DSP Project Report

資工三 B05902001 廖彥綸

使用套件：

python 3.6
scipy 1.1.0
pandas 0.23.4
keras 2.1.6
numpy 1.15.4
matplotlib
scikit-learn

內容：

資料處理，在目錄 data 下產生 csv 檔

- data_128_deal.py
- data_deal.py
- vaild_data_128_deal.py
- vaild_data_deal.py

模型訓練，在目錄 model 下產生 h5 檔

- model_128.py
- model.py

預測，在目錄 data 下產生 csv 檔，產生 result.npy

- predict.py

分析

- matrix.py

執行方式(最佳結果)：

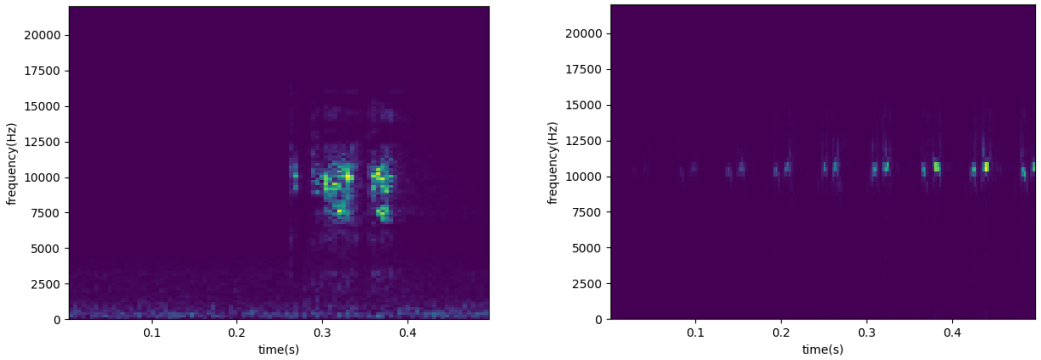
python3 data_128_deal.py	(在目錄 data 下生成 data_128.csv)
python3 vaild_data_128_deal.py	(在目錄 data 下生成 data_128_val.csv)
python3 model_128.py	(在目錄 model 下生成 128_CNN_5.h5)
python3 predict.py	(生成 results.npy)
python3 matrix.py	(val data 的誤差分析)

執行方式(另一組 model)：

python3 data_deal.py　　(在目錄 data 下生成 data.csv)
python3 vaild_data_deal.py　　(在目錄 data 下生成 data_val.csv)
python3 model.py　　(在目錄 model 下生成 256_CNN.5)

1. spectrogram：

對於不同的 window size 進行比較，在 data_deal.py 中使用 FFT 長度 256 的 hanning window (without overlapping)，data_128_deal.py 則使用 FFT 長度 128 的 hanning window (without overlapping)。



左側的圖形是長度 256 的 hanning window，右側是長度 128 的 hanning window。由圖形可看出右邊的圖形各區間較清晰，長度 256 的圖形成帶狀，分布較模糊。

2. CNN model：

第一個模型使用 LeNet。

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 6)         156
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 6)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 16)        2416
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 16)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 400)               0
_________________________________________________________________
dense_1 (Dense)              (None, 120)               48120
_________________________________________________________________
dense_2 (Dense)              (None, 84)                10164
_________________________________________________________________
dense_3 (Dense)              (None, 84)                7140
_________________________________________________________________
dense_4 (Dense)              (None, 20)                1700
_________________________________________________________________
activation_1 (Activation)    (None, 20)                0
=================================================================
Total params: 69,696
Trainable params: 69,696
Non-trainable params: 0
```

