學號：B03901103 系級： 電機三 姓名：陳學平

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。

這次我採用的model是RNN，先將input sequence 通過 embedding layer，再經過LSTM，最後再通過一層dense layer到達output layer，output layer的neroun數量即為tags的數量。

在此model架構下，sofmax並不適用，因為本次作業的資料是 multi-label，所以在output layer會有不只一個neuron的值必須是 1的情形，但softmax會使得output layer之所有neuron的加總之值為1，因此用softmax並不合適。為了使每個neuron的值皆能介於0到1之間，最後我選擇使用了sigmoid 作為output layer的activation function，並且設定threshold為0.3，當某個neroun 之值大於threshold，便將對應的tag視為這個test case包含的標籤。

2. (1%)請設計實驗驗證上述推論。

固定其他各層layer之參數、epochs的數量、以及threshold選定之值，只改變output layer的activation function以及對應的model之loss function。比較兩個 case在validation set上之 F1-score。

Case1：softmax、 categorical cross-entropy

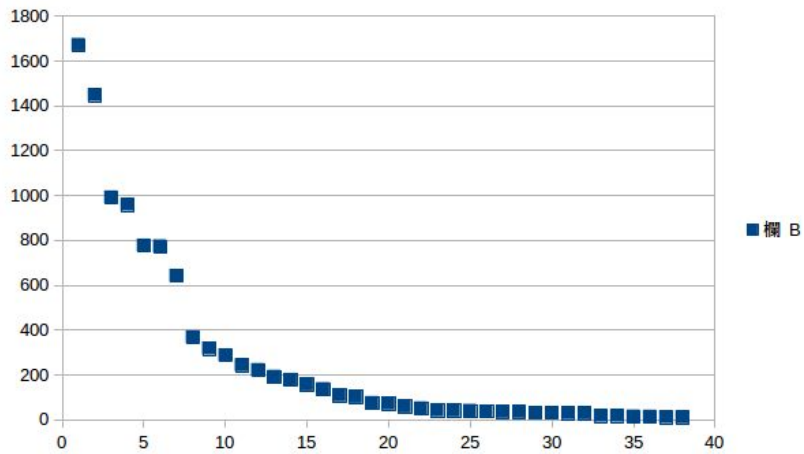
F1-score = 0.26

Case2: sigmoid、binary cross-entroy

F1-score= 0.50

由以上結果可驗證， softmax的確並不適合在這個case上使用。

3. (1%)請試著分析tags的分布情況(數量)。



上圖為 38個 tag 在 training data上的數量分佈情形，顯而易見得，各個tag的數量分佈並不平均，一個主要的原因是有的 tag 的概念比較 廣義，而有

的十分限定。舉個例子來說，數量最多的 tag是 fiction(1672個)，而數量最少的tag是utopian-and-distopian-fiction(11個)，在概念上前者涵蓋了後者，因此前者自然會被標記在更多的文章中。

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

本次作業，我直接使用了glove 的pre-trained word2vec model作為word embedding，以下簡介其作法：

在training 時，首先會建立一個 word-word co-occurence matrix，用來紀錄在corpus中任意兩個字共同出現在一篇文章中的頻率。而training 的目標即是希望對於 任意的兩個字的 word vectors，其內積會等於兩個字共同出現的機率之對數值，如此一來，兩個字的word vectors 在空間上的對應關係集會和語義相關。為了達成以上目標，loss function 大致上是定義成各個 word vectors 的內積與共同出現機率之對數值間的誤差之平方和。由於在 train word vectors 時，我們並沒有output(vectors)的正確答案，因此training 的方式為 unsupervised learning。

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

在 validation set 上，bag of word的 f1-score 是 0.516，而RNN的 f1-score 是 0.502。從結果上來看，bag of word的效果似乎比RNN略好一點。但從概念上來說，因為bag of word無法利用文字的先後順序來判斷語義，因此RNN應該能夠達到更好的performance。然而在實作上，RNN 在 training 上所需花費的時間遠大於 bag of word，因此在兩個 model付出相同時間的情況下，使用者能夠比較容易得將接在 bag of word上的DNN的參數調整的比較好。此外，bag of word雖然無法利用文字的先後順序，但以這次作業的例子來說，這似乎並不會失去太多的資訊，只要bag of word能夠辨別各個tag對應的關鍵字，要將 text分類正確就十分足夠了。