學號：R05921064 系級：電機碩一 姓名：羅嵩詠

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。

回答: softmax 不適合做為這次作業的 output layer，因為當我們在使用 softmax 的時候，基本上它的輸出是一個 38 維的向量，元素都介於零到一之間，而且所有元素加總為一。而且在將預測 label 結果寫到 csv 檔案時，Threshold 的設定不是那麼容易，或著說辦不到。Softmax 較適用於每個 text 只有 single class 的情況。

最後選用 sigmoid，它的輸出形式同樣是一個 38 維的向量，元素都介於零到一之間，不過元素間機率分布就不像 softmax 要遵守一個數學關係，所以我們可以比較容易嘗試出好的 threshold 找出最有可能的幾個 label。而這種算法可以運用到一個 text 有多組 label 的狀況。

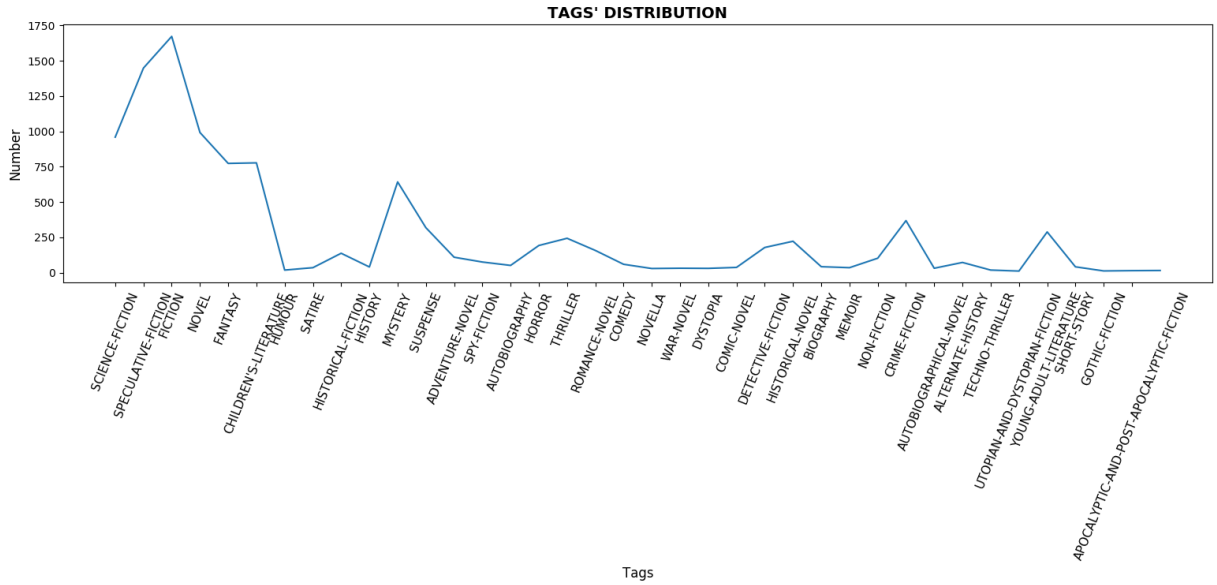
2. (1%)請設計實驗驗證上述推論。

回答: 透過幾次測試，我想驗證最後一層 layer 如果是 softmax，threshold 設定是很困難的。

進行三次實驗，唯一的變因是 threshold，分別為 0.4、0.1、0.05。第一個 0.4 因為太大，所以在最後得到的結果，很多 text 沒有被 label 到，在 train data 切出的 validation set 上得到最好的準確率為 0.18183。第二個 0.1，我預期會得到較好準確率，因為共有 38 組 label，平均起來機率是 1.0/38=0.026，所以真正的 label 要給高一些的機率，所以設為 0.1，但是最終最佳準確率為 0.16169。第三個 0.04，得到最好準確率為 0.19786。雖然實驗給出的數據數量並不多，但也足見使用 softmax 做為網路的最後一層，threshold 可能需要對 data 有更多的認識，才得以設下一個好的 threshold，以得到比較好的準確率。

3. (1%)請試著分析 tags 的分布情況(數量)。

回答: 對 train_data (共 4964 筆)的 tags 進行統計作圖，如下。



前十五名為:

FICTION : 1672.0

SPECULATIVE-FICTION : 1448.0

NOVEL : 992.0

SCIENCE-FICTION : 959.0

CHILDREN'S-LITERATURE : 777.0

FANTASY : 773.0

MYSTERY : 642.0

CRIME-FICTION : 368.0

SUSPENSE : 318.0

YOUNG-ADULT-LITERATURE : 288.0

THRILLER : 243.0

HISTORICAL-NOVEL : 222.0

HORROR : 192.0

DETECTIVE-FICTION : 178.0

ROMANCE-NOVEL : 157.0

觀察得知，屬於虛構(fiction)、小說(novel)的 text 在 data set 中，占了很大一部分。

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

回答: 自 Glove 網站下載 glove.6B.zip，選取 embedding dim 為 100 的檔案，檔案中有許多的字詞以及對應的 vector representation。這些 word vector 是基於 Wikipedia2014 與 Gigaword5 預先訓練過的。在 python 程式中，將它們讀進一個字典，接著把 data 的 word 一個一個依序拿到字典裡查詢它們的 vector representation，一邊查詢一邊將得到的結果儲存起來。另外，在出現 unseen word 時，該字詞在字典裡會查詢不到 vector representation，就以零向量表示。

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

回答:

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 128)	2850176
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 128)	16512
dropout_2 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 64)	8256
dropout_3 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 38)	2470

Total params: 2,877,414
Trainable params: 2,877,414
Non-trainable params: 0

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 306, 100)	5186700
gru_1 (GRU)	(None, 128)	87936
dense_1 (Dense)	(None, 200)	25800
dropout_1 (Dropout)	(None, 200)	0
dense_2 (Dense)	(None, 140)	28140
dropout_2 (Dropout)	(None, 140)	0
dense_3 (Dense)	(None, 38)	5358

Total params: 5,333,934
Trainable params: 147,234
Non-trainable params: 5,186,700

左圖是 bag of word 的網路架構，右圖是 RNN 的網路架構。
Training 時間:bag of word 快於 RNN
Train_data 的準確率:bag of word 高於 RNN
Validation_data 的準確率: bag of word 高於 RNN
Kaggle public set 的準確率: bag of word(0.47132)低於 RNN(0.50599)
Kaggle private set 的準確率: bag of word(0.46069)低於 RNN(0.48587)
結論:在訓練的時候，bag of word 傾向會 overfitting，整體在 kaggle 上的準確率 RNN 估計出來的模型表現較好。