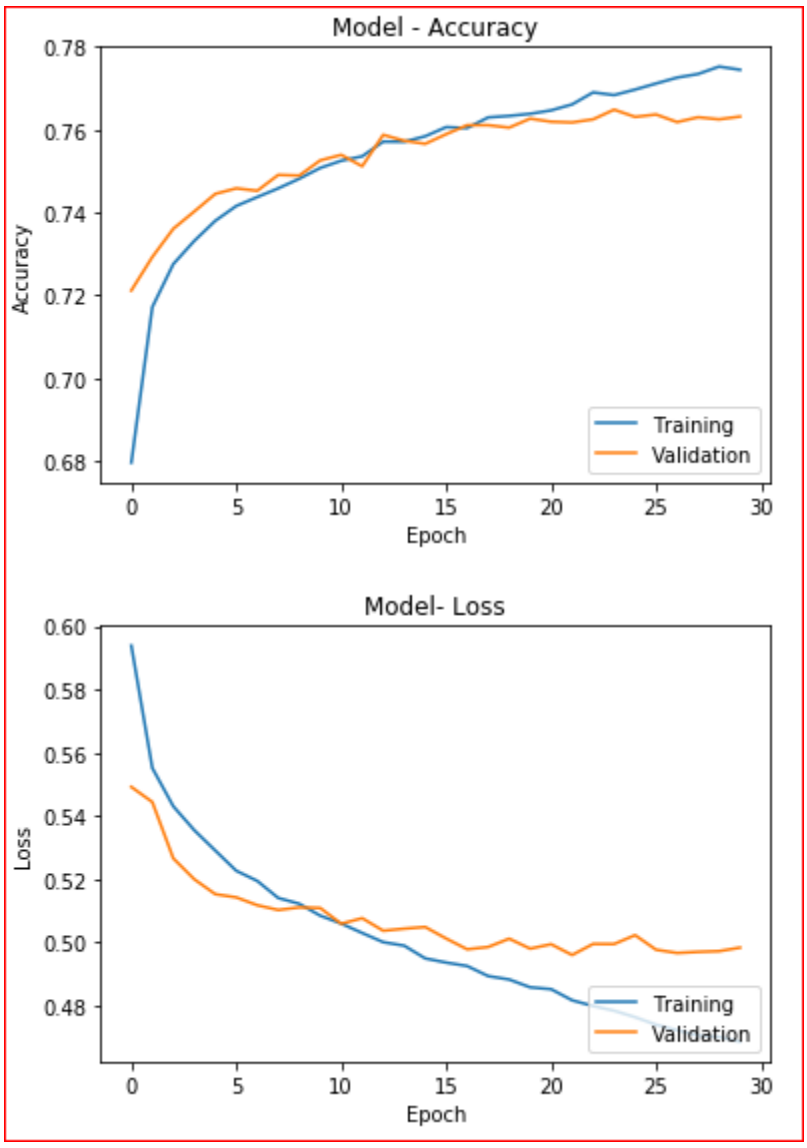
Homework 4 Report

Professor Pei-Yuan Wu
EE5184 - Machine Learning

Problem 1. (0.5%) 請說明你實作之 RNN 模型架構及使用的 word embedding 方法,回 報模型的正確率並繪出訓練曲線 *。(0.5%) 請實作 BOW+DNN 模型,敘述你的模型架構, 回報正確率並繪出訓練曲線。

A. RNN
   - 甲、 Word embedding
     i. 將 training data 及 testing data 一起做 embedding
     ii. 參數部分只設 size = 250, sg = 1
   - 乙、 模型架構
     i. Embedding layer
     ii. Bidirectional(GRU(units=250, activation='relu', recurrent_activation='hard_sigmoid', return_sequences=True, dropout=0.5, recurrent_dropout=0.5))
     iii. GRU(units=250, activation='relu', recurrent_activation='hard_sigmoid', dropout=0.5, recurrent_dropout=0.5)
     iv. Dense(250, activation='relu'
     v. Dropout(0.5)
     vi. Dense(1, activation='sigmoid')
   - 丙、 Kaggle 分數:0.75980
   - 丁、 訓練曲線



