學號：R06922002 系級：資工碩一 姓名：姚嘉昇

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: R06942082 黃釋平、R06942054 潘仁傑)

答：先用 gensim 拿所有 data(train_label, train_nolabel, test)去 train 一個 word2vec 的 model，之後把 training data 的每個 word 都轉換成 vector，每一句話都變成(39, 100)的 vector，其中 39 是一句話最長的長度，100 是 word2vec 的維度。這樣轉換完之後，就可以直接丟給 RNN 了，而我這次是使用 GRU+Dense，還沒做 semi 就可以通過 strong baseline 了。

模型架構：

```
175     print('Building model...')
176     model = Sequential()
177     model.add(GRU(units=128, input_shape=(MAX_SEQUENCE_LENGTH, EMBEDDING_DIM),
178                   dropout=0.1, recurrent_dropout=0.1))
179     model.add(Dense(units=256, kernel_initializer='glorot_normal'))
180     model.add(BatchNormalization())
181     model.add(Activation('relu'))
182     model.add(Dropout(0.2))
183     model.add(Dense(numClasses, activation = 'softmax'))
184     print ('Built the model.')
185     print (model.summary())
```

```
Layer (type)                 Output Shape              Param #
=================================================================
gru_1 (GRU)                  (None, 128)               87936
_________________________________________________________________
dense_1 (Dense)              (None, 256)               33024
_________________________________________________________________
batch_normalization_1 (Batch (None, 256)               1024
_________________________________________________________________
activation_1 (Activation)    (None, 256)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 514
=================================================================
Total params: 122,498
Trainable params: 121,986
Non-trainable params: 512
```

訓練過程：Epochs=16, optimizer='adam', loss function='categorical_crossentropy'

準確率：public score=0.82204, private score=0.82229

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:)

答：BOW 最多儲存 16384 個字，然後就丟給 Fully connected classifier 了。而我在做 BOW 的時候，因為是使用 binary 的模式，或許改成 count 準確率會更好。

```
44      print ('Vectorizing sequence data...')
45      tokenizer = Tokenizer(num_words=16384)
46      tokenizer.fit_on_texts(xTrain)
47      xTrain = tokenizer.texts_to_sequences(xTrain)
48      xTrain = tokenizer.sequences_to_matrix(xTrain, mode='binary')
49      print('xTrain shape:', xTrain.shape)
50
51      print('Building model...')
52      # Create the model.
53      model = Sequential()
54      # Fully-connected classifier.
55      model.add(Dense(units=64, input_dim=xTrain.shape[1]))
56      # model.add(BatchNormalization())
57      model.add(Activation('relu'))
58      model.add(Dropout(0.5))
59      model.add(Dense(numClasses, activation = 'softmax'))
60      print ('Created the model.')
61      print (model.summary())
```

模型架構：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 64)                1048640
_________________________________________________________________
activation_1 (Activation)    (None, 64)                0
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 130
=================================================================
Total params: 1,048,770
Trainable params: 1,048,770
Non-trainable params: 0
```

訓練過程：Epochs=5, optimizer=’adam’, loss function=’categorical_crossentropy’

準確率：public score=0.55649, private score=0.55893

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:)

答：

BOW 只計算一句話每種字出現的次數，並不考慮每種字出現的順序，所以這兩句話雖然字的排列順序不同，但是每種字出現的次數一模一樣，所以兩句話丟進 BOW 模型後出來的情緒分數會一樣。

而 RNN 除了讀取字，還會記錄上一次讀了哪些字，所以字的順序會對最後情緒分數的輸出有很大的影響，以至於兩句話隨然字都一樣，但是意思卻不同。

BOW 的情緒分數

class	0	1
today is a good day, but it is hot	0.48462033	0.51537961
today is hot, but it is a good day	0.48462033	0.51537961

RNN 的情緒分數

class	0	1
today is a good day, but it is hot	0.93877846	0.06122153
today is hot, but it is a good day	0.02313557	0.97686440

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:)

答：

下圖是我去除標點符號的過濾器，我認為去除標點符號之後準確率會上升的原因是，還沒去除標點符號前，「Hi,I'm good」這句話有可能會被切成「Hi,I'm」「good」或者「Hi,」「I'm」「good」，導致我在建立 word2vec 的 model 時，明明是字典裡有的字，卻會有查不到這個字的問題，因為標點符號都被黏進去前後的字了。而尤其是 testing data 的每句話最後面都有\n，若不去掉的話，testing data 每句話的最後一個字都變成「good\n」，字典中很難查詢到這個字！所以若是先把標點符號取代成空格，將字與標點符號分開，就可以避免查不到字的問題了！

表、有無標點符號的準確率影響

	Public score	Private score
包含標點符號	0.79158	0.78875
去除標點符號	0.82229	0.82204

```
# Characters filters.
filters = '!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~\t\n'
for c in filters:
    item = item.replace(c, ' ')
```

圖、標點符號的過濾器

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:)

答：

我在做 label 的時候，有用 0.7 跟 0.95 當作 threshold，但是 0.95 的效果比較好一些，雖然說 0.70 得到的 label data 數量較多，但是有可能有些是 label 錯的資料，所以得到了一堆不是很乾淨的資料，可能會導致整個 model train 歪掉，也就是 model 一直學到不對的東西，最後可能會學壞。

而使用 0.95 的話，比較能確保之後要繼續 train 的 data 是比較可以信任的，雖然說拿到的 label data 比較少，但是可以透過重新再做一次 semi-supervised 的過程，再重新 label 一次，有可能經過多次來回 label 後，得到的 label data 也會比開 0.7 做一次還多，這樣雖然要來來回回 train 比較多次，但可以確保整個 model 不會 train 歪掉。

Threshold	Label data amount	Public score	Private score
No semi-supervised	200000	0.81598	0.81697
0.70	200000+908414	0.81913	0.81812
0.95	200000+588769	0.82026	0.81958