Homework 4 Report Problem Set

Professor Pei-Yuan Wu
EE5184 - Machine Learning

電子所碩一 R07943004 莊育權

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。

* 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。

A. RNN + Word Embedding

1. 模型架構

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 40, 128)	4926848
gru_1 (GRU)	(None, 40, 512)	984576
gru_2 (GRU)	(None, 512)	1574400
dense_1 (Dense)	(None, 512)	262656
dense_2 (Dense)	(None, 512)	262656
dense_3 (Dense)	(None, 1)	513

Total params: 8,011,649
Trainable params: 3,084,801
Non-trainable params: 4,926,848

整個 RNN 架構第一層為 embedding layer，將 input sequence 轉換到 word embedding 的 vector 上，之後接兩層 512 的 GRU 與兩層 512 的 Dense layer，最後接到 output layer。

2. Word Embedding

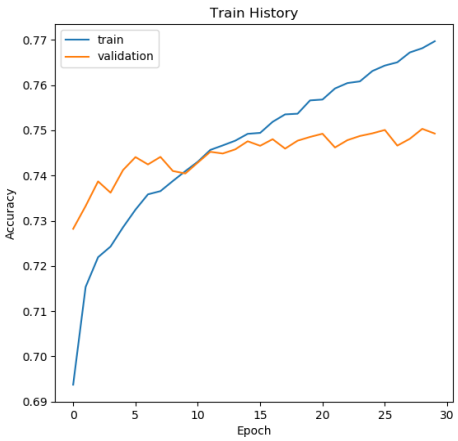
這邊運用到 gensim 裡面的 word2vec 函式，將 train_x.csv 和 test_x.csv 一起去做 word embedding，這裡的 window = 10、min count = 5、embedding dimension = 128。訓練完成後，將訓練好的 model 存進 embedding matrix 裡面給 embedding layer 使用。

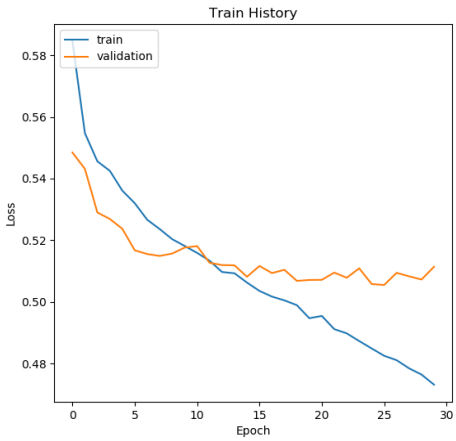
3. 訓練方式

資料前處理有先將 stop words 去除掉。訓練的時候將前 24000 筆資料當成 validation set，後 96000 當成 training set，並在模型訓練過程中加上 callbacks 函式中的 ModelCheckpoints，根據每個 epoch 訓練出來的 validation accuracy 將最好的 model 存取下來。

4. 訓練曲線

此 RNN 模型訓練 30 個 epoch 之後，訓練中最好的 validation accuracy 是 0.75033，而 kaggle public score 為 0.75377。





B. DNN + BOW

1. 模型架構

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 512)	5120512
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 512)	262656
dropout_2 (Dropout)	(None, 512)	0
dense_3 (Dense)	(None, 512)	262656
dropout_3 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 512)	262656
dropout_4 (Dropout)	(None, 512)	0
dense_5 (Dense)	(None, 1)	513

