Homework 5 - Text Sentiment Classification

學號：b05901033 系級： 電機二 姓名：莊永松

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？(no Collaborators)

word embedding 部分使用 gensim，把label data跟no label data都丟進去train好word2vec之後才開始弄RNN[1][2]，可避免只使用keras的embedding層而沒有把word2vec訓練完全的缺點。另外對於出現次數過少而被word2vec淘汰的字，之後在train rnn時會找不到，對這些字我有用regex將他們經過一些處理之後再從word2vec的model裡再次尋找，處理方法有二：1. 外國人打字常常故意疊字，如great變greeeaaaat，因此我用regex把這些找不到的字中間的疊字都簡化成只有一個。2. 有些 typo 如"o"打成"0"，就直接用regex取代，如此一來大概可以救回5000個找不到的字，而原本找不到的字有約20000個。

模型架構 RNN 架構很簡單(使用softmax是因為我把output弄成兩個class)

<Input>
[Embedding](use gensim model)
Spatial_Dropout1D(0.4)
LSTM(輸出維度=196)
<Output> Dense(2,activation='softmax'))

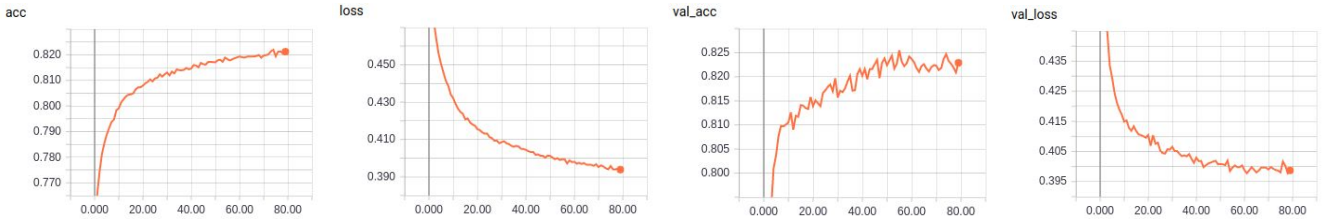
```
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 250)         13944250
_________________________________________________________________
spatial_dropout1d_1 (Spatial (None, None, 250)         0
_________________________________________________________________
lstm_1 (LSTM)                (None, 196)               350448
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 394
=================================================================
Total params: 14,295,092
Trainable params: 350,842
Non-trainable params: 13,944,250
```

其中LSTM的參數為：

dropout=0.2, recurrent_dropout=0.2,
activation='sigmoid', inner_activation='hard_sigmoid'

訓練過程 初始化使用Xavier uniform initializer，batch size設256，epoch數設40，另有使用Early Stopping，訓練過程如下

training data acc　training data loss　validation acc　validation loss



準確率 在kaggle上public score為**0.82671**，private score為**0.82343**，後來將我所訓練出最好的三個model做ensemble後，在kaggle上public score為**0.82966**，private score最佳則為**0.82647**

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？(no Collab.)

模型架構 先將training data使用 keras 的 Tokenizer 處理，取出現頻率大的前5000字來做BOW。模型架構非常簡單(使用softmax是因為我把output弄成兩個class)

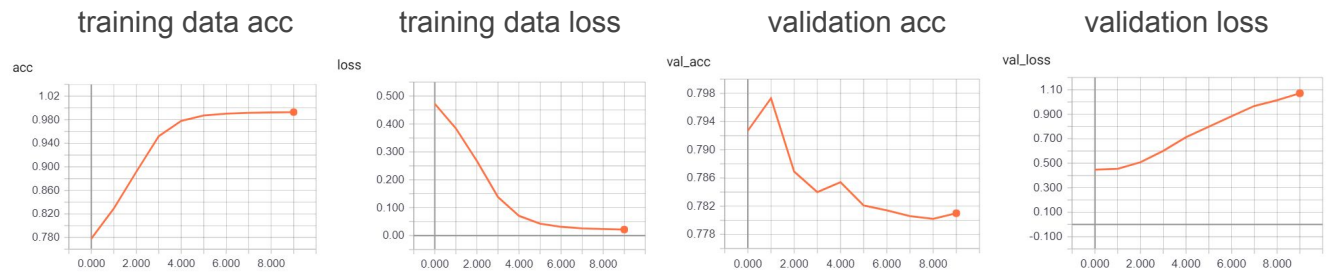
因為我認為都已經使用BOW模型了，只需要訓練出每個字對於positive及negative的影響程度多寡的一一對應關係即可，因此只要架一層就好了，另一方面bow還蠻吃記憶體的，少層一點也好XD

