Machine Learning HW6 Report

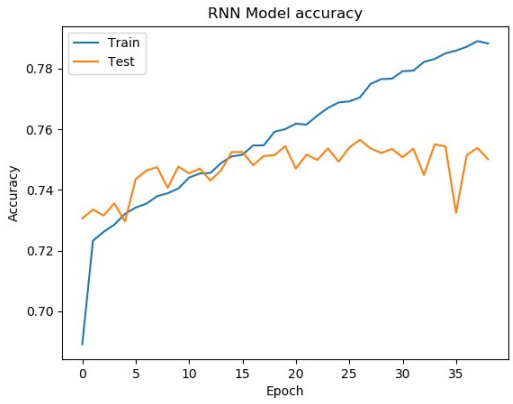
學號：B06902052 系級：資工二 姓名：張集貴

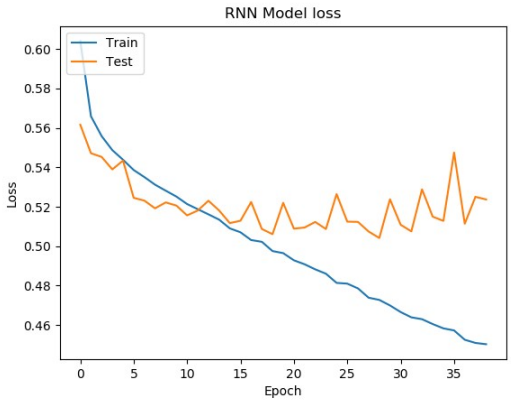
1. (1%) 請說明你實作之 RNN 模型架構及使用的 word embedding 方法，回報模型的正確率並繪出訓練曲線*

```
sentences = word2vec.LineSentence('../tot.cut')
model = word2vec.Word2Vec(sentences, size=300, iter=35, sg=1, workers=20)
```

使用 gensim 的 word2vec 以上圖的訓練參數訓練出 word embedding，再從其建構出 Keras 的 Embedding layer 所需的 weight matrix，再建構下圖的 RNN 模型。

```
model = Sequential()
wm = np.load('wm.npy')
model.add(Embedding(wm.shape[0], output_dim=wm.shape[1], weights=[wm], input_length=MAX_LENGTH, trainable=False))
model.add(Bidirectional(GRU(units=64, return_sequences=True, dropout=0.2, recurrent_dropout=0.2)))
model.add(Dropout(0.2))
model.add(Bidirectional(GRU(units=32, return_sequences=True, dropout=0.2, recurrent_dropout=0.2)))
model.add(Dropout(0.4))
model.add(Bidirectional(GRU(units=16, return_sequences=True, dropout=0.2, recurrent_dropout=0.2)))
model.add(Dropout(0.4))
model.add(Bidirectional(GRU(units=8, return_sequences=True, dropout=0.2, recurrent_dropout=0.2)))
model.add(Bidirectional(GRU(units=4, return_sequences=True, dropout=0.2, recurrent_dropout=0.2)))
model.add(Bidirectional(GRU(units=2, return_sequences=False, dropout=0.2, recurrent_dropout=0.2)))
model.add(Dense(units = 1, activation ='sigmoid'))
rms = optimizers.RMSprop(clipnorm=1.)
model.compile(optimizer = rms, loss = 'binary_crossentropy', metrics = ['accuracy'])
```



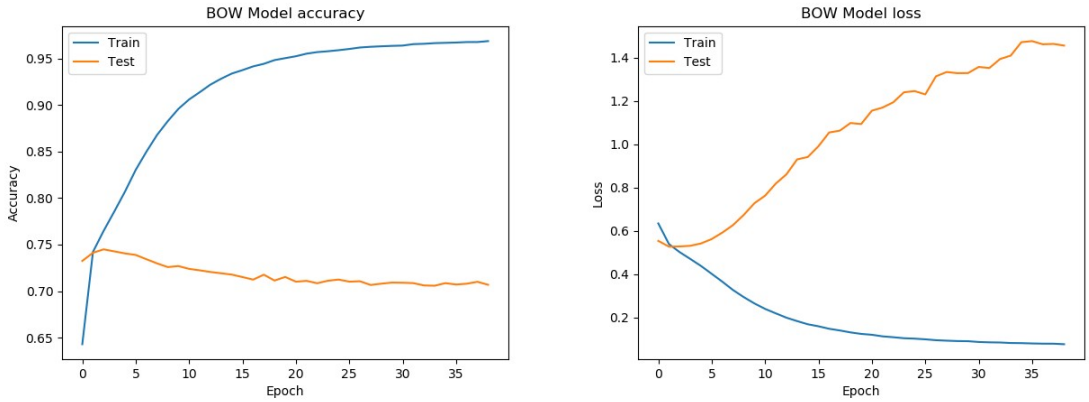


Kaggle Public Score	0.74980
Kaggle Private Score	0.75130

2. (1%) 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*。

預處理時，只保留詞頻最高的 MAXV=5000 個字，其他設為 Unknown，並以下圖的架構以及跟 RNN 一樣的訓練參數進行訓練。

```
model = Sequential()
model.add(Dense(units = 128, input_dim=MAXV+2,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(units = 64, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(units = 32, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(units = 16, activation='relu'))
model.add(Dense(units = 8, activation='relu'))
model.add(Dense(units = 4, activation='relu'))
model.add(Dense(units = 1, activation ='sigmoid'))
rms = optimizers.RMSprop(clipnorm=1.)
model.compile(optimizer = rms, loss = 'binary_crossentropy', metrics = ['accuracy'])
```



Kaggle Public Score	0.73950
Kaggle Private Score	0.73510

3. (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。

一開始訓練時，將訓練資料拆成 training set 跟 validation set， 並只用 training set 訓練，訓練出的模型稱為模型 1。
之後用全部的訓練資料跟一樣的訓練參數訓練出的模型稱為模型 2。
把模型 1 跟模型 2 輸出的機率相加除以二，稱為模型 3。

模型	1	2	3

Kaggle Public Score	0.74980	0.75120	0.75590
Kaggle Private Score	0.75130	0.74850	0.75220

可以發現雖然模型 1 跟模型 2 在 Public 跟 Private 的表現是逆序的，但在兩者模型 3 都是最好的。
模型 3 是 Ensemble learning 裡的 bagging 的應用，可以降低模型的 variance，因此能提昇準確率。

4. (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。

   不做斷詞的效果較差，正確率大概低了 1%，因為不做斷詞，模型要自行學習詞彙的意思，單字被斷開，意思可能改變，所以造成這種結果。

5. (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。

	RNN	BOW
"在說別人白痴之前，先想想自己"	0.6829186	0.60904586
"在說別人之前先想想自己，白痴"	0.76160294	0.60904586

BOW 的兩者分數一樣，因為他的資料表示方式不會保留順序，所以這兩句對他來說是一樣的句子。
第二句的語氣較第一句強烈，我自己會認為第二句較有惡意。
RNN 因為能考慮順序，所以給第一個句子的分數較第二個低，可能是他能學到如何判斷語氣的關係。