Total params: 5,908,993
Trainable params: 5,908,993
Non-trainable params: 0

整個 DNN 架構為四層 512 的 Dense layer，每層後面配上一層 Dropout layer = 0.5。

2. Bag of Word

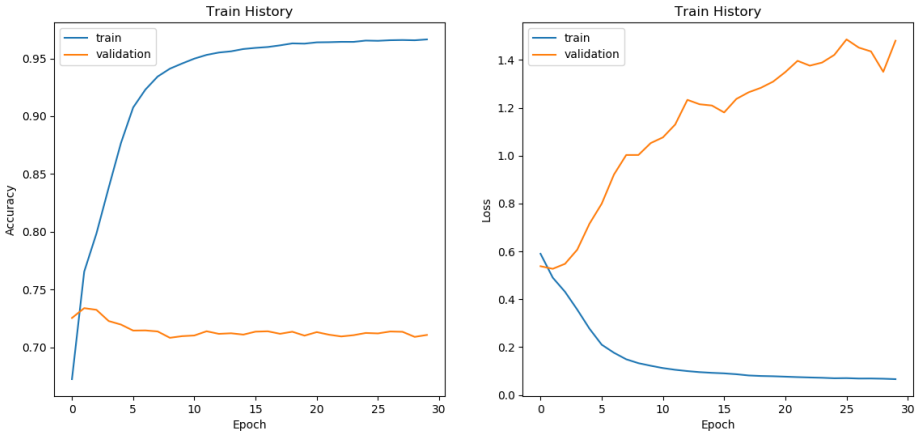
這邊運用到 keras 裡面的 tokenizer 將訓練資料做 BOW

3. 訓練方式

資料前處理有先將 stop words 去除掉。訓練的時候將前 24000 筆資料當成 validation set，後 96000 當成 training set，並在模型訓練過程中加上 callbacks 函式中的 ModelCheckpoints，根據每個 epoch 訓練出來的 validation accuracy 將最好的 model 存取下來。

4. 訓練曲線

此 DNN 模型訓練 30 個 epoch 之後，訓練中最好的 validation accuracy 是 0.73388，而 kaggle public score 為 0.73440。可以從訓練曲線中發現 overfitting 的現象。



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. Data Preprocessing

在將斷詞後的資料丟進去 Word Embedding 之前，有先將資料中的 stop words 去除掉。Stop words 就是中文裡對於判斷語意上是沒甚麼用的詞，例如你、我、他。去除掉 stop words 可以使得模型在判斷一段話的語意的時候，比較不會受到無意義的詞所影響。

B. Word Embedding

在做 Word Embedding 的時候，主要有調兩個參數分別是 window 和 embedding dimension。Window 這個參數主要是在做 Word embedding 的時候，會看上下文的範圍。這邊做實驗發現設成 10 會比設成 5 的準確率約高 1%，代表看上下文的範圍比較廣的確對於這個 Case 是比較有效的。Embedding dimension 是將詞投射到幾維空間的參數，這邊做實驗發現設成 128 會比 64 的準確來得高，代表這些詞在高維空間比較能代表彼此之間的相對關係及意義。

C. Model

這邊 RNN 模型架構試了 GRU 和 LSTM 兩種模型，整個架構同樣是兩層 512 RNN 加上兩層 512 DNN。發現 GRU 的準確率會比 LSTM 好一些，推測是 GRU 的模型架構相對於 LSTM 來說複雜度較低，比較不容易 overfitting。

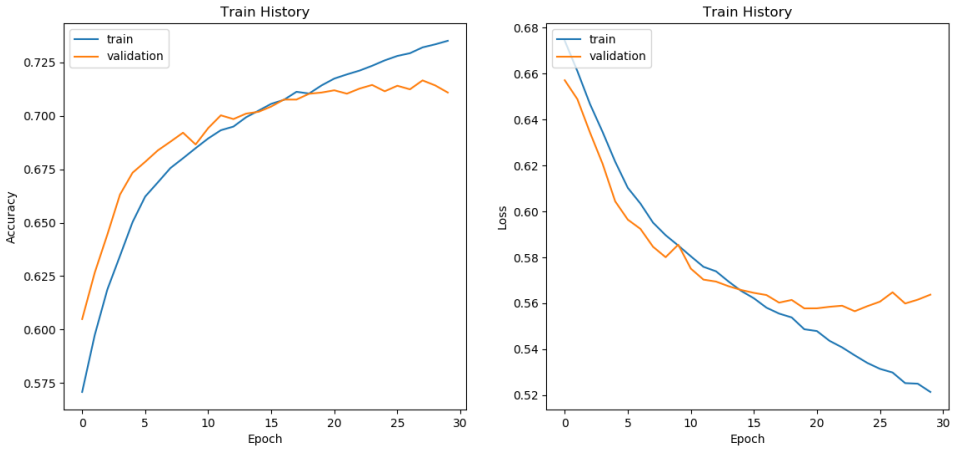
Problem 3. (1%) 請比較不做斷詞 (e.g., 以字為單位) 與有做斷詞，兩種方法實作出來的效果差異，並解釋為何有此差別。

