定與準確度也有關。根據 public score 與 private score 的結果，我將 threshold=0.8 選為我的 best model。