學號：B06902052 系級： 資工二 姓名：張集貴

1. (2%) 請說明你實作的 CNN model，其模型架構、訓練參數和準確率為何？並請用與上述 CNN 接近的參數量，實做簡單的 DNN model，同時也說明其模型架構、訓練參數和準確率為何？並說明你觀察到了什麼？
(Collaborators:)

答：

為方便實驗，我將 train.csv 中的資料 shuffle 後切 1/3 為 validation set(x_test, y_test)，其餘 2/3 為 training set(x_train,y_train)。以下準確率為模型在 validation set 上的準確率。

CNN 模型架構：

```
model = Sequential()
model.add(Conv2D(128, (3,3), input_shape = (48,48,1), activation = 'elu'))
model.add(BatchNormalization())
model.add(Dropout(0.2))
model.add(Conv2D(128, (3,3), activation = 'elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.2))

model.add(Conv2D(256, (3,3), activation = 'elu'))
model.add(BatchNormalization())
model.add(Dropout(0.3))
model.add(Conv2D(256, (3,3), activation = 'elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.3))

model.add(Conv2D(512, (3,3), activation = 'elu'))
model.add(BatchNormalization())
model.add(Dropout(0.4))
model.add(Conv2D(512, (3,3), activation = 'elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.4))

model.add(Flatten())
model.add(Dense(units = 512, activation ='elu'))
model.add(BatchNormalization())
model.add(Dense(units = 512, activation ='elu'))
model.add(BatchNormalization())
model.add(Dense(units = 7, activation ='softmax'))
```

CNN 訓練參數：

```
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
model.fit(x_train,y_train,validation_data=(x_test,y_test),batch_size=100,epochs=100)
```

CNN 總參數量：5,900,679

CNN 準確率：0.6077960068584797

DNN 模型架構：

```
model = Sequential()
model.add(Flatten(input_shape=(48,48,1)))
model.add(Dense(units = 512, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.2))
model.add(Dense(units = 512, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.2))

model.add(Dense(units = 512, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.3))
model.add(Dense(units = 666, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(units = 1024, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.4))
model.add(Dense(units = 1024, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.4))

model.add(Dense(units = 1024, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.4))
model.add(Dense(units = 1024, activation ='elu'))
model.add(BatchNormalization())
model.add(Dropout(0.4))

model.add(Dense(units = 7, activation ='softmax'))
```

DNN 訓練參數：

```
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
model.fit(x_train,y_train,validation_data=(x_test,y_test),batch_size=100,epochs=100)
```