B. BOW+DNN

甲、Dictionary

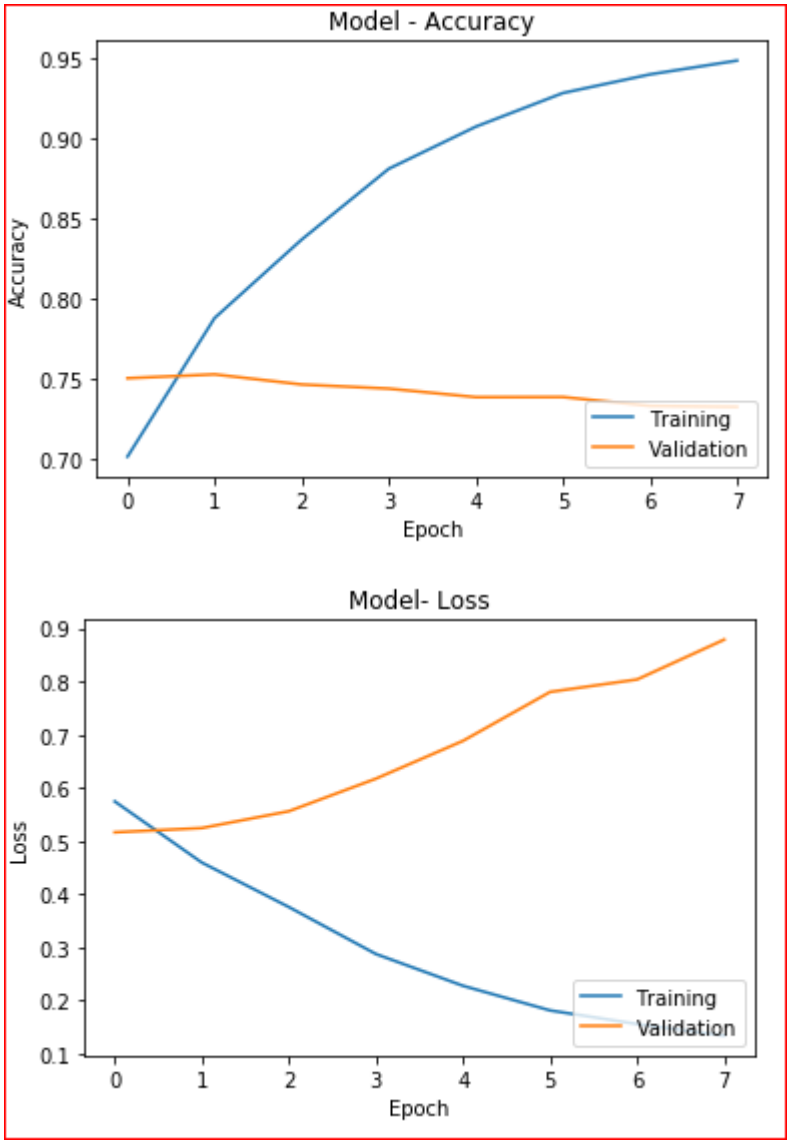
i. Training data 及 testing data 一起做成一個 dictionary

乙、模型架構

i. Dense(units=250, input_shape=(vocab_size + 1,), activation='relu')
ii. Dropout(0.5)
iii. Dense(units=250, activation='relu')
iv. Dropout(0.5)
v. Dense(units=250, activation='relu')
vi. Dropout(0.5)
vii. Dense(1, activation='sigmoid')

丙、Kaggle 分數:0.75322

丁、訓練曲線



* 訓練曲線 (Training curve):顯示訓練過程的 loss 或 accuracy 變化。橫軸為 step 或 epoch,縱軸為 loss 或 accuracy。