這邊主要是在 Word segmentation 的時候是以字為單位去做切割，而不是去調用 jieba 的函式，其餘像是去除 stop words、word embedding、模型架構、訓練方式，都與第一題的 RNN+Word Embedding 相同。

可由下表看出以字為單位的訓練出來的準確率比用 jieba 函式的準確率來低得多，推測是因為 jieba 函式有將有意義的"詞"取出來，而如果只有字去做訓練，

就無法得知兩個字何在一起詞的意義，例如水果，被分成"水"和”果”的意思就完全不一樣，因此以字為單位訓練出來的才會比較差。

	以字為單位	用 jieba 函式
Validation Accuracy	0.71654	0.75033



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。

這邊所使用的 RNN 和 DNN 的與第一題的模型架構以及處理原則相同。

在 RNN 的部分可以看出兩者 output 其實是差不多的，我去看 jieba 切割出來的詞句分別會是「說 白痴 之前 先 想想 」和「說 之前 先 想想 白痴」，主要只差在白痴的位置不一樣，對於我們而言雖然感覺後者比較惡意，但或許對於 RNN 而言，這兩句其實在講的事情都是差不多的，於是分數相近。

在 DNN 的部分，由於用 jieba 分割出來的詞除了順序之外都是一樣的，因此這兩句 BOW 出來的數值都會一樣，所以出來的 output 數值也才一模一樣。

以這兩個句子而言，會發覺 DNN 的分數會比 RNN 還來的要高，猜測是因為這兩句話在詞句的順序關係其實是不怎麼重要，對於 DNN 而言，只要句子出現”白痴”就屬於分數比較高的惡意留言，因此即使 RNN 考慮了詞句順序也不見得會比沒有考慮詞句順序的 RNN 好。

	在說別人白痴之前，先想想自己	在說別人之前先想想自己， 白痴
RNN + Word Embedding	0.53272563	0.5022071
DNN + BOW	0.63255614	0.63255614

Problem 5. (1%) In this exercise, we will train a binary classifier with AdaBoost algorithm on the data shown in the table. Please use decision stump as the base classifier. Perform AdaBoost algorithm for T = 3 iterations. For each iteration (t = 1, 2, 3), write down the weights u_t^n used for training, the weighted error rate ϵ_t, scaling coefficient α_t, and the classification function $f_t(x)$. The initial weights u_1^n are set to 1 (n = 0, 1, ..., 9). Please refer to the course slides for the definitions of the above notations. Finally, combine the three classifiers and write down the final classifier.

x	0	1	2	3	4	5	6	7	8	9
y	+	-	+	+	+	-	-	+	-	-

藍色網底表示分類成+，橘色網底表示分類成-，紅字代表分類錯誤

Iteration 1:

$f_1(x)$, $\epsilon_1 = 0.2$, $d_1 = 2$, $\alpha_1 = 0.693$										
x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
y	+	-	+	+	+	-	-	+	-	-
u_1^n	1	1	1	1	1	1	1	1	1	1

Iteration 2:

$f_2(x)$, $\epsilon_2 = 0.3125$, $d_2 = 1.48$, $\alpha_2 = 0.394$										
x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
y	+	-	+	+	+	-	-	+	-	-
u_2^n	0.5	2	0.5	0.5	0.5	0.5	0.5	2	0.5	0.5

