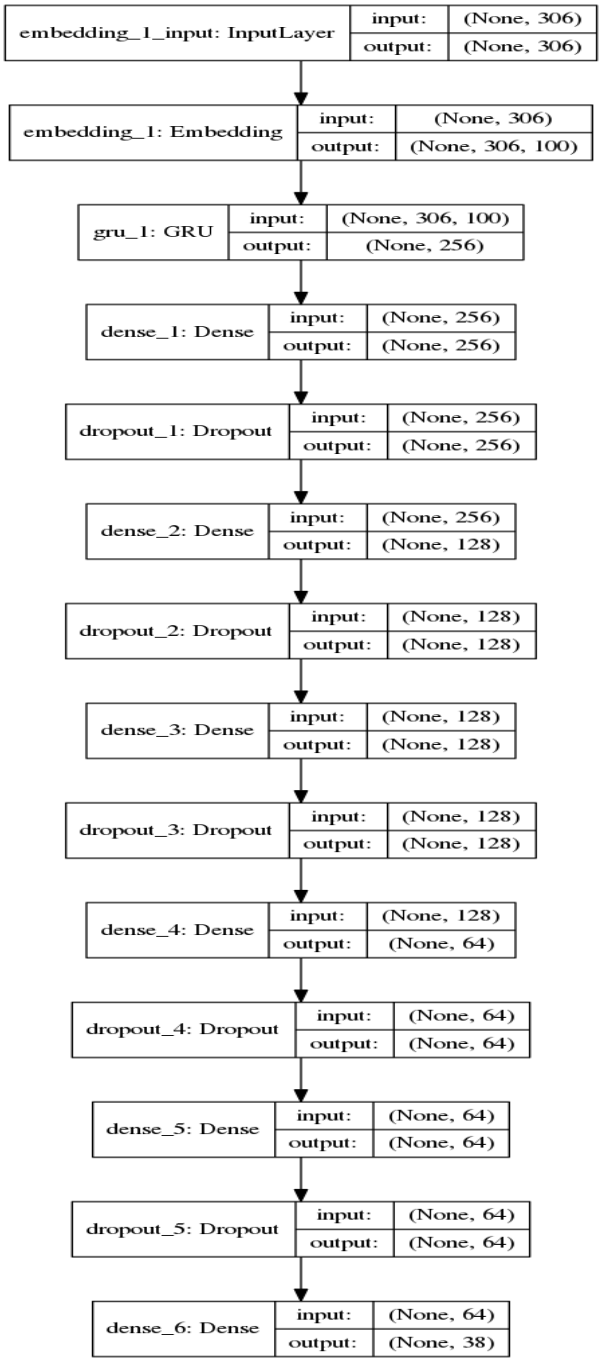
ML2017 HW5 Report

系級：生技三　　學號：b03b02014　　姓名：張皓鈞

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。

下圖為 public score:0.50735 的模型，我將各個訓練資料的句子轉換成每個單字在所有訓練和測試資料文章中的 word-index，再使用 GloVe 的 word-embedding，所使用的 RNN 模型：



我認為 softmax 不適合作為 output layer 的 activation。雖然 softmax 所輸出的值可以被解釋成各個 label 的機率分佈，但是這個機率分佈是同一個分佈，因此總和起來會等於 1，所以在訓練的 gradient descent 時，當其中一個 class 被增加的同時間其他 class 會下降。而這樣並不適用於本次作業 multi-label 的問題上。

最後我選擇 sigmoid 作為 output layer 的 activation，因為 sigmoid 可以將輸出值壓到 0 到 1 之間，但是各個 class 之間是互相獨立的，如此每個 class 輸出的值可以解釋成各個 class 的機率，每個 class 的機率分佈都是互相獨立的。