Problem 2. (1%) 請敘述你如何 improve performance(preprocess, embedding, 架構等), 並解釋為何這些做法可以使模型進步。

A. 一開始 word2vec 只用 training data, 參數為 sg=0 (default), 隨便疊 embedding+ 5 層 LSTM 發現跑超久，最後 shutdown
B. 改疊兩層 LSTM+sigmoid => kaggle 0.73697
C. 經過一連串亂試發現常常 overfitting, 後來有明顯進步在 LSTM 改成兩層 GRU+ReLu => kaggle 0.74660

D. 下一次成長在 word2vec 用 training data + testing data 且參數 sg=1, 我自己覺得加入 testing data 讓資料量增加可能是進步的原因, sg 也有單獨試過, 設成 1 在我的 case 裡面效果較好 => kaggle 0.75450
E. 加入 Bidirectional 並且在 GRU 加入 dropout 以及多一層 dense, 個人覺得一段文字雙向判讀可能會有加強的效果所以才這樣做, dropout 是為了 overfitting => kaggle 0.76130

Problem 3. (1%) 請比較不做斷詞 (e.g., 以字為單位) 與有做斷詞,兩種方法實作出來的 效果差異,並解釋為何有此差別。

有做斷詞 kaggle 分數: 0.76130
不做斷詞 kaggle 分數: 0.76112

此結果為相同架構下有無斷詞，這個結果出乎我預料，我原本覺得沒有斷詞應該會蠻不好的，我個人覺得應該是 RNN 本身在 learn 的時候就會考慮到前後文。

Problem 4. (1%) 請比較 RNN 與 BOW 兩種不同 model 對於" 在說別人白痴之前,先想 想自己" 與" 在說別人之前先想想自己,白痴" 這兩句話的分數(model output),並討論造 成差異的原因。

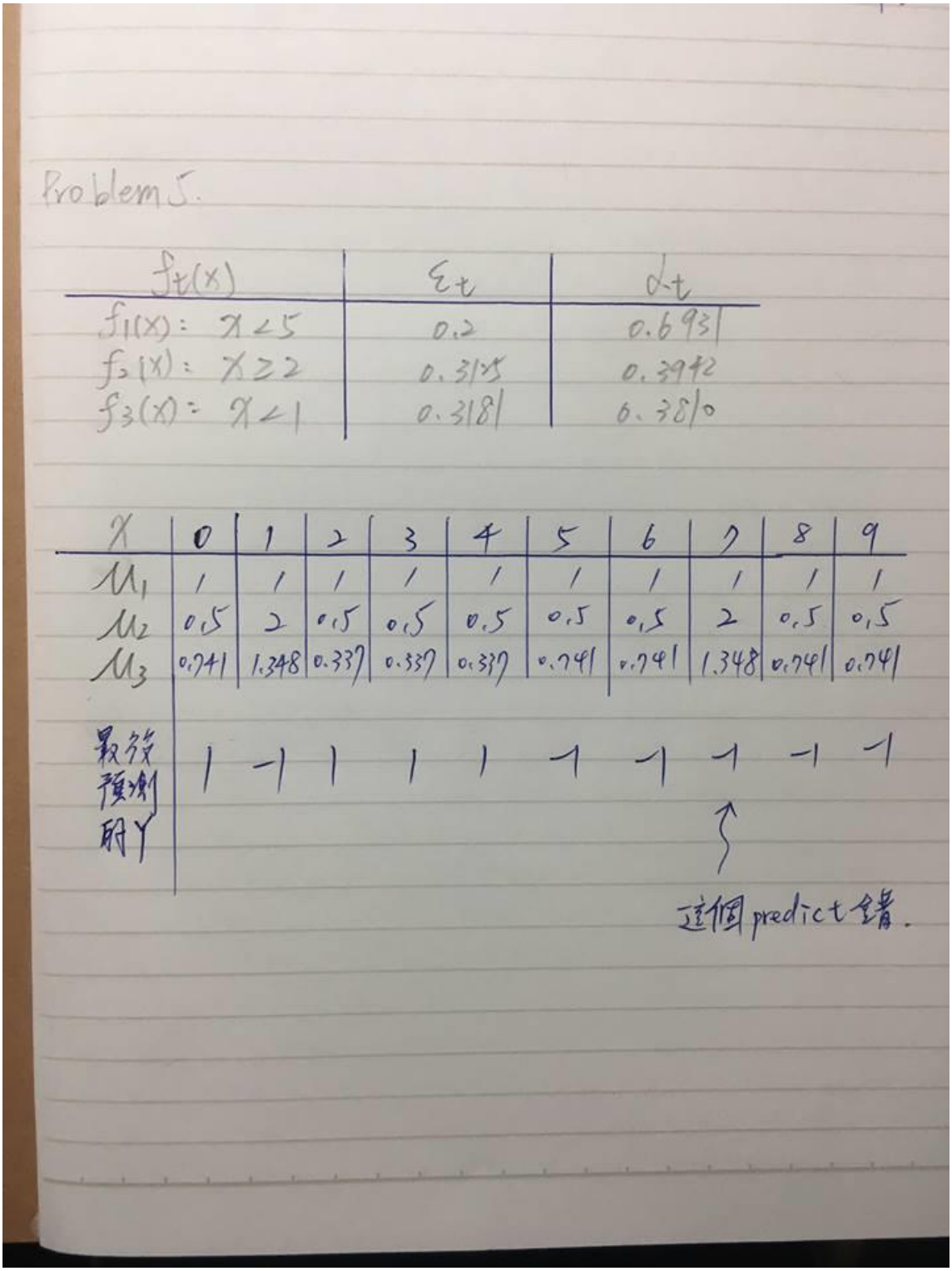
A. RNN
   甲、 在說別人白痴之前,先想 想自己: 0.5220395
   乙、 在說別人之前先想想自己,白痴: 0.6081922

