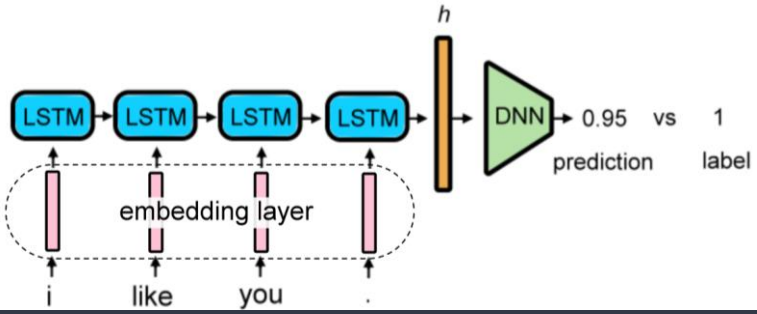
學號：R06946003 系級： 資料科學碩一 姓名：湯忠憲

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？

答：



Layer (type)	Output Shape	Param #
input_22 (InputLayer)	(None, 39)	0
embedding_1 (Embedding)	(None, 39, 300)	55285500
bidirectional_43 (Bidirectio	(None, 39, 1024)	3330048
bidirectional_44 (Bidirectio	(None, 512)	2623488
dense_53 (Dense)	(None, 256)	131328
dropout_32 (Dropout)	(None, 256)	0
dense_54 (Dense)	(None, 128)	32896
dropout_33 (Dropout)	(None, 128)	0
dense_55 (Dense)	(None, 64)	8256
dropout_34 (Dropout)	(None, 64)	0
dense_56 (Dense)	(None, 2)	130

Total params: 61,411,646
Trainable params: 6,126,146
Non-trainable params: 55,285,500

圖一、RNN model 架構

<u>預處理:</u>

(1) 把 n‘t 三個字元黏起來，變成 nt，所以 don‘t 就會變成 dont。

(2) 數字都變成 1，減少 token 數量。8 變成 1，9942 變成 1111。

(3) 將連續出現的三個字母變成單一字母，例如:daaammmn 變成 damn。

(4) 標點符號也做上述處理

(5) 用 gensim.parsing.porter.PorterStemmer()做 stemming，目的也是為了在保留完整的資訊下，減少 token 數。

模型架構:

圖一是 Best model 的 RNN 架構，主要是用 gensim word2vec 提供 pre-trained 好的 weights (100 維)給 embedding layer。在不刪減標點符號，並做預處理之後，剩下的 token 數量為 182733 個，所以 embedding matrix 的 shape 是(182733, 100)。接著是兩層的 Bidirectional LSTM，output dimension 分別為 512 和 256 (若使用 256 和 128 效果也差不多)。兩層 LSTM 使用的 dropout rate 分別為 0.2 和 0.35，kernel_initializer 是 he_uniform，用 tanh activation function。最後再接三層 Dense layers 和 dropout，透過 sigmoid 函數激活，並用 softmax function 輸出預測，這部份對於整體準確度的影響較小。總共參數約六千多萬個，其中有五千五百萬是 Embedding 中 non-trainable 參數。

訓練過程:

loss function 使用 categorical cross entropy，optimizer 使用 Adam。

Epoch 數量的部份不超過八個，一個 epoch 約跑十分鐘 (看完全部 training data，約 18 萬筆)，同時利用 validation set 觀測最佳的 epoch，並使用該 epoch 所存取的 model。

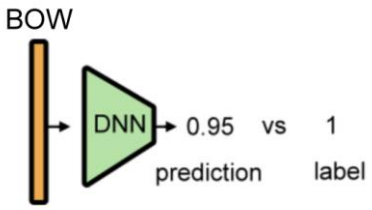
準確率:

單一 model 可以到 0.8279。若利用 data bagging 的方式訓練多個模型，最終的 ensemble model (10 個) 在 Kaggle 上的準確率可以上 0.8367，但是要 predict 非常久。

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？

答：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 1024)              20481024
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 2050
=================================================================
Total params: 20,483,074
Trainable params: 20,483,074
Non-trainable params: 0
_________________________________________________________________
```



圖二、BOW model 架構

模型架構:

先將句子利用 keras 中的 tokenizer.texts_to_matrix，mode 調成 count，將一個句子變成詞袋向量。我設的維度是 20000 維，也就是只看前 20000 個出現頻率高的字彙做成向量。因此模型的 input shape 就是(200000,20000)。架構的部分使用兩層 Dense layer，第一層為輸出 1024 維，第二層用 softmax function 輸出 2 維預測值。

訓練過程:

使用 cross entropy 做為 loss function，optimizer 使用 Adam。這個模型大概 2、3 個 epochs 就可以收斂了。

準確度:

約為 0.7939，表現較 RNN 模型差一點，但還是有一定的水平。

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。

答：

```
['todai is a good day, but it is hot', 'todai is hot, but it is a good dai']

 array([[ 0.19574326,  0.80425668],
        [ 0.17650943,  0.82349062]], dtype=float32)
```

圖三、 Bag of word model 預測

```
['todai is a good day, but it is hot', 'todai is hot, but it is a good dai']

 array([[ 0.53794986,  0.4620502 ],
        [ 0.00855446,  0.99144554]], dtype=float32)
```

圖四、RNN model 預測

圖三與圖四分別顯示兩個 model 對句子預測的機率分布。我們可以發現 Bag of word model 將兩者都歸類為正面情緒，但是 RNN model 將第一個句子預測成負面 (雖然沒有很高的信心)，第二個預測成正面。會造成這樣的差異主要是因為 Bag of word model 只有針對字數做統計的特徵，因此相同詞彙組成的兩個句子會有相似的結果。而 “good” 應該算是個蠻強的特徵詞彙，因此 BOW model 將他預測為正面。RNN model 因為有考量到詞彙的前後關係，所以會對“but”這類的轉折語起反應，可以視為 model 有學到句子 semantic 的例子。

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。

答：

tokenize 的方式可藉由 Keras 的 Tokenizer(filters = …)調整。若不包含標點符號，cross-validation 的準確率約 0.81625，包含的話則是 0.82395。可以發現有標點符號可以有較好的準確率，相差約 1%以內。標點符號在這個 dataset 還算蠻重要的特徵，例如有人會使用連續的驚嘆號!!!!!!，或者是?!表達較激動的語意。若是有處理好我覺得應該是可以提升準確率。

5. (1%) 請描述在你的 semi-supervised 方法是如何標記 label，並比較有無 semi-supervised training 對準確率的影響。

答：

Semi-supervised 的部分我實作的是 Self-training。把 train 好的 7 個 model 們 ensemble 對 un-label data 做預測，並將這些預測後的值轉成該比 un-label data 的 label。最後將這些 data 視為 label 好的，與原本的 label data 一起 train，可以不斷循環。我設的 threshold 是 0.97，可以多出 34 萬的訓練資料，總共 54 萬多比訓練資料。準確率約為 0.82618，進步的空間沒有想像中的大。若再反覆做也無法有顯著的提升。