2. (1%) 請設計實驗驗證上述推論。

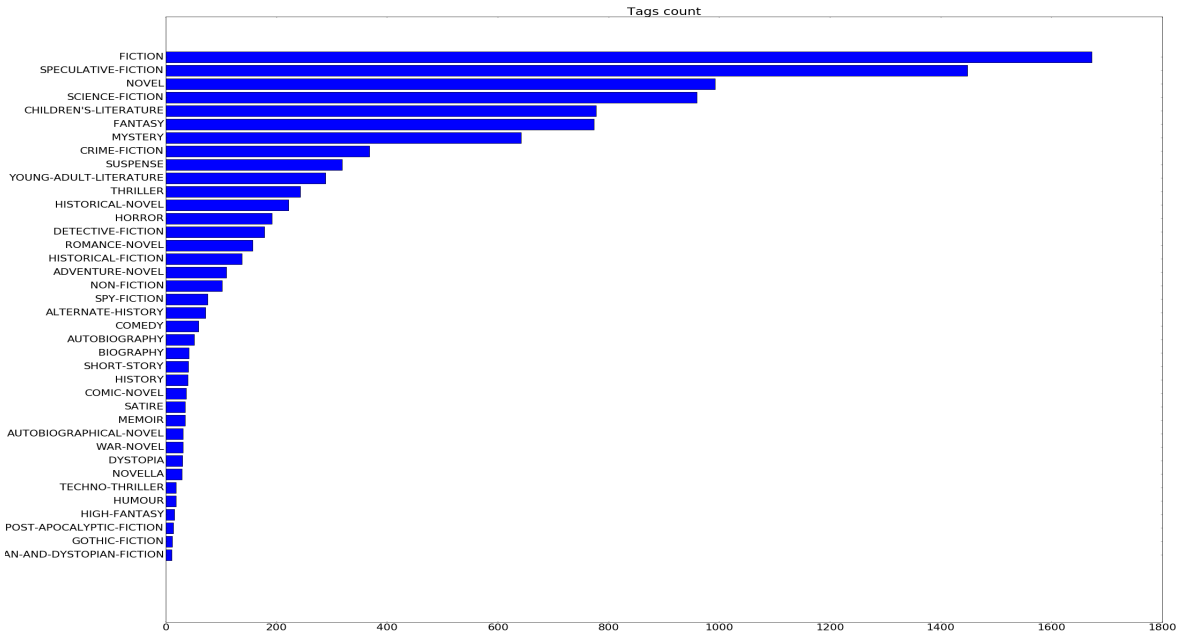
我使用上述的 RNN 架構，只更改 output layer 的 activation，將其變為 softmax，進行與 RNN 相同的訓練過程，訓練結果為：

validation loss	validation f1 score	kaggle public score
4.8199	0.1352	0.40406

在訓練過程中觀察發現 val f1 score 上升的非常緩慢，而且也沒辦法很好的代表 public score。再加上輸入訓練資料觀察輸出的結果，大部份的數值都是 0，只有少數 class 的值大於 0，符合上述對 softmax 只預估一個機率分佈的推論，表示在訓練過程中，gradient descent 只最大化其中一個 class 的機率同時下降其他 class 的機率預估值。

3. (1%) 請試著分析 tags 的分布情況 (數量)。

下圖為 38 種 tag 在 training data 中的出現次數：



我發現 Tag 的數量十分不平均，其中數量佔最多的 tag 為 FICTION，接下來分別為 SPECULATIVE-FICTION、NOVEL 和 SCIENCE-FICTION。而 HUMOUR, HIGH FANTASY 這幾類的數量在 training data 中佔的非常少數。

此 label 不平衡的現象會影響到訓練 model 時的準確度，會使預測出來的結果大部份都會預測 FICTION 或是 SPECULATIVE-FICTION、NOVEL 或 SCIENCE-FICTION 等數量較多的 tag。因此進行分類的時候必須也要考慮到各 tag 出現的數量。

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

我使用的 word embedding 是採用 GloVe 模型，此模型用了 Wikipedia 2014 + Gigaword 5 的文字訓練，共有 6 Billion 個 tokens, 400K 個單字，其中我使用了 100 維的 word vectors 來代表每個訓練資料中的單字。

GloVe 的原理主要是計算所有 corpus 每個單字之間的 co-occurrence 的機率比 X_ij，並且找出一組 w 和 b 可以最小化 w 和 b 扣掉 $logX_ij$ 的平方，而這組 w 和 b 即能用於將對應的單字轉成 word embedding vector。訓練出來的 word vectors 的 dot product 會等於該兩個單字共同出現的 log 機率比。

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

此題的 bag of word 實作為第 1 題 RNN 的架構去除掉 GRU 和 Embedding，進行相同的訓練過程，下表為 bag of word 的訓練結果:

validation loss	validation f1 score	kaggle public score
5.4917	0.4850	0.49062

可以發現在相同的 NN 結構下，RNN 的表現在 validation 和 public score 上都比 bag of word 還要好，符合我們對 RNN 的了解。

因為 RNN 有考慮到單字和單字之間的關係，每次輸入不同單字的 word vector 時，會有前一個 word vector 經過轉換後的 vector 影響這次的 gradient descent，造成 RNN 的 weight 有考慮到單字間的排列關係。

而 Bag of word 只有單純將每個單字在文章中出現的次數記錄下來，以單字出現次數當作該單字的 feature，直接輸入進 DNN 進行 gradient descent，如此並沒有考慮到單字在文句中出現的順序關係，而是將每個單字視為獨立的樣本，如此雖然在此問題可以有尚可接受的分數，而且在訓練過程中比較容易，不太需要調整參數，但是在本質上還是沒辦法跟 RNN 相比。