```
Epoch 73/75
338/338 [==============================] - 7s 20ms/step - loss: 0.1323 - acc: 0.9617 - val_loss: 0.2257 - val_acc: 0.9359
Epoch 74/75
338/338 [==============================] - 7s 20ms/step - loss: 0.1333 - acc: 0.9629 - val_loss: 0.2363 - val_acc: 0.9324
Epoch 75/75
338/338 [==============================] - 7s 20ms/step - loss: 0.1336 - acc: 0.9602 - val_loss: 0.2406 - val_acc: 0.9312
2324/2324 [==============================] - 0s 146us/step
result 93.115 , loss 0.241
```

第二個模型則在 LeNet 中各層加上 PReLU、BatchNormalization、Dropout(0.25)。

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 6)         156
_________________________________________________________________
p_re_lu_1 (PReLU)            (None, 28, 28, 6)         4704
_________________________________________________________________
batch_normalization_1 (Batch (None, 28, 28, 6)         24
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 6)         0
_________________________________________________________________
dropout_1 (Dropout)          (None, 14, 14, 6)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 16)        2416
_________________________________________________________________
p_re_lu_2 (PReLU)            (None, 10, 10, 16)        1600
_________________________________________________________________
batch_normalization_2 (Batch (None, 10, 10, 16)        64
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 16)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 5, 5, 16)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 400)               0
_________________________________________________________________
dense_1 (Dense)              (None, 120)               48120
_________________________________________________________________
p_re_lu_3 (PReLU)            (None, 120)               120
_________________________________________________________________
dropout_3 (Dropout)          (None, 120)               0
_________________________________________________________________
dense_2 (Dense)              (None, 84)                10164
_________________________________________________________________
p_re_lu_4 (PReLU)            (None, 84)                84
_________________________________________________________________
dropout_4 (Dropout)          (None, 84)                0
_________________________________________________________________
dense_3 (Dense)              (None, 84)                7140
_________________________________________________________________
dense_4 (Dense)              (None, 20)                1700
_________________________________________________________________
activation_1 (Activation)    (None, 20)                0
=================================================================
Total params: 76,292
Trainable params: 76,248
Non-trainable params: 44
```

```
Epoch 73/75
338/338 [==============================] - 13s 40ms/step - loss: 0.2454 - acc: 0.9153 - val_loss: 0.2737 - val_acc: 0.8972
Epoch 74/75
338/338 [==============================] - 13s 40ms/step - loss: 0.2473 - acc: 0.9117 - val_loss: 0.2915 - val_acc: 0.8898
Epoch 75/75
338/338 [==============================] - 13s 40ms/step - loss: 0.2473 - acc: 0.9148 - val_loss: 0.2458 - val_acc: 0.9135
2324/2324 [==============================] - 1s 386us/step
result 91.351 , loss 0.246
```

其他參數的使用包括：optimizer 為 Adam(lr = 0.00005, decay = 1e-6)、weight_decay = 0.000025、未指定 initailization、進行 75 個 epochs、batch_size 為 32 及長度 128 的 hanning window。為了減少變因皆使用相同的模型訓練。

兩者進行 75 個 epochs 的結果為前者更佳一些，可觀察到加上 PReLU、BatchNormalization、Dropout 後花費時間更多，loss 下降的速度更慢，可以更好的避免 overfitting 的情況，但原本的 LeNet 模型已經有不錯的判斷能力，增加其他控制量也不一定能使預測更精確。

3. 訓練結果：

使用模型 LeNet 加上 PReLU

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 6)         156
_________________________________________________________________
p_re_lu_1 (PReLU)            (None, 28, 28, 6)         4704
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 6)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 16)        2416
_________________________________________________________________
p_re_lu_2 (PReLU)            (None, 10, 10, 16)        1600
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 16)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 400)               0
_________________________________________________________________
dense_1 (Dense)              (None, 120)               48120
_________________________________________________________________
dense_2 (Dense)              (None, 84)                10164
_________________________________________________________________
dense_3 (Dense)              (None, 84)                7140
_________________________________________________________________
dense_4 (Dense)              (None, 20)                1700
_________________________________________________________________
activation_1 (Activation)    (None, 20)                0
=================================================================
Total params: 76,000
Trainable params: 76,000
Non-trainable params: 0
```

長度 256 的 hanning window，有大約 80%的準確度

```
338/338 [==============================] - 8s 24ms/step - loss: 0.1742 - acc: 0.9590 - val_loss: 0.9405 - val_acc: 0.8163
Epoch 73/75
338/338 [==============================] - 8s 24ms/step - loss: 0.1683 - acc: 0.9608 - val_loss: 0.9238 - val_acc: 0.8128
Epoch 74/75
338/338 [==============================] - 8s 24ms/step - loss: 0.1630 - acc: 0.9608 - val_loss: 0.9812 - val_acc: 0.8180
Epoch 75/75
338/338 [==============================] - 8s 24ms/step - loss: 0.1640 - acc: 0.9607 - val_loss: 1.1661 - val_acc: 0.7917
2324/2324 [==============================] - 0s 158us/step
result 79.174 , loss 1.166
```

長度 128 的 hanning window，準確度將近 92%

