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: b03902002陳家陞建議使用 bidirection。

分成兩個部分解釋。

一、Word Embedding

我表現最好的模型是將所有資料 (train_label, train_nolabel, test) 都以空格斷詞、去掉數字後（保留標點符號），再用 gensim 的 word2vec train出 embedding，word2vec 採用 skip_gram，其他重要參數為 size=200, window=10, min_count=2（剩下都是 default）。之所以要設 min_count=2 是因為這次的資料集中很多火星文，有約一半的 words 只出現1次。

二、NN

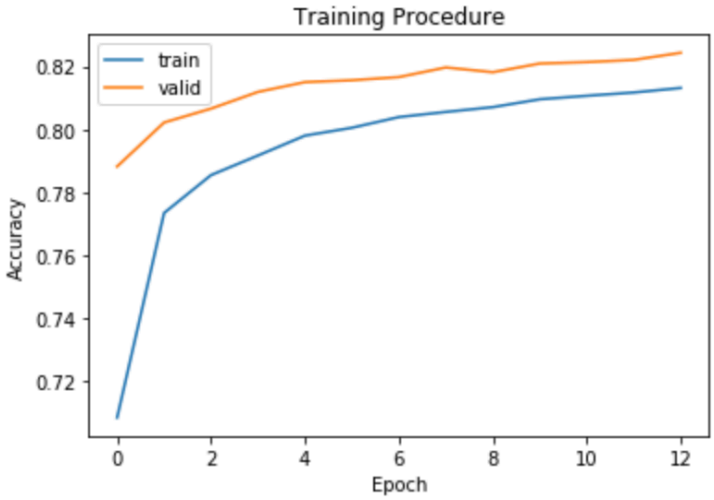
我先自行把 training data 轉成 word vector，所以就不用再加 embedding layer。剩餘 structure如下：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_3 (Bidirection (None, 36, 256)           252672
_________________________________________________________________
bidirectional_4 (Bidirection (None, 256)               295680
_________________________________________________________________
dense_13 (Dense)             (None, 256)               65792
_________________________________________________________________
batch_normalization_10 (Batc (None, 256)               1024
_________________________________________________________________
leaky_re_lu_10 (LeakyReLU)   (None, 256)               0
_________________________________________________________________
dropout_10 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_14 (Dense)             (None, 128)               32896
_________________________________________________________________
batch_normalization_11 (Batc (None, 128)               512
_________________________________________________________________
leaky_re_lu_11 (LeakyReLU)   (None, 128)               0
_________________________________________________________________
dropout_11 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_15 (Dense)             (None, 64)                8256
_________________________________________________________________
batch_normalization_12 (Batc (None, 64)                256
_________________________________________________________________
leaky_re_lu_12 (LeakyReLU)   (None, 64)                0
_________________________________________________________________
dropout_12 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_16 (Dense)             (None, 2)                 130
```

首先是兩層的雙向 GRU，units = 128, activation = ‘tanh’, dropout_rate = 0.5，然後接三層FC，皆有BN, LeakyReLU, Dropout layer，units分別為 256, 128, 64，最後再接一層 units=2, activation=‘softmax’ 的 FC，optimizer= rmsprop (lr = 0.001), loss=‘categorical_crossentropy’。

以下是訓練過程與準確率：

Data	Accuracy
Train	0.81
Valid	0.822
Test	0.823



