學號：b05502026　系級：機械四　　姓名：林冠宇

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)

我的模型有四個主要的部份組成：word-embedding layer、CNN layer、RNN layer、fully connected layer，以下分別介紹他們的細節。

Word-embedding layer：我使用 skip gram 的方法訓練 word embedding 模型，幾個重要的改變包括加入未標記的文本進行 word vector 的訓練、拉長訓練的 iter 至 20、增加 word vector 的維度至 512、更改取樣的 negative count 至 15。除此之外，我也將每一句話的句長增長至 35

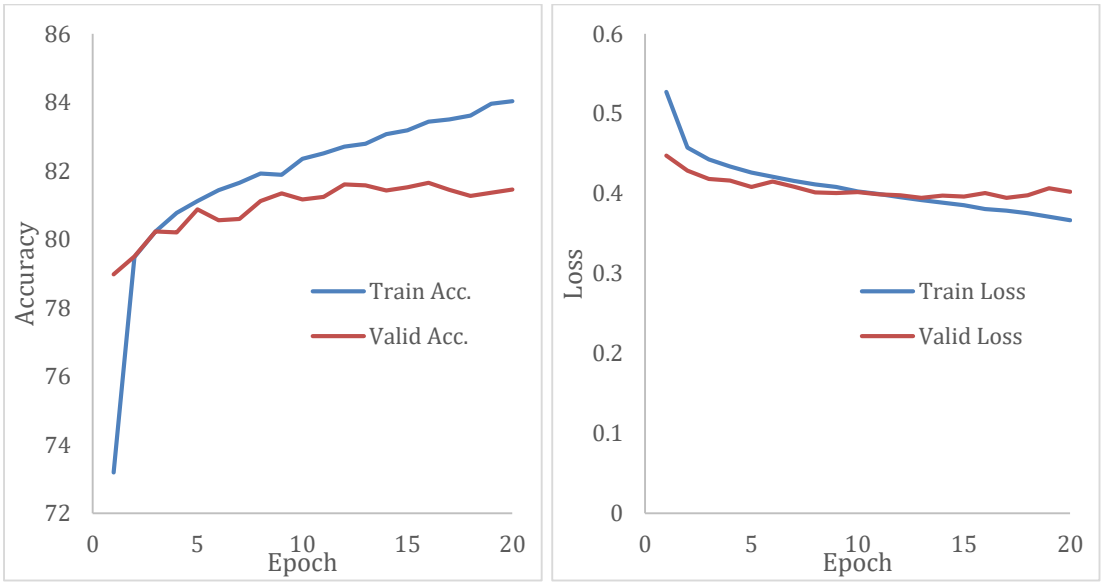
CNN：我的 CNN 是插在 word embedding 跟 RNN 之間，他的目的是將 word embedding 完的結果做簡單的捲積與池化，這樣的目的是模擬 bigram 或是 trigram 的詞意。原本的模型是將 word vector 直接送進 RNN 層，我認為這樣容易受到單一字詞影響結果，因此我以 CNN 捲積池化的方式先篩選鄰近字詞，希望藉此抓出更好的特徵送進 RNN 層。我的 CNN 只有做一次簡單的捲積池化，我發現增加層數並不會增加準確率。

RNN：我將原本使用的 lstm 改成 GRU、hidden dimension 改成 100、GRU 層數設定為 3 層，並在訓練時加入 0.5 的 drop out，其中我認為最有用的修改是增加 GRU 的層數。

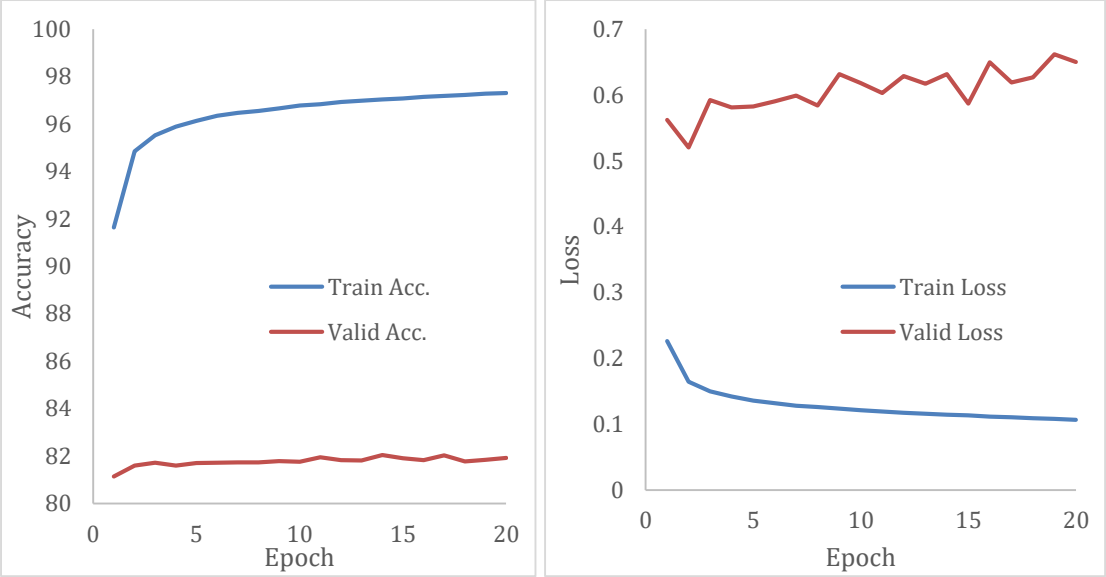
Fully connected layer：就是個簡單的三層全連接層，從 100 降至 50 降至 25 最後降至 1，其中訓練時會使用 0.5 的 dropout。

