ML hw4 report

學號：b07901112 系級：電機二 姓名：劉聿珉

1. (1%) 請說明你實作的RNN的模型架構、word embedding 方法、訓練過程(learning curve)和準確率為何？ (盡量是過public strong baseline的model)

我的RNN架構為：

sen\_len = 35

fix\_embedding = True

batch\_size = 128

epoch = 10

lr = 0.0005

我的LSTM模型參數為：

embedding\_dim=250,

hidden\_dim=250,

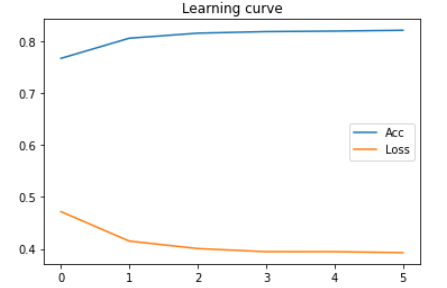
num\_layers=5,

dropout=0.7，

RNN有使用biderectional。

word embedding的方法則是使用word2vector函式，Word2Vec的參數如下：size = 250, window = 5, min\_count = 5, workers = 12, iter = 10, sg = 1。

Learning curve如圖：(此圖為val 的 Acc和Loss)



Kaggle上最好的val acc為82.538。

經由多次實驗可以發現Val Acc每次大約在3次epoch之後就會飽和或是開始下降。但是Training Acc則會繼續上升，但是之後的train就對正確率沒有太大的幫助。

1. (2%) 請比較BOW+DNN與RNN兩種不同model對於"today is a good day, but it is hot"與"today is hot, but it is a good day"這兩句的分數(過softmax後的數值)，並討論造成差異的原因。

我BOW+DNN跑出來的分數兩句都是0.61152，而RNN跑出來的分數第一句是0.645332第二句則是0.97456。

BOW是Bag of word的簡寫，在此模型下會將一段文本（比如一個句子或是一個文檔）用一個裝著這些詞的袋子來表示，所以他完全不會考慮到任何的文法或是語句順句的問題，所以用BOW+DNN跑這兩句話的分數當然獲一樣(因為他們的字詞都一樣只是排序不同而已)。

但是RNN的情形不一樣，之前的輸入會影響到接下來的輸出，他有類似人腦一樣的記憶功能，所以前後文會影響到最後的分數，而就這兩句來看，他們的雖然字一樣但是排序不一樣，所以丟到RNN裡面之後，分數自然就會不一樣。

1. (1%) 請敘述你如何 improve performance（preprocess、embedding、架構等等），並解釋為何這些做法可以使模型進步，並列出準確率與improve前的差異。（semi supervised的部分請在下題回答）

Biderectional：BRNN的基本想法是每一個訓練列向前和向後分別是兩個RNN，而且這兩個都連線著一個輸出。這樣的結構可以提供輸出層的輸入列完整的過去和未來的上下文資訊，使輸出的點不會只看見向前的文字。

Word embedding：加入nolabel data，可以讓embedding時的data變多，參考的data也變多，以得到比較好的結果。

Trainning時做clipping：加clipping可以讓model在train時如果踩到懸崖可以不至於爆掉，但是但部分的時間都沒有發揮作用，所以我最後式沒有加。

參數們：

sen\_len：一次考慮更長的句子可以得到更多的資訊，得到更完整的句子，但是如果調太長可能會稀釋掉每個字詞的重要性這樣反而會使正確率降低。

hidden\_dim, num\_layers：我將這兩個參數調大，試圖train更多參數，以達到更好的結果，但在這同時，我也將dropout調大，這樣可以避免overfitting。

1. (2%) 請描述你的semi-supervised方法是如何標記label，並比較有無semi-supervised training對準確率的影響並試著探討原因（因為 semi-supervise learning 在 labeled training data 數量較少時，比較能夠發揮作用，所以在實作本題時，建議把有 label 的training data從 20 萬筆減少到 2 萬筆以下，在這樣的實驗設定下，比較容易觀察到semi-supervise learning所帶來的幫助）。

這題的架構我是用第一題所用到的model下去改的，幾乎所有的參數都維持不變。在第四題的model中我設的high\_threshold為0.8，low\_threshold為0.2，在第一次用有label的data train完一次之後，便拿他去測試那些沒有label的data，並將分數超過0.8的data設成1低於0.2就設成0，經過這樣的步驟之後我們有label的data就變多了，於是將這及果得到的data再拿去train一次。我發現在沒有使用unlabel的data的狀況下，val acc大約在76.3422，train第二次之後就變成了77.4086，正確率有提高一些，但是沒有我想像中的顯著，我想如果label以及trainning的次數可以再多幾次或許可以拿到更好的結果(也有可能是我threshold參數調得不好)。

semi-supervised的想法就像是用已經train好的model再對一群沒有label的data重新label一次，再把那些data拿來用，這樣我們就得到更多data，更有機會得到正確率更高的結果。這種方法通常在有label 的training data相對少的時候會有比較顯著的效果。