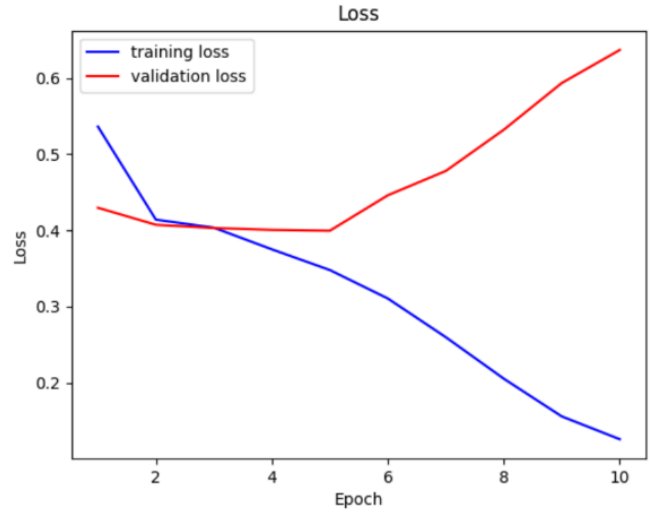
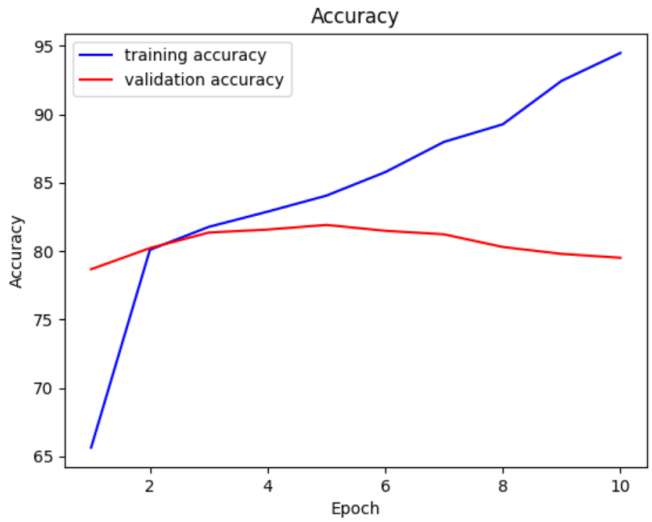
學號：b05901070　系級：電機四　　姓名：蔡昌廷

Collaborator : b05901071 孫鍾恩

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)

RNN 整個架構是先將 input 的 word 經過 embedding，形成 300 維的 vector，接著 sentence 丟進 hidden layer = 500 且三層的 LSTM 去跑，在 LSTM 裡面有使用 dropout 來避免 overfitting，再來將 LSTM 的 output 取其最後一層的 hidden state 經過一個 output 為 1 的 Linear 層，在此 Linear 層也有使用 dropout，最後將其通過 Sigmoid function ，透過其值是否大於 0.5 來判斷是不是正面含意的句子。
Word embedding 的方法是透過 gensim 的 word2vec 方式，使用 skip-gram 將 train data 跟 test data 的 word 轉換成 300 維的 vector。
以下為訓練過程的 learning curve。





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。
在實作 BOW 的時候我將 training data 二十萬筆的 data 的 word 都丟進去 bag 裡面，最後一個 sentence 會形成 82117 維的 vector，接著丟進去 DNN 的 model 做預測，以下是我實作 DNN model 的架構。

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Linear-1              [-1, 1, 1000]      82,118,000
              ReLU-2              [-1, 1, 1000]               0
            Linear-3                 [-1, 1, 1]           1,001
           Sigmoid-4                 [-1, 1, 1]               0
================================================================
Total params: 82,119,001
Trainable params: 82,119,001
Non-trainable params: 0
----------------------------------------------------------------
```

在經過 5 個 epoch 後
training accuracy 為 98.855 %
validation accuracy 為 77.206 %
“today is a good day, but it is hot”經過 softmax 數值為 0.98347 (正面)
"today is hot, but it is a good day"經過 softmax 數值為 0.98347 (正面)

而使用我自己的 LSTM model 去測試得到以下結果。
“today is a good day, but it is hot”經過 softmax 數值為 0.1485 (負面)
"today is hot, but it is a good day"經過 softmax 數值為 0.9810 (正面)

從以上結果來看會發現用 LSTM model 做出來的結果比較正確，兩個 model 不同在於 BOW 將 word 轉成 vector 的過程是根據 word 出現的數量來決定，並不會考慮到前後文的影響，也因此在此題的句子中兩句都會被轉換成相同的 vector，以至於丟進去 DNN 偵測出來後會是相同的數值，至於 LSTM 的 word embedding 會考慮到前後文，也因此在最後偵測出來的結果也比較符合預期，透過這個題目讓我們了解處理文字的時候也必須考慮到前後文的關係。

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）

word2vector：在原本 word2vector 中，我只使用 train data 跟 test data 去實作，後來我把 unlabel data 也加進去 w2v 的 data 裡面，另外因為增加更多 data，我也把 min_count 的標準提高，藉此忽略掉一些較少出現的 word，最後在 train 的效果上看起來有稍微改善一些，改善的原因推測為透過更大量的 data，可以使 word 轉換到更能合適表現其語意的 vector。
Validation accuracy 改善前：81.85%、改善後：82.05%

調整 sentence length：用 LSTM training 會需要輸入固定長度的 sentence，而一開始我設定 length = 20，一句話長度大於 length 的部分會被砍掉，在查看 data 後發現，有些 data 的長度其實比 20 還長，因此調高 sentence length 可以避免某些 data 的後半部分直接被砍掉，進而增加 performance。
Validation accuracy 改善前：81.85%、改善後：82.011%

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）。

   此處 semi-supervised 使用 self training 的方式，我將 unlabel data 經過 model 後 output 超過 0.6 才將其標記 1(正面)，而 output 要低於 0.4 才將其標註 0(負面)，最一開始只用 20000 筆 label data 去 train model，並用另外 20000 筆 label data 做 validation，最後原本 120 萬筆 unlabel data 因此有被標記的有 897379 筆，接著把這 897379 筆加上原本的 20000 筆 label data 合併成新的 training data 重新 train 一個 model 去觀察 validation 的變化

	原始 20000 筆	加上標註後的 897379 筆
Validation acc	67.745 %	69.019 %

   從上面結果發現使用 semi-supervised 之後準確率確實有稍微提升一些，造成此現象的原因推測是因為在 unlabel data 裡面有一部份 data 標準答案的機率分布跟一開始只用 20000 筆 train 出來的 model 是類似的，因此當使用最初 model 預測 unlabel data 出來的數值超過設定值後，就可以更相信這些 answer 是正確，把這些 data 加進去 training 後再訓練就可以提升最後 validation 的準確率。