B03705013 資管四 徐逸然

- Model description (2%)
  1. RNN :

     我做出來最好的 RNN 模型如下圖，四層雙向 GRU，從上到下分別有 512, 256, 128, 128 個 cell (由於都是雙向的固實際 cell 數量還要乘 2)，接著不通過任何額外的 dense layer 直接接上 output layer。

```
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_1 (Bidirection (None, 777, 1024)         1695744
_________________________________________________________________
bidirectional_2 (Bidirection (None, 777, 512)          1967616
_________________________________________________________________
bidirectional_3 (Bidirection (None, 777, 256)          492288
_________________________________________________________________
bidirectional_4 (Bidirection (None, 777, 256)          295680
_________________________________________________________________
time_distributed_1 (TimeDist (None, 777, 40)           10280
=================================================================
Total params: 4,461,608
Trainable params: 4,461,608
Non-trainable params: 0
_________________________________________________________________
None
Train on 3141 samples, validate on 555 samples
```

  2. RNN+CNN :

     我的 CNN 模型採用 CONV2D，每個 timestep 原本只有一個 39 維 mfcc features，我將其前後一個 timestep 的 feature 也加進來，reshape 成 39 * 3 的矩陣。也就是說每一個 timestep 可以視為一張(39*3)的照片，模型如下圖:

```
Layer (type)                 Output Shape              Param #
=================================================================
time_distributed_22 (TimeDis (None, 777, 37, 1, 16)    160
_________________________________________________________________
dropout_7 (Dropout)          (None, 777, 37, 1, 16)    0
_________________________________________________________________
time_distributed_23 (TimeDis (None, 777, 592)          0
_________________________________________________________________
bidirectional_19 (Bidirectio (None, 777, 1024)         3394560
_________________________________________________________________
bidirectional_20 (Bidirectio (None, 777, 512)          1967616
_________________________________________________________________
bidirectional_21 (Bidirectio (None, 777, 256)          492288
_________________________________________________________________
time_distributed_24 (TimeDis (None, 777, 40)           10280
=================================================================
Total params: 5,864,904
Trainable params: 5,864,904
Non-trainable params: 0
```

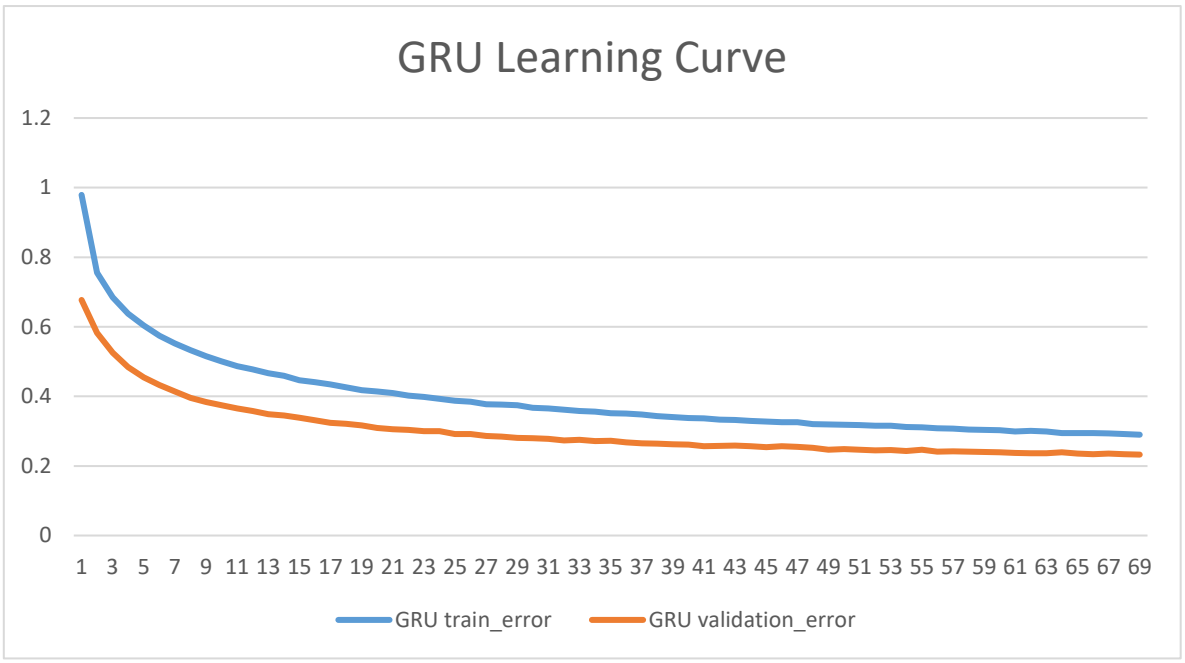
- How to improve your performance (1%)