```
338/338 [==============================] - 8s 24ms/step - loss: 0.1306 - acc: 0.9588 - val_loss: 0.2953 - val_acc: 0.9122
Epoch 73/75
338/338 [==============================] - 8s 24ms/step - loss: 0.1305 - acc: 0.9630 - val_loss: 0.2994 - val_acc: 0.9157
Epoch 74/75
338/338 [==============================] - 8s 24ms/step - loss: 0.1233 - acc: 0.9641 - val_loss: 0.2796 - val_acc: 0.9178
Epoch 75/75
338/338 [==============================] - 8s 24ms/step - loss: 0.1234 - acc: 0.9632 - val_loss: 0.2808 - val_acc: 0.9195
2324/2324 [==============================] - 0s 163us/step
result 91.954 , loss 0.281
```

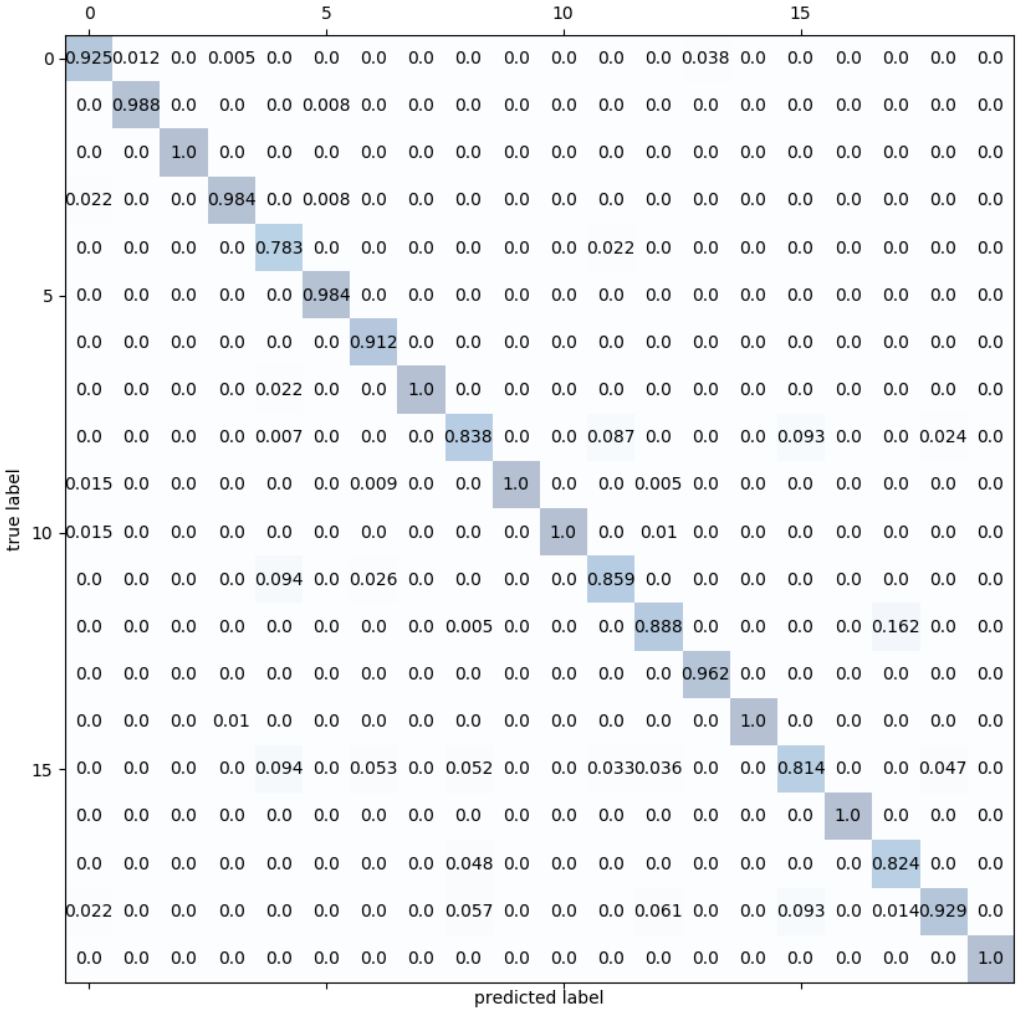
在這次作業長度 128 的 window 更有利於判斷準確性。有嘗試過不同 learning rate 和 epochs 的結果，越大的 learning rate 對 loss 初始的下降顯著，但最後可能因為每一步的距離過大，有正確率反而下降的風險。epochs 數 50 次時兩個 model 的正確率皆有 80%以上，但使用長度 256 時後面的 epochs 沒有任何明顯進步。

4. 混淆類別：

以 confusion matrix 比較哪些類別較容易混淆。

第 4 類 drums_MidTom 和第 11 類 drums_FloorTom 可能擁有相似的聲音容易互相誤判。第 17 類 guitar_chord1 容易判斷為第 8 類 guitar_chord2。第 8 類同時有容易和第 15 類 guitar_3rd_fret 產生誤判等...。

很多相似的聲音判別度並不高，但有些聲音甚至連人耳都難以分辨。



5. 心得：

處理資料室這次作業最費時的事是一開始處理資料部分，需要每一筆取 spectrogram 壓縮至 32*32 後分別處理，我直接將 32*32 的資料壓成 1024 加 label 儲存為 csv 檔，之後的訓練資料職鳩用處理好的 csv 檔。疊 model 並沒有遇到太大的困難，但我還是認為過程類似黑箱的狀態。只能多多嘗試幾種可能性。