學號：B07901169 系級：電機三　　　姓名：楊宗桓

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*

RNN 架構: 依序從上往下

embedding	Embedding(embedding.size(0),embedding.size(1))
Recurrent (lstm)	LSTM(embedding_dim =250, hidden_dim = 150 , num_layers = 1, batch_first=True)
linear	Dropout(), Linear(150 , 100, bias=True), BatchNorm1d(100), ReLU(), Dropout(), Linear(100, 100, bias=True), BatchNorm1d(100), ReLU()
classifier	Dropout(), Linear(100, 1), Sigmoid())

Embedding 方法:

參考: https://monkeylearn.com/sentiment-analysis/

我運用 spcay 來對文字作 preprocess，採用的 model 是 en_core_web_lg

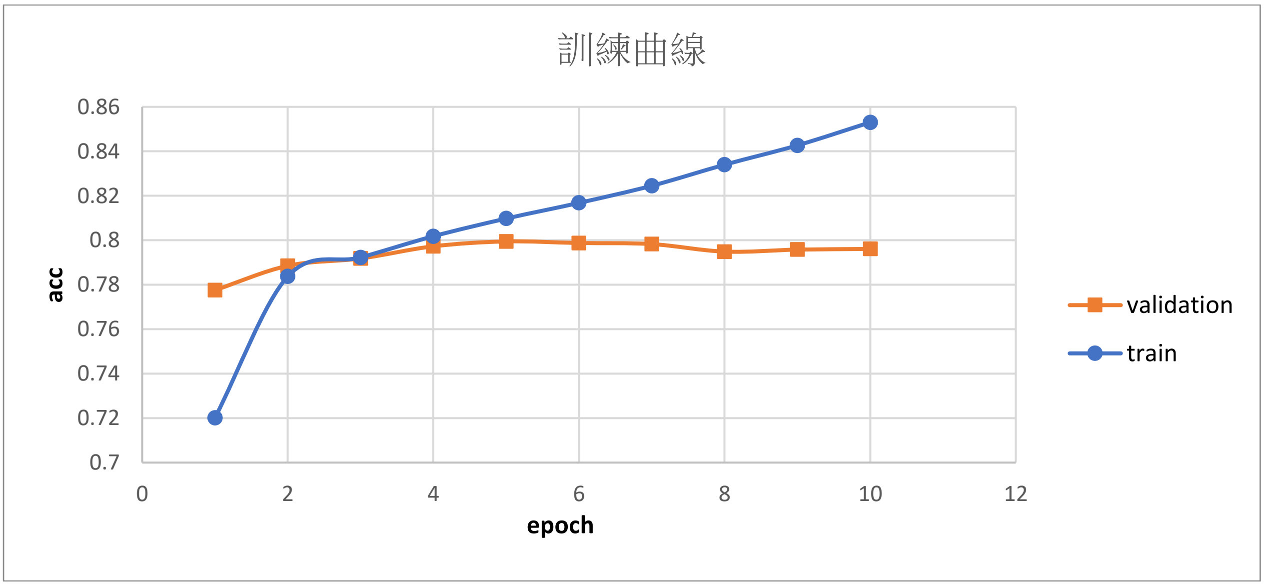
首先是清掉所有標點符號、stopword(ie 無意義文字)，再將所有字都轉為型(token.lemma_)

最後是依照助教給的 code 作 padding、deleting 到相同的文字長度。

作為預先處理後再用套件 word2vec model 訓練，參數為

Word2Vec(data, size=250, window=5, min_count=5, workers=12, iter=10, sg=1)

模型訓練曲線:



Kaggle 正確率: 0.79580

2.(0.5%) 請實作 BOW+DNN 模型，敘述你的模型架構，回報模型的正確率並繪出訓練曲線 * 。

BOW:

一樣先用 spcay 來對文字作 preprocess，採用的 model 是 en_core_web_lg

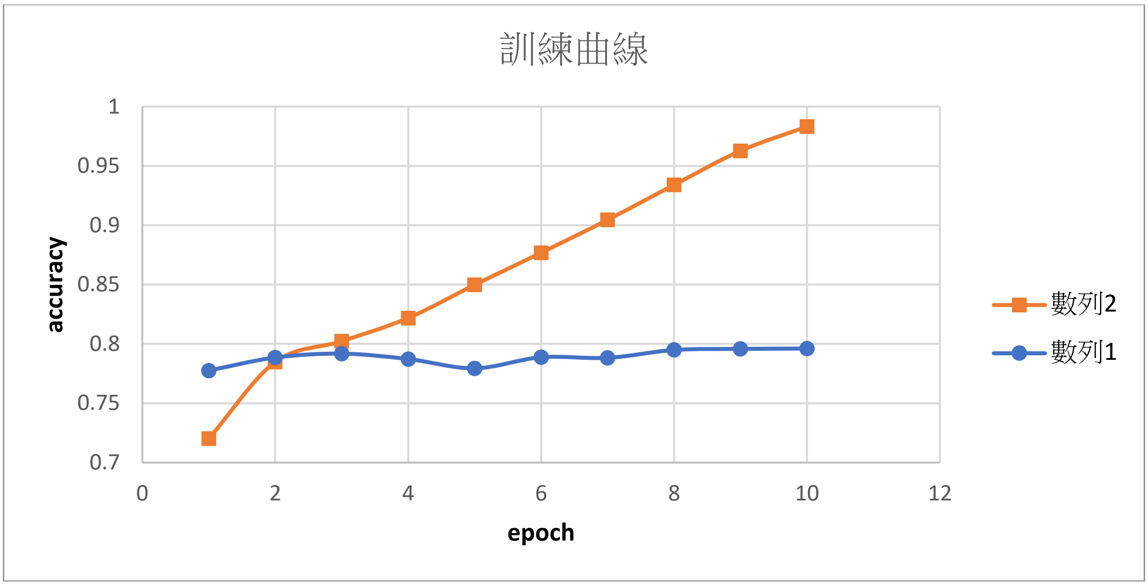
首先是清掉所有標點符號、stopword(ie 無意義文字)，再將所有字都轉為型(token.lemma_)

最後再統計有哪些字並存起來。

DNN 架構:

Linear	(81642 是來自 bow 的長度) Linear(81642 , 100 , bias = True) , BatchNorm1d(100) , .ReLU() , Dropout() , Linear(100 , 100 , bias = True) , BatchNorm1d(100) , ReLU() , Dropout() , Linear(100 , 100 , bias = True) , BatchNorm1d(100) , ReLU() , Dropout() , Linear(100 , 100 , bias = True) , BatchNorm1d(100) , ReLU()
classifier	Dropout(), Linear(100, 1), Sigmoid()

訓練曲線



Kaggle 正確率: 0.76548

3.(0.5%) 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。

參考: https://monkeylearn.com/sentiment-analysis/

用 spacy 的作 preprocess 可增加正確率

(1) 標點符號和 stop word 對文意幾乎沒有影響，將他們刪除不僅不會影響模型準確率，還可以使模型需要讀的 data 下降，增加訓練的效率。

(2) 用 token.lemma_將每個詞轉成原型: 不同詞可能因為文法而有不同的變化，但是他們在文句中帶有的意思是一樣，若是將他們視為不同數據輸入模型，將會使他們經過模型後得到的分數不同而不合理，所以將他們轉成原型可以增進效率。另外也可以透過這種方法降低詞彙的總數，減少需要讀的 data

4.(0.5%) 請比較 RNN 與 BOW 兩種不同 model 對於 "Today is hot, but I am happy" 與 "I am happy, but today is hot" 這兩句話的分數（model output），並討論造成差異的原因。

	"Today is hot, but I am happy"	"I am happy, but today is hot"
RNN	0.9138	0.7644
BOW+DNN	0.9846	0.9846

越趨近 1 代表該 sentence 越正向

由此可看出 RNN 會因句子中詞的先後順序不同而有不同的分數，但是 BOW 因為是統計詞彙的數量，兩個句子經過 BOW 後是得到相同的 vector 所以經過模型後會得到相同的分數，而使得 BOW+DNN 無法判別兩個句子的差異

5.(3%)Math problem:

https://drive.google.com/file/d/1fEu87banB4s6Yjku1dA5sMcnwCugEPBF/view?usp=sharing

1

1.

