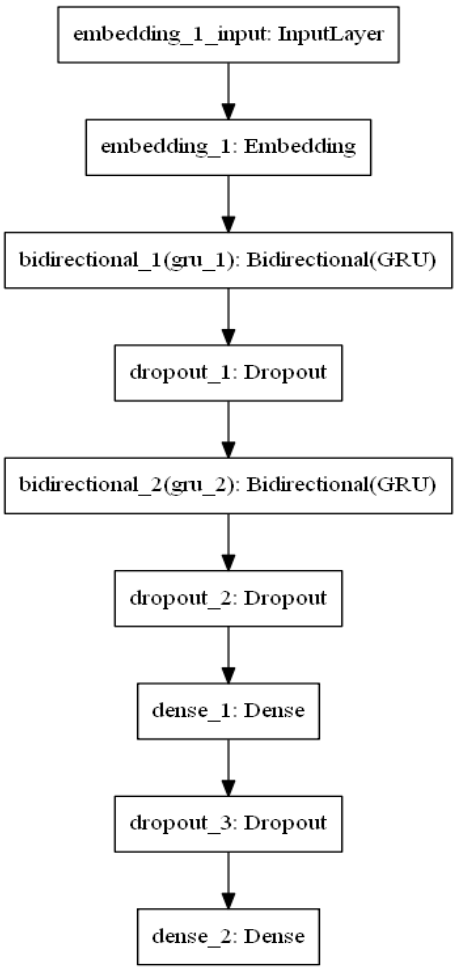
學號：B04902089 系級： 資工三　姓名：林政豪

1. (1%) 請說明你實作的 RNN model，其模型架構、訓練過程和準確率為何？
2. (1%) 請說明你實作的 BOW model，其模型架構、訓練過程和準確率為何？

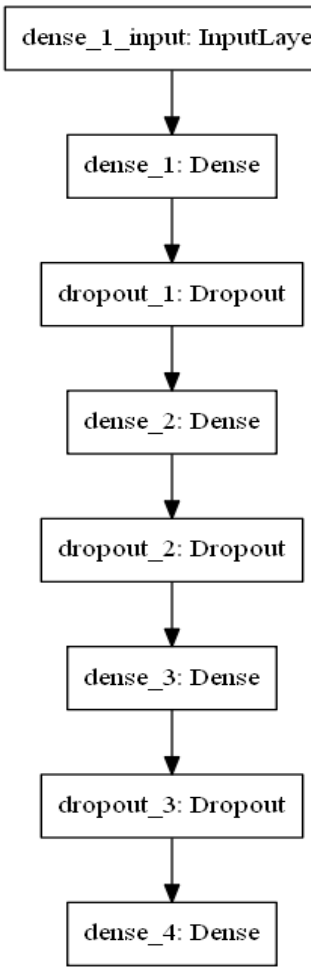
RNN



Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 55, 128)	3995648
bidirectional_1 (Bidirection	(None, 55, 256)	197376
dropout_1 (Dropout)	(None, 55, 256)	0
bidirectional_2 (Bidirection	(None, 256)	295680
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 128)	32896
dropout_3 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 1)	129

Total params: 4,521,729
Trainable params: 4,521,729
Non-trainable params: 0

BOW



Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 1024)	12289024
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 256)	262400
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 64)	16448
dropout_3 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 1)	65

Total params: 12,567,937
Trainable params: 12,567,937
Non-trainable params: 0

	RNN(acc)		BOW(acc)	
Epoch	Training	Validation	Training	Validation
1	0.7720	0.8056	0.7672	0.7871
2	0.8179	0.8153	0.8199	0.7952
3	0.8379	0.8202	0.8639	0.7915
4	0.8539	0.8174	0.9096	0.7867
5	0.8709	0.8167	0.9390	0.7869

	RNN dropout(0.4、0.4、0.5)	BOW dropout(0.5、0.5、0.5)
模型	把 label、no label、 testing 丟進 gensim 文字的處理是 encode 成 ascii 去掉無關字， 且沒有去掉標點符號。	把 label、no label、testing 丟進 keras 的 Tokenizer 文字處理使用 Tokenizer. texts_to_matrix

Optimizer 都是使用預設參數的 adam，loss function 都是 binary_crossentropy。

兩者都是先用 sklearn shuffle 過，拿十六萬筆作 Training，四萬筆作 Validation。

3. (1%) 請比較 bag of word 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"這兩句的情緒分數，並討論造成差異的原因。

	RNN	BOW
First sentence	0.36352733	0.58970451
Second sentence	0.73681247	0.58970451

RNN 會考慮字在句子中的順序，而基本的 BOW 則沒有考慮，故在交錯語序時 RNN 結果會不一樣，而不影響 BOW 的結果。

4. (1%) 請比較"有無"包含標點符號兩種不同 tokenize 的方式，並討論兩者對準確率的影響。

原本的 RNN 是將讀入字串 encode 成 ascii 藉此去掉無關符號，並留下標點。

而去掉標點的版本使用 gensim.utils.tokenize 去掉標點。

	RNN 有標點(acc)		RNN 無標點(acc)	
Epoch	Training	Validation	Training	Validation
1	0.7720	0.8076	0.7747	0.8070
2	0.8179	0.8183	0.8147	0.8155
3	0.8379	**0.8232**	0.8304	0.8198
4	0.8539	0.8194	0.8465	**0.8228**
5	0.8709	0.8177	0.8604	0.8185
Kaggle(紅色)	0.82442		0.82066	

起初我認為，標點符號(如驚嘆號、問號等等)，會使得附近的字對情緒表達會相對重要，去掉標點會影響 model 判斷，所以沒有去掉，由上表的可以觀察出我的推論正確，有標點符號的 RNN 不管在 Training、Validation、Kaggle 上都表現得比沒標點的好。

5. (1%) 請描述在你的 semi-supervised 方法是如何標記 label，並比較有無 semi-surpervised training 對準確率的影響。

我的 semi-supervised 是使用 train 好的 model 進行標記，如果輸出 >0.96 或<0.04 時才會標記加進 pseudo_label.txt，並於下一次訓練時加入 Training data，否則就保持原樣輸出到 left.txt，第二輪開始就從 left.txt 裡面讀取、標記。

所有 Trainging 之前先 shuffle 過一次，每次固定以同樣的四萬筆 train_label 中的資料作 validation，其餘都作為 Training data。

	單位	initial	Semi-1	Semi-2	Semi-3	Semi-4	Semi-5
Training data	筆資料	160,000	532,623	834,199	969,260	1,060,761	1,142,138
Epoch = 1	Val acc	0.80870	0.82250	0.82717	0.83225	0.83680	0.83910
Epoch = 2	Val acc	0.81527	0.83017	0.82825	0.84018	0.84288	0.84550
Epoch = 3	Val acc	0.81660	**0.83178**	**0.83593**	0.84075	**0.84547**	0.84847
Epoch = 4	Val acc	**0.82040**	0.82893	0.83445	**0.84205**	0.84465	**0.84877**
Epoch = 5	Val acc	0.81745	0.82747	0.83510	0.84186	0.84439	0.84800
Kaggle(紅色)	acc	0.82042	0.82292	0.82275	0.82395	0.82423	0.82438

由表可以看出，semi-supervise 經過越多輪，training data 就越多，且因為是之前 model 信心較高的結果，所以幾乎都是對的，會使得訓練結果變好，表中 Val acc 會越來越高，且每一輪的 Validation 為同一組，所以可以看出經過 semi-supervise 之後準確率會變高。