另外在介紹我的 semi supervised 訓練方式，我使用 0.6、0.4 的 threshold 去做 pseduo-labeling。在訓練這些資料的時候，我有在訓練中加入 L2 regularization，目的是不讓錯誤標記的雜訊過度影響模型準確率，L2 regularization 的權重我設定為 1e-6。

下圖顯示還沒加入 pseduo-labeling 資料的模型訓練 accuracy 與 loss：



下圖顯示加入 pseduo-labeling 資料之後的模型訓練 accuracy 與 loss：



比較加入 pseduo-labeling 資料前後的訓練過程我們可以發現：加入 pseduo-labeling 資料之後雖然 validation loss 上升，但是 validation accuracy 也同時上升，這樣的現象我認為是因為當資料量上升後，模型因為接收到更多不同資訊，導致在判斷時無法像原本資訊較少時一樣有信心，因此模型估計的數字離 0.5 較接近，導致 loss 在計算時看起來比較大，但是整體而言準確率有提升。

2. (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。

句子	BOW+DNN	RNN
today is a good day, but it is hot	0.4272	0.1843
today is hot, but it is a good day	0.4272	0.9591

上表顯示兩句話在不同的 model 下得到的分數，可以發現 BOW+DNN 的方法在不同的句子中得到相同的分數，這是因為雖然兩句話中的單字順序不同，但是 BOW 是以一個向量就表示完整句話的意思，因此並不能表示單字間的順序關係。反觀 RNN，是以一個向量表示一個單字，因此一個句子是由一個向量組所組成，如此除了每個單字的意思能更完整的表示之外，單字之間的順序也可以被 model 考慮進去，因此在以上兩句話中產生不同的結果。

目前所使用的 BOW 還有一個問題是沒有移除 stopword，一般而言句子常常出現 I、you、to、at…等連接詞或代名詞，這些句子雖出現頻率高但是對句子本身的意思卻難有貢獻，甚至在不同句子中可能有不同的情緒，因此未移除 stopword 的 BOW 可能會被 stopword 影響其準確性。

3. (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）

Preprocess 部分，我將句子長度從 20 改為 35，會這麼做的原因是因為我觀察 training_label.txt，發現幾乎每二十句話就會出現一句話長度超過 30，因此若句長只有 20，等於有 1/3 的句子不會被解讀其句意，明顯會造成情緒分析偏差。

Embedding 部分，我除了簡單調整 gensim 的 Word2Vec 參數，最主要的改變是加入 unlabel_data 作為訓練 w2v 的文本。觀察兩個文本可以發現有標記情緒的貼文數僅有 200000 則，但是未標記情緒的有 1170000 則以上，雖未標示情緒，但是其句中單字間的關係亦可拿來訓練 word embedding。加入未標記的貼文將擴大訓練用的資料 6 倍，更能減少小資料集合帶來的誤差。

模型架構中我除了將 lstm 改為 GRU、GRU 層數改為 3 層、調整 learning rate 等參數外，最重要的改變是讓資料進入 GRU 前先通過一個 CNN 層。CNN 的優點是可以擷取資料局部中的重點、RNN 的優點則是可以考慮資料順序造成的影響，因此我可以使用 CNN 搭配 Maxpooling 的方式，將單字兩兩組合成 bigram，如此可能更容易抓出句子中的特徵，而不會被單一單字影響其輸出。利用 CNN 擷取完區域特徵後，再將這些特徵送入 RNN 分析它順序所造成的影響，如此可兼顧 CNN 及 RNN 的優點。

下表顯示每個修正後，模型預測準確度在 kaggle 上的表現為何，排序方式為時間先後，較晚進行的更動會被放置在越右方。經觀察後發現影響最大的是加入 CNN 的時候，雖然有可能是因為突破 82 面臨的難度較高，但不可否認加入 CNN 有其重大的影響力。

修正的項目	助教 code	Embedding	加入 CNN	Preprocess
Kaggle 正確率	80.512	81.149	81.849	82.271

4. (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）。

   我標記 label 的方式為當估計分數大於 0.8 或小於 0.2 時將其標記，我亦有試過其他的數字如 0.9/0.1、0.99/0.01，但我發現標準越嚴格，反而會使 semi supervised 後的結果變差，推測其原因應該是若標準變嚴格，是變相的加強模型原本既定的預測方式，而沒有辦法改變模型已經存在的 bias，較鬆散的標準才有機會讓模型學到更多不同的資料，但在此同時，卻也有可能引入更多的雜訊(model 的預測並不一定準確，可能會有標記錯誤讓模型學錯的情形)，故在進行 semi-supervise learning 時需要在納入更多資料與避免錯誤標記間平衡。
   我目前的平衡還沒有抓得很好，故僅能將 validation accuracy 自 81.713 進步至 82.021，這甚至是採用了全部 20 萬筆的 train 資料，而非題目建議的 2 萬筆，若以題目建議的 2 萬筆操作時，反而出現 validation accuracy 下降的情形，推測就是因為還沒有辦法降低錯誤標記造成之影響的緣故。
   我之前也有試過老師上課投影片中所述的 Entropy-based regularization，亦即將訓練的 loss 分成兩部分：有標記的資料以原本分類使用的 cross entropy 計算，而未標記的資料則是計算他們被分開的程度作為 loss，亦即資料之 entropy。但這樣的做法完全訓練不起來，model 無法做出同時讓兩種 loss 一起降低的更新，因此更新參數的方向相當混亂，無法控制。最後我就放棄這個方法了。