(a).

$x_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$ $x_2 = \begin{bmatrix} 4 \\ 8 \\ 5 \end{bmatrix}$ ..... $x_{10} = \begin{bmatrix} 10 \\ 11 \\ 7 \end{bmatrix}$

以下計算都用 numpy 輔助

$u = \frac{1}{10} \sum_{i=1}^{10} x_i = \begin{bmatrix} 5.4 \\ 8 \\ 4.8 \end{bmatrix}$

$\underset{\text{(covariance Matrix)}}{\Sigma} = \begin{bmatrix} 12.04 & 0.5 & 3.28 \\ 0.5 & 12.2 & 2.9 \\ 3.28 & 2.9 & 8.16 \end{bmatrix}$

Eigen vectors: $\begin{bmatrix} 0.39986 & -0.67818 & -0.61659 \\ 0.33759 & 0.73479 & -0.58881 \\ -0.85214 & -0.02729 & -0.52260 \end{bmatrix}$

Corresoponding Eigen values: 5.47203 11.63052 15.29744

The Eigen vectors of covariance Matrix is the principal axes.

that is:

$\underset{v_1}{\begin{bmatrix} -0.61659 \\ -0.58881 \\ -0.52260 \end{bmatrix}}, \underset{v_2}{\begin{bmatrix} -0.67818 \\ 0.73439 \\ -0.02729 \end{bmatrix}}, \underset{v_3}{\begin{bmatrix} 0.39986 \\ 0.33759 \\ -0.85214 \end{bmatrix}}$

Eigen Value 大→小 #

(b).

$x_i \xrightarrow[\text{component}]{\text{principle}} \left(x_i \cdot \left[v_1, v_2, v_3 \right] \right)^T$, denote x_i'

$x_1' = \begin{bmatrix} -3.36201 \\ 0.70874 \\ -1.48140 \end{bmatrix}$ $x_6' = \begin{bmatrix} -7.19137 \\ -1.83698 \\ 3.29720 \end{bmatrix}$

$x_2' = \begin{bmatrix} -9.78989 \\ 3.02598 \\ 0.07942 \end{bmatrix}$ $x_7' = \begin{bmatrix} -14.95793 \\ -0.47406 \\ -1.36988 \end{bmatrix}$

$x_3' = \begin{bmatrix} -13.61180 \\ -6.53657 \\ -2.41866 \end{bmatrix}$ $x_8' = \begin{bmatrix} -7.07758 \\ 3.81330 \\ 3.04814 \end{bmatrix}$

$x_4' = \begin{bmatrix} -9.93478 \\ 5.06051 \\ 1.16015 \end{bmatrix}$ $x_9' = \begin{bmatrix} -12.85888 \\ -3.95173 \\ 0.97350 \end{bmatrix}$

$x_5' = \begin{bmatrix} -12.36227 \\ 6.83599 \\ 5.02124 \end{bmatrix}$ $x_{10}' = \begin{bmatrix} -16.29378 \\ 1.10551 \\ 1.74703 \end{bmatrix}$

(c).

Average ~~Constructing~~ Error

(Preserving v_1, v_2 only)

$= \frac{1}{10} \sum_{i=1}^{10} \left| x_i - W^T (W x_i) \right|^2$

where $W = \begin{pmatrix} v_1^T \\ v_2^T \end{pmatrix}$

$= 6.06817$ #

2 (a)

(a)

① $\because (AA^T)^T = (A^T)^T A^T = AA^T$

$(A^TA)^T = A^T(A^T)^T = A^TA$

$\Rightarrow$ Both AA^T, A^TA are symmetric

② $\because \forall x \in R^m$,

$x^T(AA^T)x = (x^TA)\cdot(x^TA)^T = \|x^TA\|^2 \geq 0$

By definition, AA^T is positive semi-definite

$\forall y \in R^n$

$y^T(A^TA)y = (y^TA^T)(y^TA^T)^T = (Ay)^TAy = \|Ay\|^2 \geq 0$

By definition A^TA is positive semi-definite

③ let $\lambda \neq 0$ be eigen value of AA^T

$\Rightarrow (AA^T)v = \lambda v$ for $v \in R^m$ (eigen-vector)