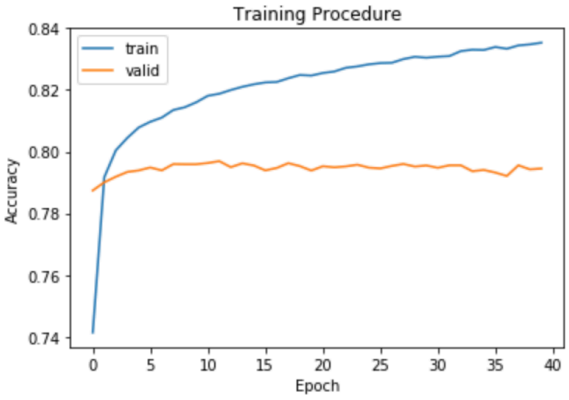
可以發現 RNN 雖然 train 很慢，但是收斂得挺快，其實 5 個 epoch 就差不多了，但我後來繼續 train 到 17。另外，Train 總是表現比 Valid 差一些，推測是因為 Training data 遠多於 Valid，然後我的 Valid 恰好相對較容易預測（Valid 是直接取 train_label 的後 2w 筆）。

2.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: 無。

我的 BOW feature 是一樣是看完所有 data 後建 dictionary（保留標點符號），然後保留出現次數大於 76 次的 words，這樣會剩 10021 個 words，使得每個 sentence 的 feature 是 10022 維（含 unknown）各 sentence 的 feature 就是用 word count 來決定。
NN的架構、訓練過程與準確率如下：

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 256)	2565888
batch_normalization_1 (Batch	(None, 256)	1024
leaky_re_lu_1 (LeakyReLU)	(None, 256)	0
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 256)	65792
batch_normalization_2 (Batch	(None, 256)	1024
leaky_re_lu_2 (LeakyReLU)	(None, 256)	0
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 128)	32896
batch_normalization_3 (Batch	(None, 128)	512
leaky_re_lu_3 (LeakyReLU)	(None, 128)	0
dropout_3 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 64)	8256
batch_normalization_4 (Batch	(None, 64)	256
leaky_re_lu_4 (LeakyReLU)	(None, 64)	0
dropout_4 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 2)	130



Data	Accuracy
Train	0.816
Valid	0.796
Test	0.798

DNN 架構就是比 RNN 時的 FC 部分再多了一層 units=256 的 FC。ACC 是取第 10 個 epoch 的 model 計算。BOW 的表現顯然會比較差，比較令我意外的是收斂速度也很快。

3.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: 無。

	BOW	RNN
Sentence 1	[0.35, 0.65]	[0.91, 0.09]
Sentence 2	[0.35, 0.65]	[0.03, 0.97]

右表中 [x, y] 分別表示預測為 class0, 1（負面, 正面）的機率。可以發現兩點：(1) BOW 對兩個句子預測的分數相同，這是因為 BOW 不會考慮句子中 words 出現的順序，因此表示兩個句子的向量是完全相同的，預測結果自然也相同。至於會被預測為正面我推測是因為 good 比 hot 的情緒更強烈。至於 RNN 因為會考慮先後關係，預測結果自然不同。(2) RNN 較能給出正確預測，可以發現對兩句它都很有信心是正或負面。推測是它有學到考慮先後關係時，這兩個句子的情緒就會很明顯。（正 but 負 以及 負 but 正）。

4.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: 無。

包含標點符號者在第一題已經討論過。右下是沒包含標點的結果：

Data	Accuracy
Train	0.808
Valid	0.818
Test	0.818

兩者 train 的 epoch 皆為 12 個，可以發現沒有包含標點符號時，準確率會略低一點。這其實滿令我意外的，也許標點符號對這個 work 的確是有所幫助，

5.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: 無。

由於 nolablel data 數量很多，我一次僅處理約 1/6 的 nolabel，每次從在任一 class prob>0.75 的資料中隨機取出 50000 筆，跟 label data 一起 train 2 個 epoch，重複 6 次後掃完所有 nolabel data。準確率如右下：

Data	Accuracy
Train	0.813
Valid	0.819
Test	0.818

之所以設定 0.75 的 threshold 是基於一個觀察，我試過取 prob>0.9或0.98等，並且只再 train 在 pseudo label 上，只要這樣 train 1-2 個 epoch 就會使得模型變得很極端（都給出超高機率），推測是因為機率超過 0.9 的本來就是一些很簡單的 case（例如包含一些明顯關鍵字）再抓他們進去 train 會使模型變成看到關鍵字就高潮。每次只隨機取 50000 筆則是怕 pseudo label 太多 bias噴發。