B. BOW
   甲、 在說別人白痴之前,先想 想自己: 0.501505
   乙、 在說別人之前先想想自己,白痴: 0.50446206

1. 用 RNN 會因為"白癡"前後字的不同而有不同的分數, 可以看出後者比較接近惡意言語
2. 理論上 BOW 字的數量一樣應該會的到一樣的結果, 後來單獨試了將第一句的中間的空白去掉後就會的到兩個相同的分數了, 個人覺得空白字元被判定為"other"

Problem 5. (1%)

利用程式跑出來的結果



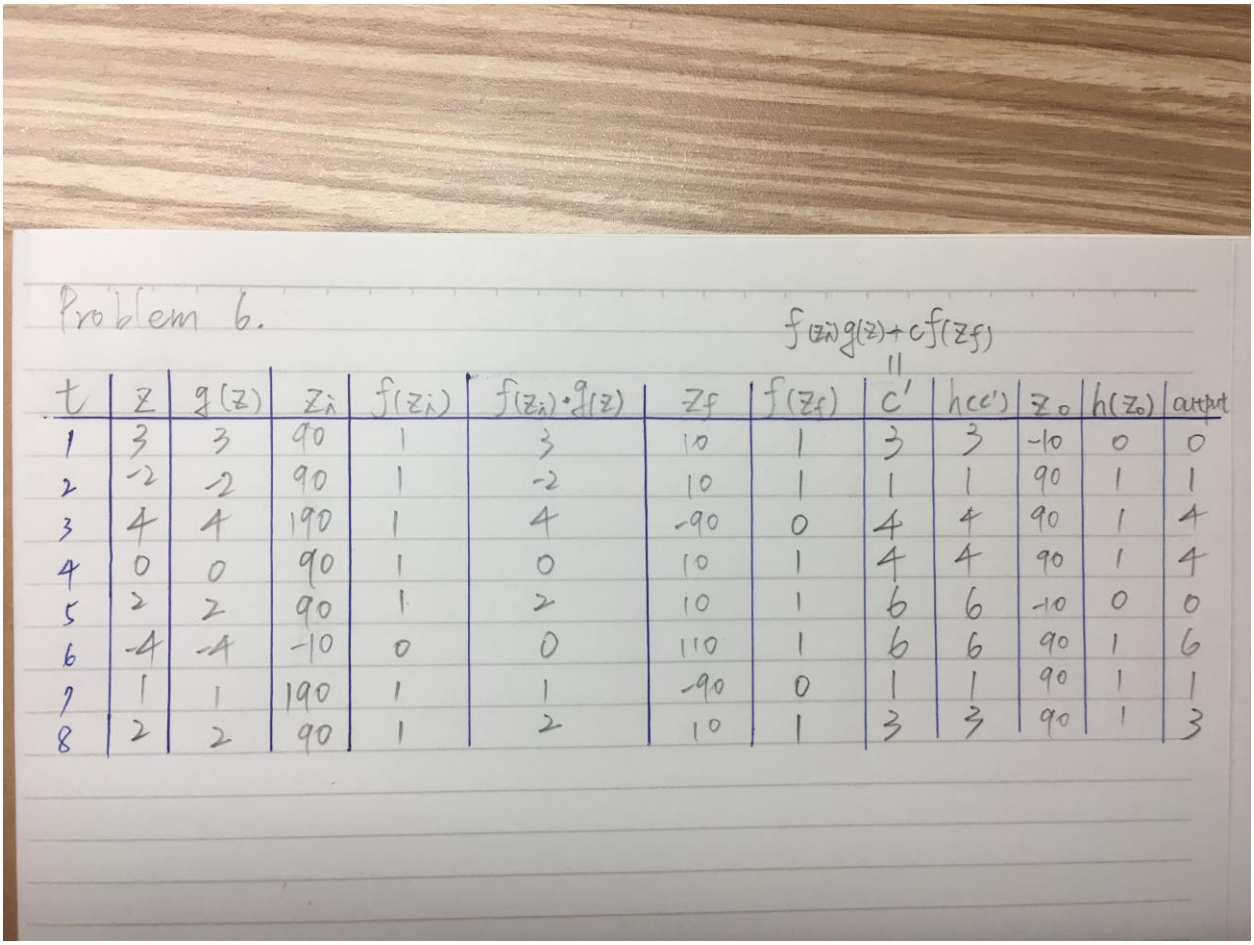
Problem 5.

$f_t(x)$	ε_t	α_t
$f_1(x)$: $x<5$	0.2	0.6931
$f_2(x)$: $x\geq 2$	0.3125	0.3942
$f_3(x)$: $x<1$	0.3181	0.3810

x	0	1	2	3	4	5	6	7	8	9
M_1	1	1	1	1	1	1	1	1	1	1
M_2	0.5	2	0.5	0.5	0.5	0.5	0.5	2	0.5	0.5
M_3	0.741	1.348	0.337	0.337	0.337	0.741	0.741	1.348	0.741	0.741
最終預測	1	-1	1	1	1	-1	-1	-1	-1	-1
用Y								↑		

這個 predict 錯.

Problem 6. (1%)



Problem 6.

$f(z_i)g(z)+cf(z_f)$

$= c'$

t	z	g(z)	z_i	$f(z_i)$	$f(z_i)\cdot g(z)$	z_f	$f(z_f)$	c'	$h(c')$	z_o	$h(z_o)$	output
1	3	3	90	1	3	10	1	3	3	-10	0	0
2	-2	-2	90	1	-2	10	1	1	1	90	1	1
3	4	4	190	1	4	-90	0	4	4	90	1	4
4	0	0	90	1	0	10	1	4	4	90	1	4
5	2	2	90	1	2	10	1	6	6	-10	0	0
6	-4	-4	-10	0	0	110	1	6	6	90	1	6
7	1	1	190	1	1	-90	0	1	1	90	1	1
8	2	2	90	1	2	10	1	3	3	90	1	3

備註：手寫部分與組員 R05621110 共同討論