Machine Learning HW6 Report

學號：B06901063 系級：電機二 姓名：黃士豪

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*

   (1) 架構：

   GRU(64)→Dropout(0.25)→Dense(128)→Dropout(0.5)

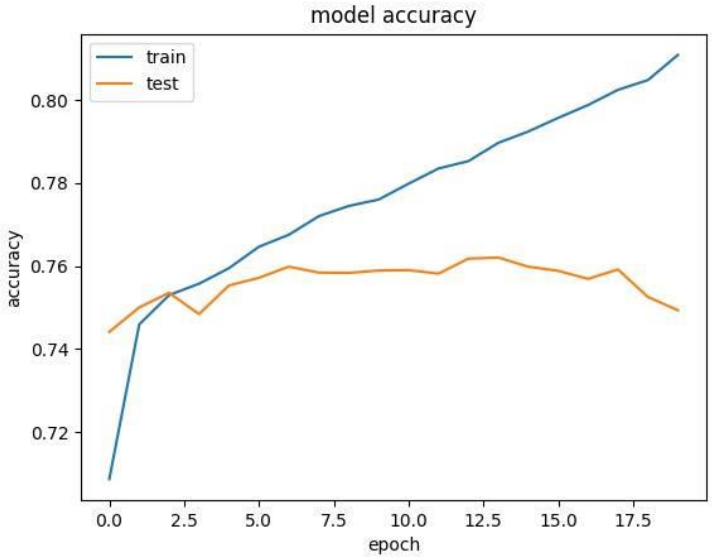
   →Dense(128)→Dropout(0.5)→Dense(1)

   (2) word embedding：

   建立一個包有 word2vec model 中所有詞的 dictionary，將每個 dic 中的詞建立編號與對應的 vector，此時只要將原 train/test data 中的詞預先轉換成這個詞所對應的編號，並於 training 時再改變成對應的 vector 即可。

   (3) 正確率

   public : 0.76050；private : 0.75290



   可以發現在前期有緩慢進步，但在後期有些下降。

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*。

   (1) 架構：

   Dense(128)→Dropout(0.25)→Dense(128)→Dropout(0.5)

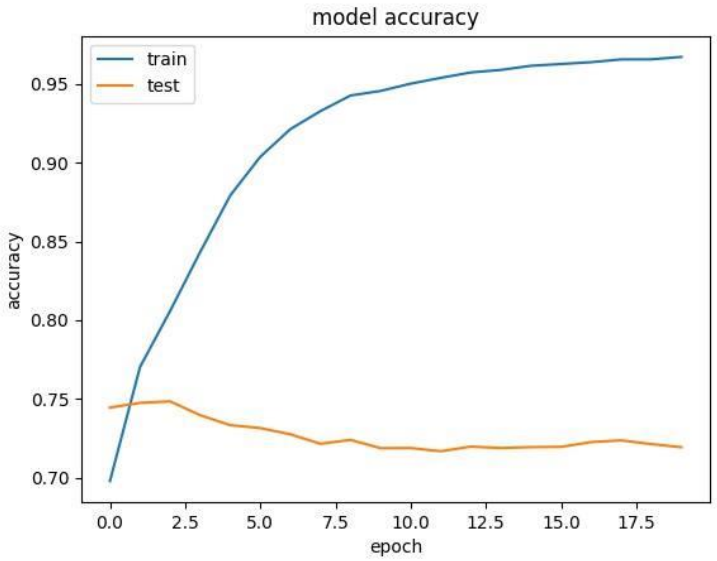
   →Dense(128)→Dropout(0.5)→Dense(1)

(2) word embedding：

藉由 word2vec model 取出出現頻率大於 40 的詞(約 6990 個)，並將其建構成 6991 個 element 的 vector，每個 element 代表一個詞，第一個則代表出現頻率不足的詞彙。每個句子對應的 vector 值相當於在此句子中每個詞出現的次數，最終送入 dnn 進行訓練。

(3) 正確率

public : 0.74080；private : 0.73980



可以發現一開始 acc 就衝得很高，但是之後就完全無法進步。

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。

   在一開始 train word2vec model 的部分，大部分的參數並不會有太多的影響，只有 sg 這個參數較重要，其中發現以 skip-gram 作為預測方法會有更好的結果，推測因為留言普遍較短，單一一個詞彙對周圍造成的影響會比周圍詞彙造成的影響大(因為周圍詞彙少)。embedding 的部分我使用與助教相同的方法，也是以編號對應的 vector 建表，在 training 時再轉換，可有效減少讀取時間。我一開始以為 rnn 架構也要跟 dnn 一樣需要一定深度，結果發現其實一層 LSTM 就能達到不錯的結果。後來，聽從別人的建議將 LSTM 改成 GRU，發現有不錯的進步，推測應該是因為 data 量與留言長度不夠，無法發揮出 LSTM 的優勢。

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。

	public accuracy	private accuracy
有做斷詞	0.76050	0.75290
不做斷詞	0.75010	0.74420

可以發現有做斷詞效果較好，原因是因為中文字有些字在單個字時是無意義的、或是意義不同的，若不做斷詞將會造成誤判或者是無法判讀。

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。

	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	0.42594057	0.5615986
BOW	0.7497933	0.7497933

可以由上表發現 RNN 可以明確看出第一句是不帶惡意但第二句是帶著惡意的，但 BOW 卻不行，原因是因為 RNN 會考慮前幾個詞的影響，但 BOW 不會，它只要看到相同的詞組成就當成是一樣的意思，因此才會在看到「白痴」這個詞之後將兩句話都判讀成帶有惡意。