$\because (A^TA)(A^Tv) = A^T(\underbrace{AA^Tv}) = A^T\lambda v$

let $u = A^Tv$ a eigen-vector

$(A^TA)u = \lambda u$ λ is also eigen value of A^TA

$\Rightarrow$ A^TA and AA^T share the same non-zero eigen-values #

(b).

let $y_1, y_2 \dots y_{2m} \in R^m$:

$y_1 = \begin{bmatrix} \sqrt{m} \\ 0 \\ 0 \end{bmatrix}$ $y_2 = \begin{bmatrix} -\sqrt{m} \\ 0 \\ 0 \end{bmatrix}$ $y_3 = \begin{bmatrix} 0 \\ \sqrt{m} \\ 0 \end{bmatrix}$ $y_4 = \begin{bmatrix} 0 \\ -\sqrt{m} \\ 0 \end{bmatrix}$ ......

2 (b)

$(A^TA)u = \lambda u$ $\quad$ λ is also eigen value of A^TA

$\Rightarrow A^TA$ and AA^T share the same non-zero eigen-values #

(b).

let $y_1, y_2, \dots, y_{2m} \in R^m$:

$$y_1 = \begin{bmatrix} \sqrt{m} \\ 0 \\ 0 \\ \vdots \end{bmatrix} \quad y_2 = \begin{bmatrix} -\sqrt{m} \\ 0 \\ 0 \\ \vdots \end{bmatrix} \quad y_3 = \begin{bmatrix} 0 \\ \sqrt{m} \\ 0 \\ \vdots \end{bmatrix} \quad y_4 = \begin{bmatrix} 0 \\ -\sqrt{m} \\ 0 \\ \vdots \end{bmatrix} \quad \cdots\cdots$$

$\because \Sigma$ is positive semi-definite

$\Rightarrow \exists A \in R^{m\times m}$ such that $\Sigma = AA^T$

and we can simply let $\underline{x_k = Ay_k + u}$

$$\text{mean}(x) = \frac{1}{2m}\sum_{k=1}^{2m}(Ay_k + u) = \frac{1}{2m}\,2m\,u = u$$

$$\text{Cov}(x) = \frac{1}{2m}\sum_{k=1}^{2m}(x_k - u)(x_k - u)^T$$

$$= \frac{1}{2m}\sum_{k=1}^{2m}(Ay_k + u - u)(Ay_k + u - u)^T$$

$$= \frac{1}{2m}\sum_{k=1}^{2m} Ay_k y_k^T A^T$$

$$\frac{1}{2m}\sum_{k=1}^{2m} Ay_k y_k^T A^T$$

$$= \frac{1}{2m} A\left(\sum_{k=1}^{2m} y_k y_k^T\right) A^T = A I_m A^T = AA^T = \Sigma$$

#

$\Downarrow$ $\quad\quad$ $\Uparrow$

$$\sum_{k=1}^{2m} y_k y_k^T = \begin{pmatrix} 2m & 0 & \cdots \\ & 0 & \\ \vdots & & 0 \end{pmatrix} + \begin{pmatrix} 0 & & \cdots \\ & 2m & \\ \vdots & & 0 \end{pmatrix} + \cdots = 2m I_m$$

(c).

$$\text{Trace}(\Phi^T \Sigma \Phi) = \text{Trace}(\Sigma(\Phi\Phi^T))$$

By Von Neumann's Trace Ineq.

$$\geq \sum_{i=1}^{m} \lambda_i u_{m-i+1}$$ where λ_i is eigenvalue of Σ (in decreasing

2 (c)

$$\sum_{k=1}^{2m} y_k y_k^T = \begin{pmatrix} 2m & & \\ & 0 & \\ & & 0 \end{pmatrix} + \begin{pmatrix} 0 & & \\ & 2m & \\ & & 0 \end{pmatrix} + \cdots = 2m I_m$$

