ML HW4 Report

學號：b07902040 系級：資工二 姓名：吳承軒

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)

模型架構:

```
self.gru = nn.GRU(embedding_dim=300, hidden_dim=200, num_layers=2,
batch_first=True, dropout = 0.5)
self.classifier = nn.Sequential(
    nn.Dropout(dropout),
    nn.Linear(200, 200),
    nn.Sigmoid(),
    nn.Dropout(dropout),
    nn.Linear(200, 1),
    nn.Sigmoid()
)
```

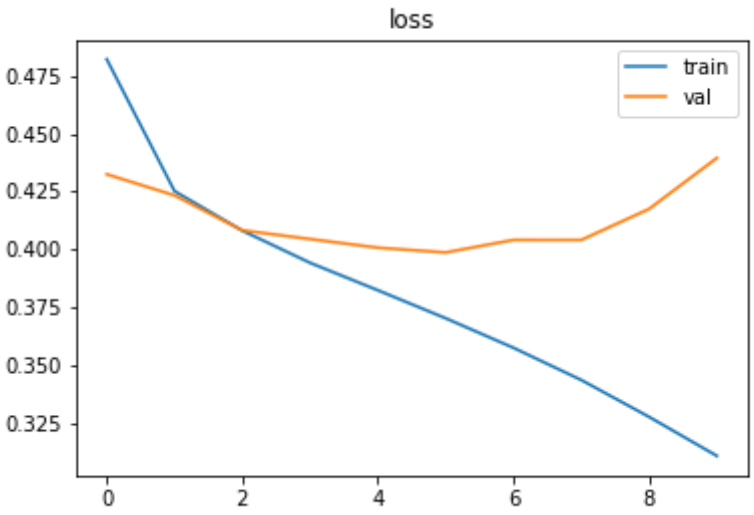
參數:

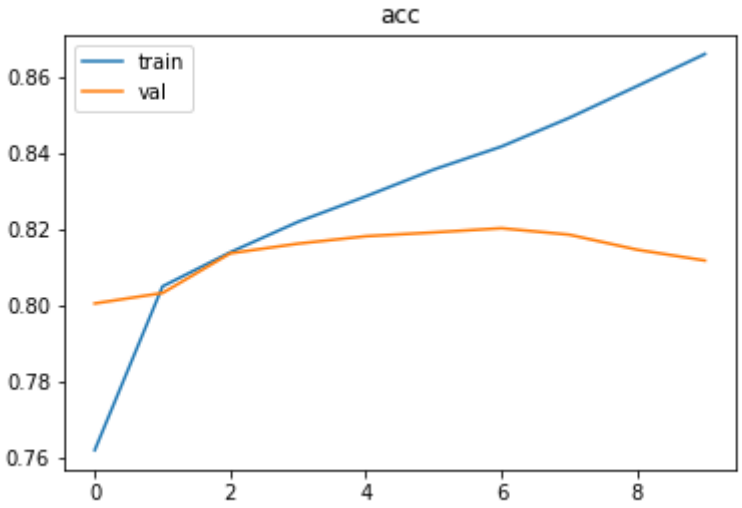
```
sen_len = 35
fix_embedding = True # fix embedding during training
batch_size = 16
epoch = 4
lr = 0.0005
```

word embedding方法:使用skip-gram

```
model = word2vec.Word2Vec(x, size=300, window=5,
 min_count=5, workers=12, iter=10, sg=1)
```

訓練過程:





正確率:

在kaggle public上為82.421%

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。

String1 = "today is a good day, but it is hot"

String2 = "today is hot, but it is a good day"

	String1	String2
RNN	0.2533	0.8798
BOW+DNN	0.6984	0.6702
In fact	0	1

tensor([0.4768, 0.3333, 0.6509, 0.4940], device='cuda:0')

RNN得到的結果會受前後文影響，而BOW+DNN只看每個單字的詞頻。

RNN應該已發現在good後面接but的，通常為負面語意；而在but後面加good則通常為正面語意，所以才得出較接近0和1的結果。

BOW+DNN對兩句話的結果幾乎一樣，都預測為正面，是因為這兩句話中出現的詞是一模一樣的，而之所以預測為正面，推測是因為有good這個字，但其他字並無特別正面之意，hot可能還略帶有負面，所以得出的結果距離1較遠。

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）

1.使用兩層GRU的架構並加上dropout，因為原先的一層在torch中是內建不會有dropout的，減緩over fitting的現象。

2.將embedding_dim設為300, hidden_dim設為200，以產生更複雜的model。

3.將sentence len設為35，以包含更長的句子。

3.將batch size設為16，在更小的資料範圍內就更新，並將lr設為0.0005，避免更新過快。

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）。

使用Self-training實作：使用較好的train_model，將train_nolabel作為test 作predict，>0.95的標記為1，<0.05的標記為0，其餘捨棄，最後得到65137筆labeled的data。

在同一個train_model下，比較做semi-supervised前後在val上acc和loss的差距。

1.原始的train_data(20000筆)

2.合併由train_nolabel得到的data和原始的data(20000+65137筆)

Accuracy(%)	20000	85137
1	77.654	67.672
2	79.020	75.483
3	79.429	75.632
4	79.459	76.227
5	78.380	75.869
6	79.092	74.772
7	78.765	75.025
8	77.989	73.633
9	75.940	73.625
10	77.782	71.783

Loss	20000	85137
1	0.47774	0.65075
2	0.44895	0.63295
3	0.44088	0.62408
4	0.45029	0.61415
5	0.46468	0.60624
6	0.46671	0.60842
7	0.48879	0.60822
8	0.52404	0.60856
9	0.61179	0.59724
10	0.71215	0.61101

可以發現使用semi-supervised後，Accuracy和loss都略差於原始20000筆的數據，推估可能原因為:半監督式學習通常用於training data較少或較難取得時，透過標記原本無標籤的data來增加data量。而從使用20000筆原始資料的結果可以看出，其正確率和使用180000筆原始資料相差並沒有相當大(約82%)，說明對我的model而言，20000筆已勉強滿足所需，而後來加的65137筆中，可能會有被標錯的data，這些被標錯的data對model的負面影響大過標對的data對model的正面影響(因為20000筆已勉強充足)