Iteration 3:

$f_3(x)$, $\epsilon_3 = 0.319$, $d_3 = 1.46$, $\alpha_3 = 0.378$										
x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
y	+	-	+	+	+	-	-	+	-	-
u_3^n	0.74	1.35	0.34	0.34	0.34	0.74	0.74	1.35	0.74	0.74

Final:

$f(x) = 0.693f_1(x) + 0.394f_2(x) + 0.378f_3(x)$										
x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
y	+	-	+	+	+	-	-	+	-	-

Problem 6. (1%) In this exercise, we will simulate the forward pass of a simple LSTM cell. Figure.1 shows a single LSTM cell, where z is the cell input, z_i, z_f, z_o are the control inputs of the gates, c is the cell memory, and f, g, h are activation functions. Given an input x, the cell input and the control inputs can be calculated by

$$z = w \cdot x + b$$

$$z_i = w_i \cdot x + b_i$$

$$z_f = w_f \cdot x + b_f$$

$$z_o = w_o \cdot x + b_o$$

Where w, w_i, w_f, w_o are weights and b, b_i, b_f, b_o are biases. The final output can be calculated by

$$y = f(z_o)h(c')$$

where the value stored in cell memory is updated by

$$c' = f(z_i)g(z) + cf(z_f)$$

Given an input sequence x^t (t = 1, 2, ..., 8), please derive the output sequence y^t. The input sequence, the weights, and the activation functions are provided below. The initial value in cell memory is 0. Please note that your calculation process is required to receive full credit.

$w = [0, 0, 0, 1]$, $b = 0$
$w_i = [100, 100, 0, 0]$, $b_i = -10$
$w_f = [-100, -100, 0, 0]$, $b_f = 110$
$w_o = [0, 0, 100, 0]$, $b_o = -10$

t	1	2	3	4	5	6	7	8
x^t	0	1	1	0	0	0	1	1
	1	0	1	1	1	0	1	0
	0	1	1	1	0	1	1	1
	3	-2	4	0	2	-4	1	2

$$f(z) = \frac{1}{1+e^{-z}} \qquad g(z) = z \qquad h(z) = z$$

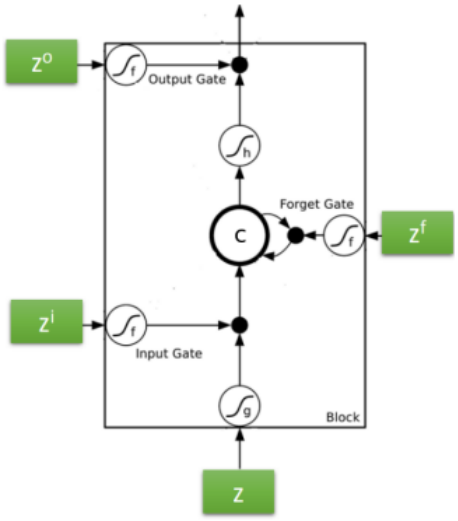


Figure 1: The LSTM cell

t = 1

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
3	90	10	-10	3	1	1	0	0	3	3	0

t = 2

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
-2	90	10	90	-2	1	1	1	3	1	1	1

t = 3

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
4	190	-90	90	4	1	0	1	1	4	4	4

t = 4

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
0	90	10	90	0	1	1	1	4	4	4	4

t = 5

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
2	90	10	-10	2	1	1	0	4	6	6	0

t = 6

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
-4	-10	110	90	-4	0	1	1	6	6	6	6

t = 7

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
1	190	-90	90	1	1	0	1	6	1	1	1

t = 8

z	z_i	z_f	z_o	$g(z)$	$f(z_i)$	$f(z_f)$	$f(z_o)$	c	c'	$h(c')$	y_1
2	90	10	90	2	1	1	1	1	3	3	3

t	1	2	3	4	5	6	7	9
y	0	1	4	4	0	6	1	3