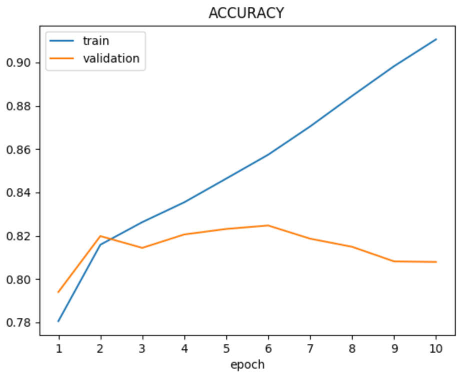
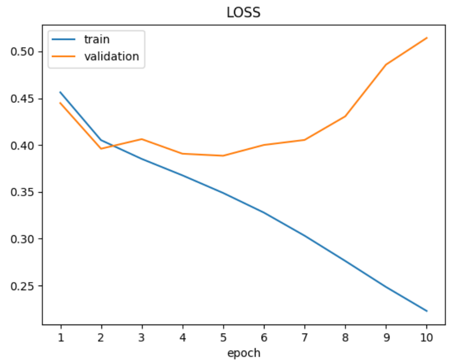
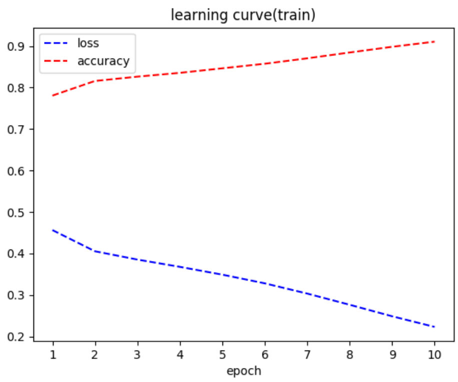
學號：B06901066　系級：電機三　姓名：孟妍

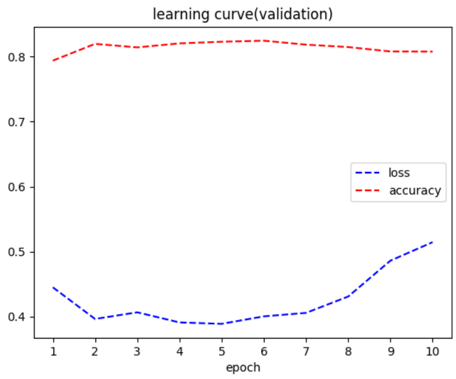
1. (1%) 請說明你實作的RNN的模型架構、word embedding 方法、訓練過程(learning curve)和準確率為何？ (盡量是過public strong baseline的model)

一開始將原本的LSTM改成bidirectional，並把sentence length加長成40就可以過strong baseline，word embedding用genism的word2vec，用所有data（both labeled and unlabeled training data and testing data）一起train。
訓練過程如下：









validation的accuracy一開始就蠻高的大約0.79左右，最好會到0.82多，但很快就會overfit，所以雖然train 10個epoch但大概最好的結果會第6,7個的時候。 kaggle上public的分數為0.82430。

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。

(1) 使用BOW+DNN，這兩句話的分數會一樣

```
loading testing data ...

load model ...
[0.5876865386962891, 0.5876865386962891]
"today is a good day, but it is hot": 0.5876865386962891
"today is hot, but it is a good day": 0.5876865386962891
save csv ...
Finish Predicting
```

由上圖，使用BOW+DNN兩句話皆會被歸類為正面，因為BOW是不考慮文法以及詞的順序，這兩句的詞都一樣只是順序有點不同，所以用BOW作為embedding會得到一樣的分數。

(2) 使用RNN，兩者output結果差很多

```
load model ...
[0.1283694952726364, 0.9900282025337219]
"today is a good day, but it is hot": 0.1283694952726364
"today is hot, but it is a good day": 0.9900282025337219
Finish Predicting
```

由上圖，"today is a good day, but it is hot" 分數很低會被歸類為負面，而 "today is hot, but it is a good day"分數很高會被歸類為正面，實際來看這樣的結果應該是合理的。跟BOW+DNN會有很大差別是因為RNN有可以由前後文判斷語意能力，因此即使兩句話裡只有詞的順序不同，通過softmax後的分數就可能有很大不同，像這裡兩句話就得到完全相反的結果。

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）**

一開始有做各種preprocess，有用一些sentiment analysis的方法，例如移掉標點符號，移掉一些stop words 之類的但做了很多preprocess去測試結果都比較差，感覺可能因為每句都蠻短的，所以標點符號也算是佔了判斷正負很重要的部分。所以後來就沒有做preprocess，先把LSTM改成bidirectional，因為改成bidirectional，所以也同時把sentence length加長，這樣就可以過strong baseline。改成bidirectional效果不錯的原因是因為這樣可以不只從，前文