<Input>
Dense(input_dim=5000,units=1000)+ReLU
Dense(2)+Softmax
<Output>

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 1000)              5001000
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 2002
=================================================================
Total params: 5,003,002
Trainable params: 5,003,002
Non-trainable params: 0
```

訓練過程 初始化使用Xavier uniform initializer，batch size設256，epoch數設1而已，不然會overfitting，loss function是cross entropy，optimizer使用Adam

以下訓練過程是epoch數設10的狀況(不然設1畫不出圖XD)，因為validation準確率一直下降，後來只用了第一個epoch時存下的model。



準確率 在kaggle上public score為0.79310，private score為0.79324

雖然準確率比RNN略差，但就模型大小及複雜度而言，已經是個效率不錯的方法，其實準確率也才低RNN 3%左右而已。

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。(no Collab.)

BOW 預測結果皆為=>0.628機率是positive，0.371機率是negative

兩句話預測結果是一樣的，因為BOW只統計字的出現次數不考慮次序。

RNN 預測結果第一句=>0.605機率是positive，0.395機率是negative

第二句=>0.979機率是positive， 0.020機率是negative

討論 第一句話其實沒有那麼正面，因為結尾是 "but it is hot" ，而第二句話正面許多，因為重點是"but it is a good day"。

=>對BOW來說，不考慮次序所以都沒差，兩句話都一樣直接判斷為正面**(62.8%)**。

=>對RNN來說，第一句話他雖然也是判斷為正面**(60.5%)**，但就沒那麼確定。

而第二句話他就非常確定是正面**(97.9%)**。

可以看出RNN的確有憑藉字詞前後順序來作為判斷的依據，BOW則不行。

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。(no Collab.)

我將`!,.:;?'"`等標點符號用regex濾掉之後再train RNN，訓練過程除了收斂速度較慢之外。最後train出來的準確率也下降了1%左右，比較如下：

對準確率的影響	public score	private score
有含標點符號	0.82413	0.82250
未含標點符號	0.81547	0.81385

可以推論，標點符號對於語意還是有一定的影響，不能隨便拿掉。

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。(no Collab.)

我試了兩種標記label的方法，第一種是拿一個表現最好的model，predict之後取大於threshold=0.8 or 小於threshold=0.2的data，拿進來做self-learning

第二種方法是拿三個model，各自predict，三個結果都同樣>0.5或是同樣<0.5的data，取來做self-learning(threshold不設高一點是因為如果再設threshold會幾乎取不到data)

對準確率的影響	public score	private score
未做unsupervised	0.82413	0.82250
第一種方法	0.82329	0.82143
第二種方法	0.82386	0.82129

結果發現，正確率並沒有明顯的增加，推測原因可能是，標記label的方法仍然有偏差，導致被分到該類別的data不一定真的是那個類別，所以訓練效果不好。另外，這次的unlabel data裡面仍有一些是比較髒的資料，也可能因此被分到錯誤的類別而影響訓練結果。

Reference:

[1]: https://gist.github.com/codekansas/15b3c2a2e9bc7a3c345138a32e029969

[2]: https://eliyar.biz/using-pre-trained-gensim-word2vector-in-a-keras-model-and-visualizing/