Machine Learning 2017 Spring
Homework 5 Report

學號：B03902048
系級：資工三
姓名：林義聖

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

   答：以 softmax 作爲 output layer 時，會使整個模型的輸出呈機率分佈

   $$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

   然而本次作業是 multi-label prediction，當多個 class 皆可能是正確答案時，他們各自都有很高的機率，而經過 softmax layer 後，每個 class 的機率馬上被均攤，甚至低到連門檻都過不了；若是下修門檻，則可能在一段 input 屬於的 label 較少時，造成 model 丟出信心度不足的預測。因此，我認爲 softmax 不適合作爲 output layer。

   我選擇的 output layer 是 sigmoid，因爲 sigmoid 的輸出是落在 0 到 1 之間，很容易設置一個合理的預測門檻。

2. (1%) 請設計實驗驗證上述推論。

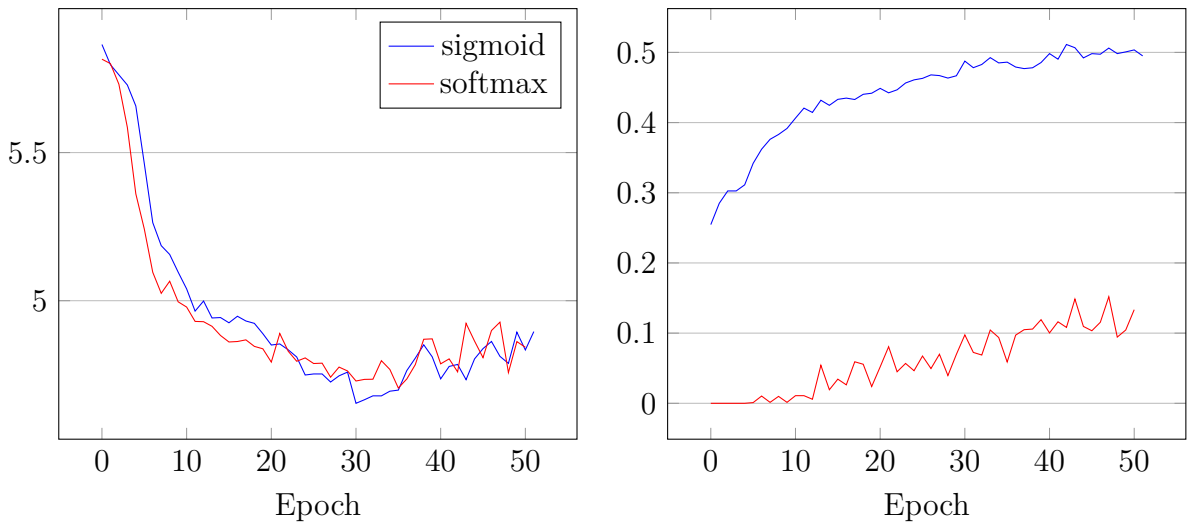
   答：以下是我的 RNN 模型的架構。

```
model.add(GRU(128, activation='tanh', dropout=0.3, return_sequences=
    True))
model.add(GRU(256, dropout=0.3))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(self.nb_tags, activation='sigmoid')) # the activation
    function here is going to be changed.

model.compile(loss='categorical_crossentropy', optimizer='adam',
    metrics=[f1_score])

model.fit(X_train, Y_train, epochs=50, batch_size=128, validation_data
    =(X_valid, Y_valid))
```

   我使用相同的架構、不同的輸出層 activation function 來比較兩種 output layer 的差異。我比較了兩種 output layer 的差異，分別是 “sigmoid” 和 “softmax”，並將在 validation set 上的結果呈現在 Figure 1。其中 Figure 1b 的差異非常地顯著，softmax 的表現完全敗給 sigmoid。



(a) Categorical cross-entropy loss (b) F1 score

Figure 1: Comparison between “sigmoid” and “softmax” as output layer

3. (1%) 請試著分析 tags 的分布情況 (數量)。

答：從 Figure 2 中可以看出，tags 的分佈很極端，其中少數幾種，如：NOVEL、FICTION 和 SPECULATIVE-FICTION 出現頻率都非常高。這種情形，使得模型在訓練階段得到的資訊不足，之後也很難正確的預測出文章類別。