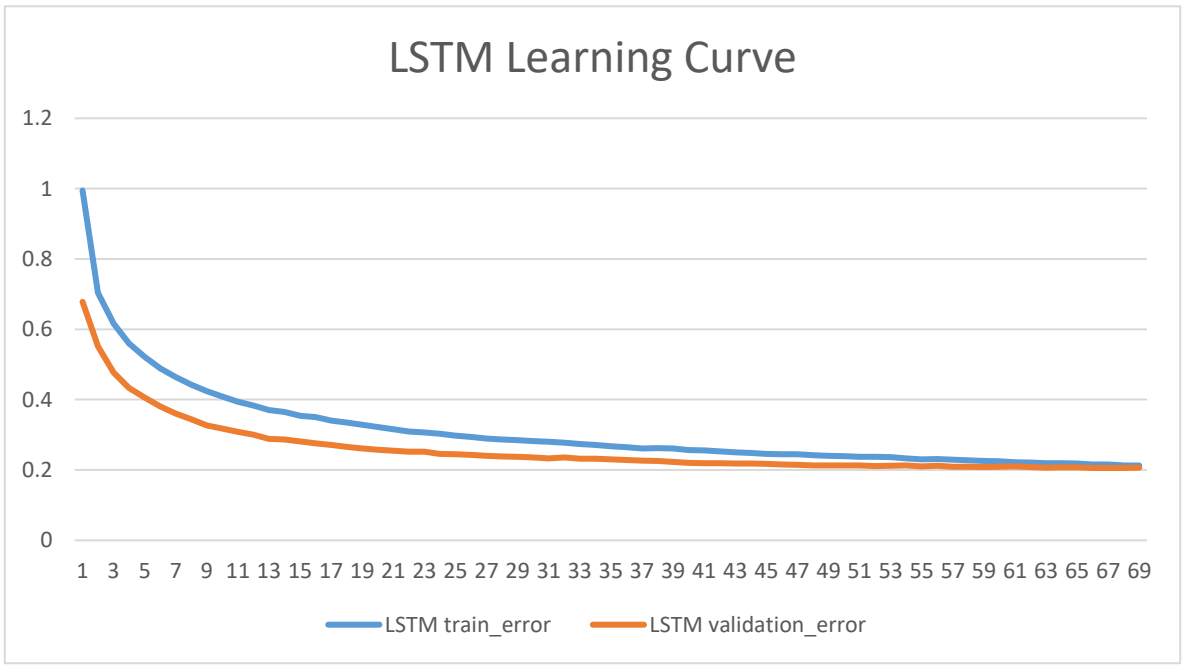
  本次作業我用了一些簡單的方法快速進步 Kaggle 分數:

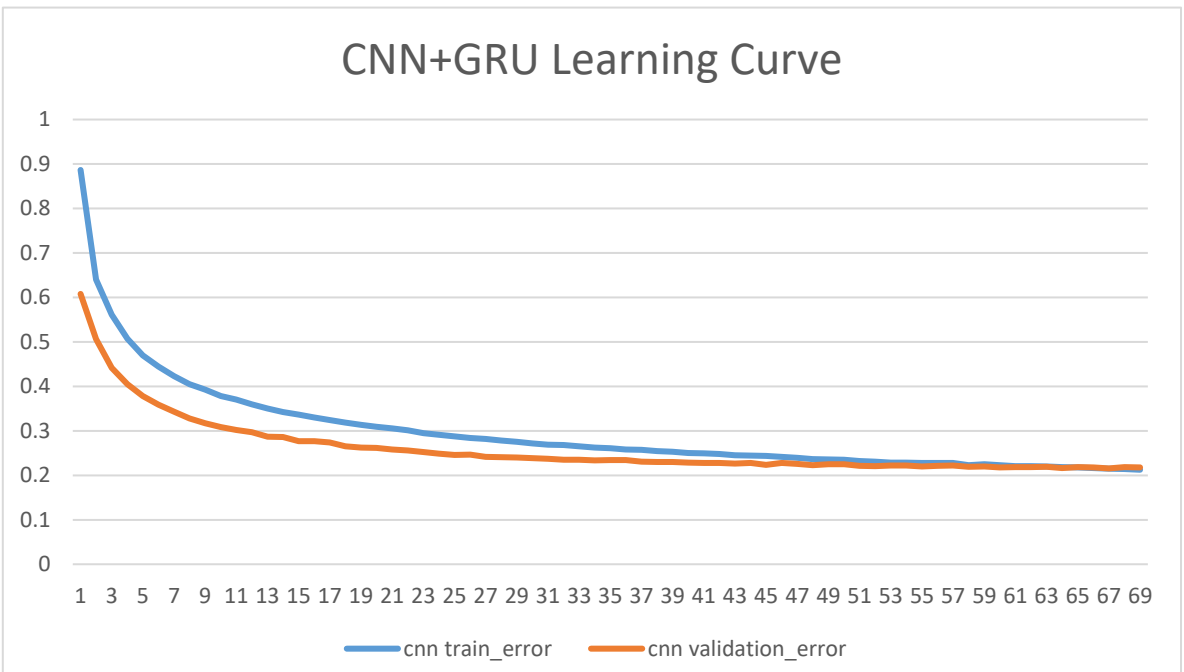
1. 觀察了 training label 之後可以發現很少會出現一單個 phone 的情況如 aabaa，大部分的 phone 都會連在一起，如 aabbaa。因此在最後寫入 csv 時，過濾掉只有單個音出現的情況，過濾後我從 8.67 分進步至 8.2 分。
2. RNN 要做雙向處理(Bidirectional)，也就是除了從左看到右之外，還要倒回來看。我去修電機系另一堂數位語音處理概論時教授有提到，某一個時間點的音其實會受到前後的聲音所影響，因此雙向 RNN 可以做的比單向 RNN 精準許多。
3. 換一張好的 GPU。這個與 model 無關，然而好的 GPU 可以更快速的發現某個 model 到底合不合適，算是最有效提升 performance 的方法，1080 換 1080ti 每個 epoch 大概加速 60%。
4. Ensemble 兩個 model 的結果。我將個分數不錯的結果(7.76, 7.64, 7.56)作 ensemble，進步至 7.06 分。這三個 model 中其中兩個都是四層 GRU，僅最後一層 CELL 數量不同，另一個 model 為同樣的架構改用 LSTM cell，即使都是很類似的模型，做 ensemble 之後還是大幅進步，可見 Ensemble 被列為 Kaggle 必備方法不無道理。

- Experimental results and settings (1%)

最後一題我將嘗試出來三個比較好的模型前 70 個 epoch 的 Learning Curve 作圖比較。LSTM 與 CNN + GRU 模型的 training error 與 validation error 在 0.21 左右的時候產生交叉，之後的訓練就會開始產生 overfit 的情況，而純 GRU 模型直到 error = 0.2 時才產生交叉，overfit 的情況發生的比較晚，最後收斂到的 error 相對其他模型也較低。然而 GRU 所需的收斂速度相對較慢，這與我在網路上看到的幾篇 paper 的結果不太一樣(大部分都是 GRU 收斂情形較好)。而依照直覺，CNN + GRU 模型應該要得到最好的 performance，資料的每一個 timestep 都會將前後各一個 timestep 的 feature 用 CNN 找出新 features 之後再丟進 GRU 做 training，然而可能由於 dropout 或是 kernal_size 沒調整到最好的參數，結果比 GRU 稍差。由於每一個 epoch 都要訓練好幾百秒，故沒有時間再多做一些嘗試。日後如果有更多運算資源可以再 CNN 部分多做研究。


GRU Learning Curve
1.2
1
0.8
0.6
0.4
0.2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69
GRU train_error
GRU validation_error


LSTM Learning Curve
1.2
1
0.8
0.6
0.4
0.2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69
LSTM train_error
LSTM validation_error


CNN+GRU Learning Curve
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69
cnn train_error
cnn validation_error