推導關係，也可以由後文推導，可以學得更完整。之後再把LSTM換成GRU，accuracy跟kaggle的分數都有再上升一些到0.82553。

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semi-supervised設定 pos_threshold = 0.85，取前2萬筆training data做training，再去predict unlabeled data的label，然後加入新預測的較有信心的data(>0.85 or <0.15)一起做training。只用20000筆labeled data去train，validation accuracy大概只能到77-78%左右。

```
Valid | Loss:0.47361 Acc: 77.126
saving model with acc 77.126
-----------------------------------------------
[ Epoch4: 79/79 ] loss:0.371 acc:10.156
Train | Loss:0.46830 Acc: 77.453
Valid | Loss:0.46854 Acc: 76.305
-----------------------------------------------
[ Epoch5: 79/79 ] loss:0.413 acc:9.375
Train | Loss:0.45649 Acc: 78.115
Valid | Loss:0.47625 Acc: 76.266
-----------------------------------------------
[ Epoch6: 79/79 ] loss:0.480 acc:9.375
Train | Loss:0.43937 Acc: 79.025
Valid | Loss:0.45706 Acc: 77.700
saving model with acc 77.700
-----------------------------------------------
[ Epoch7: 79/79 ] loss:0.342 acc:10.156
Train | Loss:0.42107 Acc: 79.668
Valid | Loss:0.44731 Acc: 77.532
-----------------------------------------------
[ Epoch8: 79/79 ] loss:0.289 acc:10.938
Train | Loss:0.40463 Acc: 80.914
Valid | Loss:0.44492 Acc: 77.759
saving model with acc 77.759
-----------------------------------------------
[ Epoch9: 79/79 ] loss:0.365 acc:9.375
Train | Loss:0.39279 Acc: 81.665
Valid | Loss:0.44711 Acc: 77.621
-----------------------------------------------
[ Epoch10: 79/79 ] loss:0.287 acc:10.938
Train | Loss:0.36883 Acc: 83.228
Valid | Loss:0.46084 Acc: 77.087
-----------------------------------------------
Get embedding ...
```

[training with 20000 labeled data]

```
[ Epoch1: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.02704 Acc: 98.535
Valid | Loss:1.63043 Acc: 75.989
saving model with acc 75.989
-----------------------------------------------
[ Epoch2: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.01029 Acc: 99.579
Valid | Loss:1.56987 Acc: 76.503
saving model with acc 76.503
-----------------------------------------------
[ Epoch3: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.01001 Acc: 99.592
Valid | Loss:1.54982 Acc: 77.215
saving model with acc 77.215
-----------------------------------------------
[ Epoch4: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.00962 Acc: 99.613
Valid | Loss:1.56778 Acc: 77.106
-----------------------------------------------
[ Epoch5: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.00930 Acc: 99.625
Valid | Loss:1.43413 Acc: 77.739
saving model with acc 77.739
-----------------------------------------------
[ Epoch6: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.00892 Acc: 99.644
Valid | Loss:1.86516 Acc: 76.800
-----------------------------------------------
[ Epoch7: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.00918 Acc: 99.641
Valid | Loss:1.69498 Acc: 77.561
-----------------------------------------------
[ Epoch8: 4668/4668 ] loss:0.000 acc:7.812
Train | Loss:0.00823 Acc: 99.673
Valid | Loss:1.77293 Acc: 77.482
-----------------------------------------------
[ Epoch9: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.00830 Acc: 99.678
Valid | Loss:1.76150 Acc: 77.739
-----------------------------------------------
[ Epoch10: 4668/4668 ] loss:0.000 acc:7.812 0
Train | Loss:0.00808 Acc: 99.685
Valid | Loss:1.73121 Acc: 77.720
-----------------------------------------------
```

[self training with pos_threshold 0.85]

(忽略Epoch旁邊的loss和accuracy，不知道為什麼印出怪怪的東西QQ)

加入新data後training accuracy變很高，loss很低，train 10個epoch loss有微微下降，training accuracy 一開始就蠻高的所以也是微微上升。在沒有做semi-supervised training之下其實validation accuracy上升幅度不大，但做了semi-supervised training validation accuracy還是跟本來差不多，loss還變大。可能因為labeled跟unlabeled data數量真的差很多，原本的model accuracy也不是非常高，或是threshold設的不夠高，導致predict出的label錯誤的數量不少，無法fit在有ground truth的unseen data上。