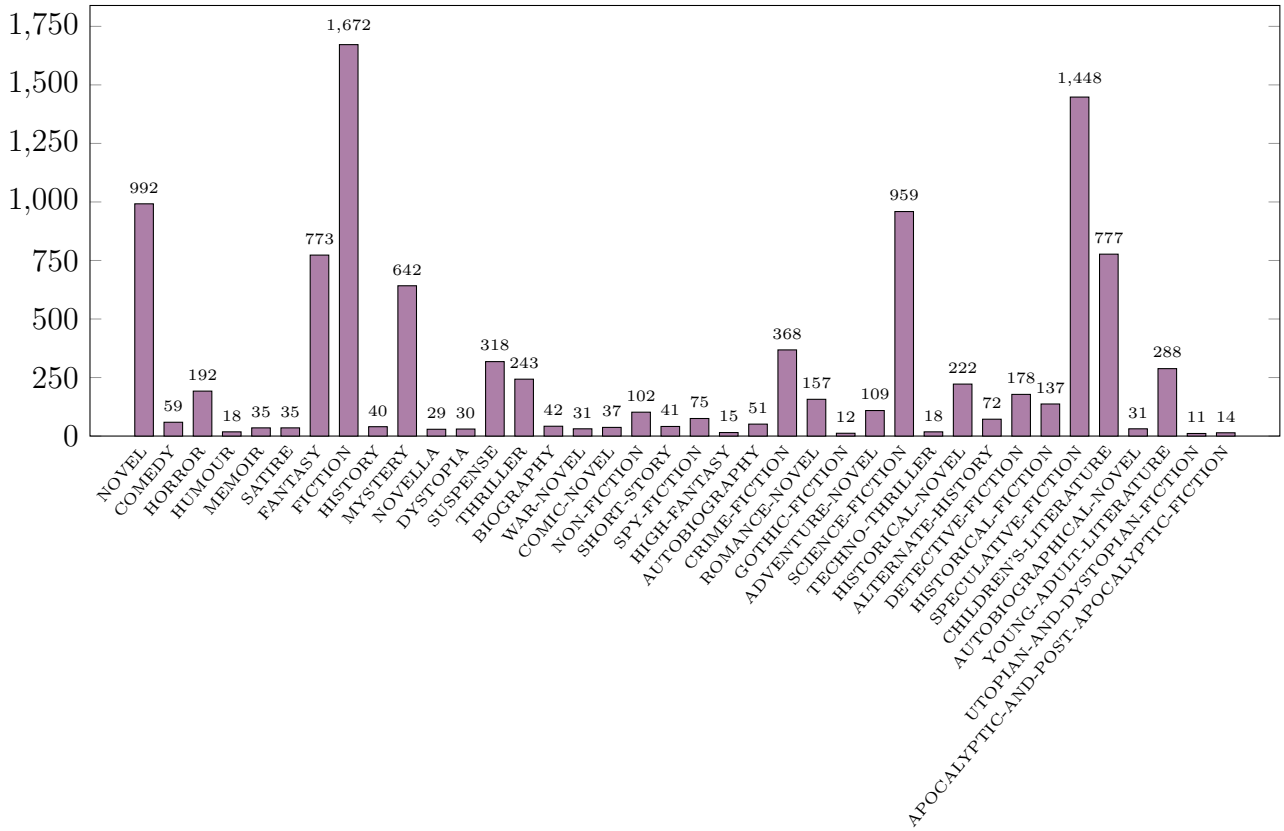


Figure 2: Tags distribution

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

答：我使用由 GloVe 提供，維度 100 的 word2vec，在訓練 RNN 模型前將訓練資料的 words 對應的 vectors 建成一個矩陣，而後將之作爲 embedding layer 的權重，並將 embedding layer 調成在訓練過程中不會更新。

事實上，如果兩個詞時常一起出現在文章中，就代表他們有相近的詞性，所以應該要有相近的 word vector。因此 GloVe 做的事情就是，令兩個相近的詞的向量取內積時，能夠盡量逼近他們在文章中的 co-occurrence，如此，便能以這些向量來描述這些詞彙之間的關係。

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

答：關於 bag of words 的模型實做，我使用了套件 `sklearn` 提供的 `TfidfVectorizer` 將訓練資料轉換成四萬多維的 tf-idf 向量，接著使用 `OneVsRestClassifier` 搭配 `LinearSVC` 來做 multi-label prediction。此外，我也有實做 DNN 版本，同樣使用訓練資料的 tf-idf 向量，搭配五層深度的 DNN 模型，並以 binary cross-entropy 作爲 objective function 來訓練，最終也能達到不錯的成效。

Table 1: Comparison between “bag of word” and “RNN”

	Public set
tfidf + svm	0.5186
rnn	0.5031
tfidf + dnn	0.4962

從 Table 1 中的分數上顯示出，經過 tf-idf 轉換過後的文章向量，藉由 `LinearSVC` 分類的表現，稍微優於 rnn；相較之下，同樣是向量化後的文章，藉由 DNN 分類出來的表現就略差一點。而實際觀察他們對測試資料的預測結果，我發現由 SVM 預測出來的結果有許多空值，亦即它的 recall 應該較低，但能夠在 public set 上有較好的表現，推測是 precision 比較高的緣故。相較之下，RNN 模型在每筆資料上都預測很多 tags，而表現比前者略差，推測應該是有較高的 recall，但 precision 較低的緣故。