學號：B04902097 系級： 資工二 姓名：陳家棋

1. (1%) 請問 softmax 適不適合作為本次作業的 output layer ? 寫出你最後選擇的 output layer 並說明理由。

softmax 不適合作為本次作業的 output layer ，因為 softmax 會讓最後一層的所有輸出的值界在 0 - 1 之間，且他們的總和為 1，因此在 label 數量有多有少的情況下，最後被 train 出來 minimize loss 的值會因為 label 數量而有不同，要找到「一個」適當 threshold 去取答案顯得不可行。故我最後挑選 sigmoid 作為 output layer，讓 38 個輸出的值都獨立的介於 0 - 1 之間。

```
model = Sequential()

model.add( Dense(input_dim = x_train.shape[1], output_dim = 512, activation='elu') )
model.add( Dropout(0.4) )
model.add( Dense(512, activation='elu') )
model.add( Dropout(0.4) )
model.add( Dense(512, activation='elu') )
model.add( Dropout(0.5) )
model.add( Dense(512, activation='elu') )
model.add( Dropout(0.5) )
model.add( Dense(output_dim = 38) )
model.add( Activation('sigmoid') )

```

2. (1%) 請設計實驗驗證上述推論。

我是用 bag of word 去實驗的，以上是我的 model 架構，只差在最後一層的 activation function，分析 model predict training data 的結果。

用 softmax : (val-f1_score : 0.4180)

總 label 數 (n)	prediction 最大前 n+1 個結果
2	0.528、0.472、1.59168252e-04
6	0.215、0.192、0.136、0.112、0.109、0.091、0.086

用 sigmoid : (val-f1_score : 0.5157)

總 label 數 (n)	prediction 最大前 n+1 個結果
2	1.0、1.0、9.37991196e-09
6	1.0、1.0、1.0、1.0、1.0、1.0、5.46586421e-08

而至於 softmax 的 val-f1_score 也可以很高是因為 label 分布的關係。

3. (1%) 請試著分析 tags 的分布情況 (數量) 。

label	數量	label	數量
FICTION	1672	ALTERNATE-HISTORY	72
SPECULATIVE-FICTION	1448	COMEDY	59
NOVEL	992	AUTOBIOGRAPHY	51
SCIENCE-FICTION	959	BIOGRAPHY	42
CHILDRENS-LITERATURE	777	SHORT-STORY	41
FANTASY	773	HISTORY	40
MYSTERY	642	COMIC-NOVEL	37
CRIME-FICTION	368	SATIRE	35
SUSPENSE	318	MEMOIR	35
YOUNG-ADULT-LITERATURE	288	WAR-NOVEL	31
THRILLER	243	AUTOBIOGRAPHICAL-NOVEL	31
HISTORICAL-NOVEL	222	DYSTOPIA	30
HORROR	192	NOVELLA	29
DETECTIVE-FICTION	178	HUMOUR	18
ROMANCE-NOVEL	157	TECHNO-THRILLER	18
HISTORICAL-FICTION	137	HIGH-FANTASY	15
ADVENTURE-NOVEL	109	APOCALYPTIC-AND-POST-APOCALYPTIC-FICTION	14
NON-FICTION	102	GOTHIC-FICTION	12
SPY-FICTION	75	UTOPIAN-AND-DYSTOPIAN-FICTION	11

label數量	1	2	3	4	5	6	7	8
data數量	2119	1415	741	441	168	65	13	2

label 的分佈除了各個 label 間出現的次數很不均勻外，data 間的分佈也非常不均勻，單一 data label 數小於等於 2 就超過全部 data 的七成，造成在使用 softmax 作為 output layer 仍能有不錯的表現。

4. (1%) 本次作業中使用何種方式得到 word embedding？請簡單描述做法。

1. 在 keras 的 model 中加入 embedding layer，跟著 model 一起 train

```
101
102   model = Sequential()
103
104   model.add( Embedding(input_length=MAXLEN, input_dim=51867, output_dim=512,  mask_zero=True) )
105   model.add(LSTM(512, dropout=0.2, recurrent_dropout=0.2, return_sequences=True))
106   model.add(LSTM(256, dropout=0.2, recurrent_dropout=0.2, return_sequences=False))
107   model.add( Dense(output_dim = 256, activation='elu') )
108   model.add( Dropout(0.3) )
109   model.add( Dense(output_dim = 128, activation='elu') )
110   model.add( Dropout(0.3) )
111   model.add( Dense(output_dim = 64, activation='elu') )
112   model.add( Dropout(0.3) )
113   model.add( Dense(output_dim = 38) )
114   model.add( Activation('sigmoid') )
115
116   model.compile( loss='binary_crossentropy', optimizer='Adam', metrics=['accuracy'] )
117
```

2. 用 glove:

trained on the non-zero entries of a global word-word co-occurrence matrix, which tabulates how frequently words co-occur with one another in a given corpus.

1. Collect word cooccurence statistics in a form of word co-ocurrence matrix X. Each element X_{ij} of such matrix represents how often word *i* appears in context of word *j*. Usually we scan our corpus in the following manner: for each term we look for context terms within some area defined by a *window_size* before the term and a *window_size* after the term. Also we give less weight for more distant words, usually using this formula:

$$decay = 1/offset$$

2. Define soft constraints for each word pair:

$$w_i^T w_j + b_i + b_j = log(X_{ij})$$

Here w_i - vector for the main word, w_j - vector for the context word, b_i, b_j are scalar biases for the main and context words.

3. Define a cost function

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - \log X_{ij})^2$$

Here f is a weighting function which help us to prevent learning only from extremely common word pairs. The GloVe authors choose the following function:

$$f(X_{ij}) = \begin{cases} (\frac{X_{ij}}{x_{max}})^{\alpha} & \text{if } X_{ij} < XMAX \\ 1 & \text{otherwise} \end{cases}$$

實作發現使用 glove 的 word vector 效果較好。

5. (1%) 試比較 bag of word 和 RNN 何者在本次作業中效果較好。

我實作上 bag of word 的成績比 RNN 來得好，bag of word 我把一些斷詞去掉，並去掉了一些非英文的詞，再將字詞轉為一樣的詞態，隨便 train 一下就可以達到 0.51 多 (但就上不去了)，但 RNN 怎麼弄最高都只有 0.47 左右，跟同學討論大家做出來的分數也都差不多。