(c)

$\mathrm{Trace}(\Phi^T \Sigma \Phi) = \mathrm{Trace}(\Sigma(\Phi\Phi^T))$

~~By Von NeuMann's Trace Ineq.~~

$\geq \sum_{i=1}^{m} \lambda_i \upsilon_{m-i+1}$ where λ_i is eigenvalue of Σ ~~(in decreasing order)~~

υ_m is eigenvalue of $\Phi\Phi^T$

let $\Phi = [v_1, v_2, \dots, v_k]$

$$\Phi\Phi^T = [v_1, v_2 \dots v_k] \begin{bmatrix} v_1^T \\ v_2^T \\ v_3^T \\ \vdots \end{bmatrix}$$

$$= \begin{bmatrix} v_1 v_1^T \\ v_2 v_2^T \\ \vdots \\ v_k v_k^T \end{bmatrix} = I_k \Rightarrow v_1, v_2, \dots v_k \text{ are orthonormal basis for } R^k$$

So $(\Phi\Phi^T) v_i = \Phi(\Phi^T v_i) = \Phi e_i = v_i$ v_i is eigenvector of $\Phi\Phi^T$ with eigenvalue $= 1$

~~let $W_1, W_2 \dots W_{m-k}$ be R^{m-k} orthonormal basis~~ (和v互補)

則 $(\Phi\Phi^T) W_i = \Phi(\Phi^T w_i) = 0$ W_i is eigenvector of $\Phi\Phi^T$ with eigenvalue $= 0$

$= \lambda_1 \upsilon_m + \lambda_2 \upsilon_{m-1} + \dots + \lambda_{m-k} \upsilon_{k+1} + \dots$

~~$= \lambda_{m-k+1} + \lambda_{m-k+2} + \dots + \lambda_m$~~

This minimum can be achieved by using $\Sigma = Q \Lambda Q^T$

(Eigen value decomposition)

2(c)續

If $Q = [A_1 A_2 \cdots A_m]$, and corresonding eigen value $\lambda_1 \ge \lambda_2 \ldots \lambda_m$.

simply take the corespoing eigen vectors of k-smallest eigen value to Φ:

that is $\Phi = [A_{m-k+1} A_{m-k+2} \ldots A_m]$

$Trace(\Phi^T \Sigma \Phi)$

$$= Trace\left(\begin{bmatrix} A_{m-k+1}^T \\ A_{m-k+2}^T \\ \vdots \\ A_m^T \end{bmatrix} \times \Sigma \times [A_{m-k+1} \ldots A_m]\right)$$

$\because \Sigma A_i = \lambda A_i$

$$= Trace\left(\begin{bmatrix} A_{m-k+1}^T \\ \vdots \\ A_m^T \end{bmatrix} \times [\lambda_{m-k+1} A_{m-k+1}, \lambda_{m-k+2} A_{m-k+2} \ldots \ldots \lambda_m A_m]\right)$$

$$= \lambda_{m-k+1} \|A_{m-k+1}\|^2 + \lambda_{m-k+2} \|A_{m-k+2}\|^2 + \cdots + \lambda_m \|A_m\|^2$$

$\because \|A_i\|^2 = 1$

$$= \lambda_{m-k+1} + \cdots + \lambda_m$$

achieves the minimum #

3.

Gradient Boosting to Minimize $L(g_T^1 \cdots g_T^k)$:

$\Rightarrow$ Find $f_t \in F_c$ such that $\frac{\partial}{\partial \alpha} L$ is minimized

$$\Rightarrow g_{t-1}^1 + g_{t-1}^2 + g_{t-1}^3 + \cdots + g_{t-1}^{k-1} + \alpha f_t^k g_{t-1}^k +$$

$$g_{t-1}^1 + g_{t-1}^2 + g_{t-1}^3 + \cdots \alpha f_t^{k-1} g_{t-1}^{k-1} + g_{t-1}^k +$$

$$\vdots$$

Total $\Rightarrow$ Find f_t such that total is minimized

#