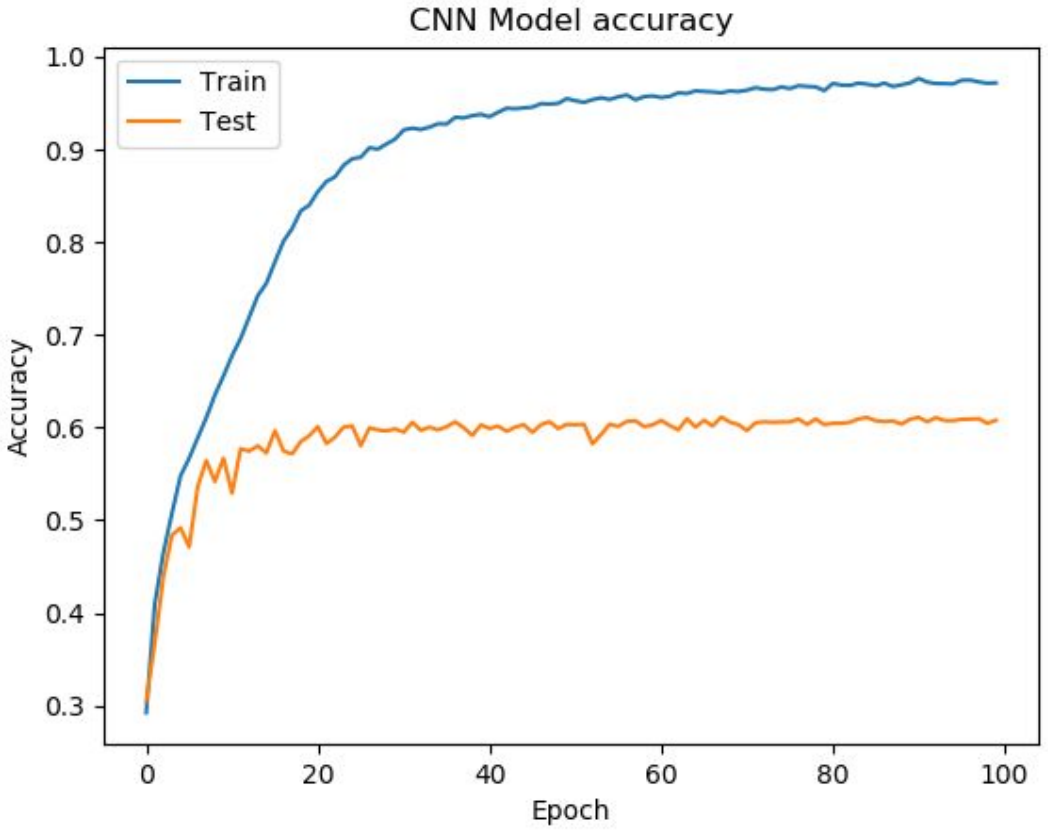
DNN 總參數量：5,911,305

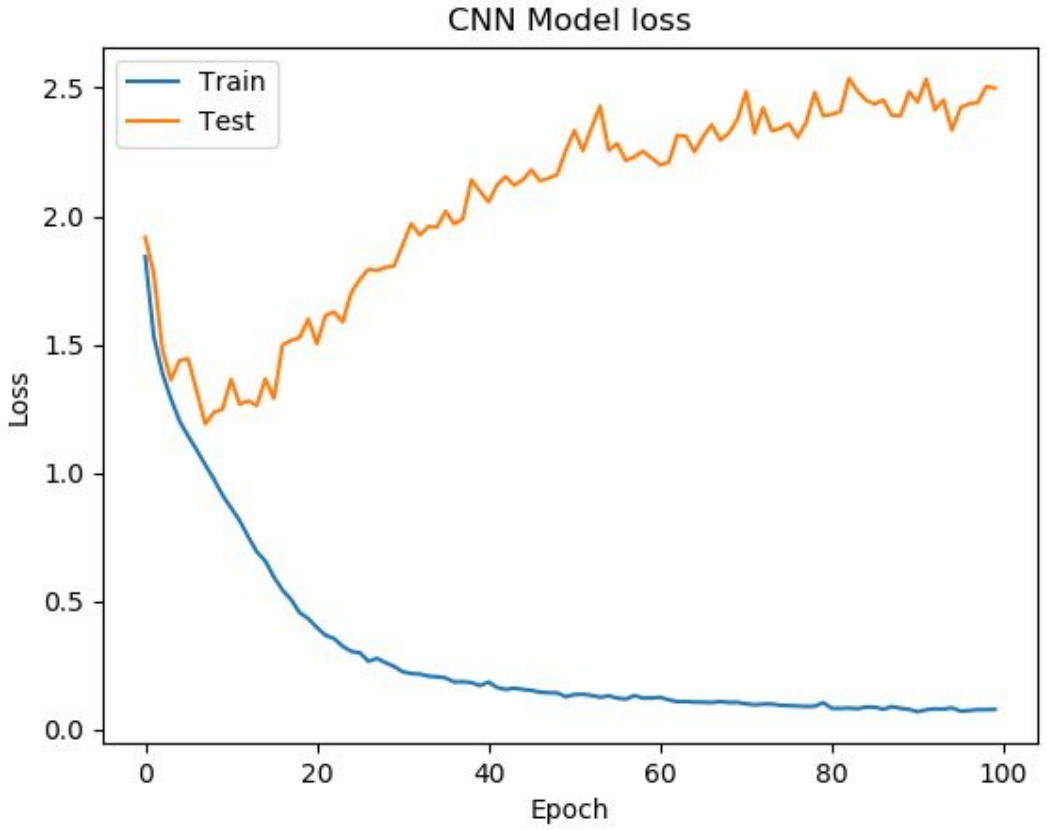
DNN 準確率：0.4082976256851767

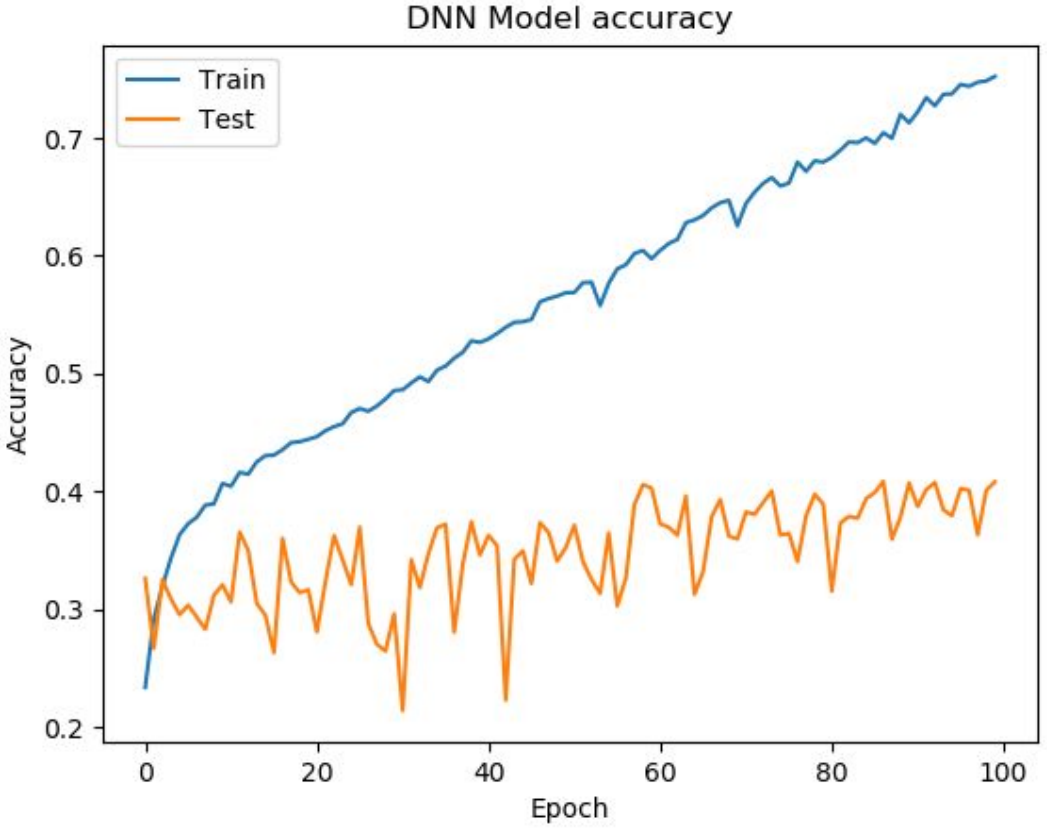
總參數量差不多的 CNN 跟 DNN，用一樣的訓練參數訓練之後，CNN 的表現明顯比 DNN 好很多。

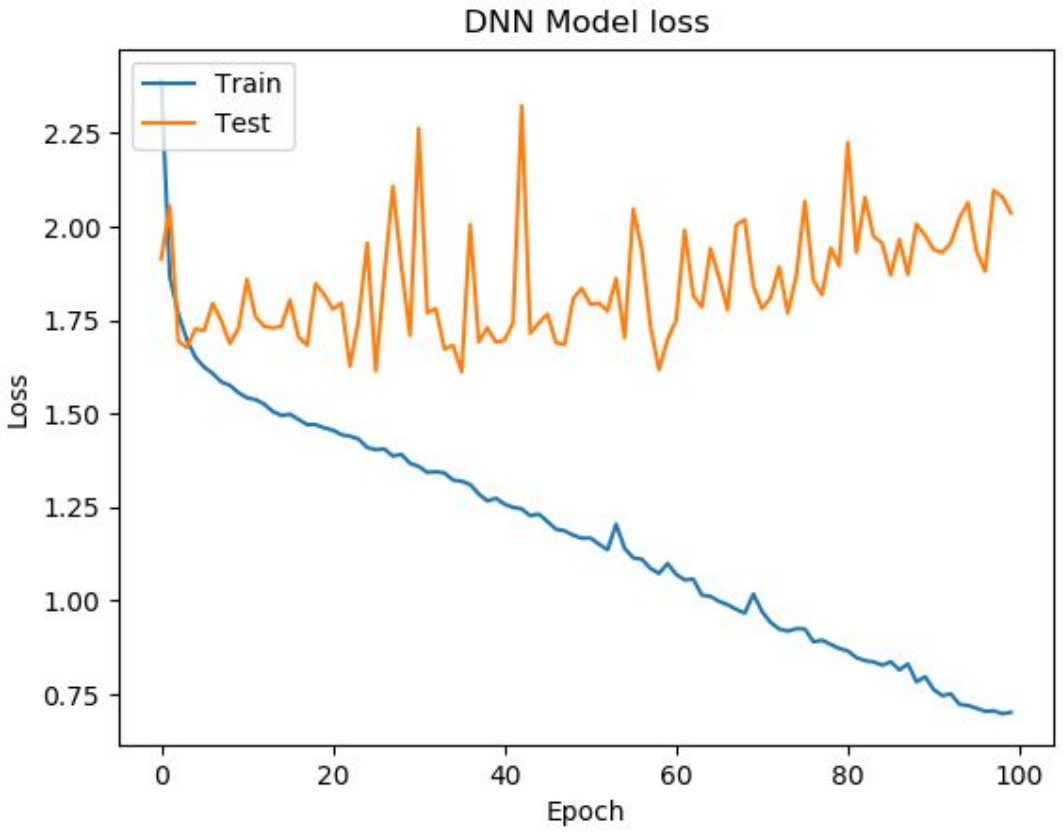
2. (1%) 承上題，請分別畫出這兩個model的訓練過程 (i.e., loss/accuracy v.s. epoch) (Collaborators:)

   答：




CNN Model loss
Train
Test
Loss
2.5
2.0
1.5
1.0
0.5
0.0
0
20
40
60
80
100
Epoch


DNN Model accuracy
Train
Test
Accuracy
0.7
0.6
0.5
0.4
0.3
0.2
0
20
40
60
80
100
Epoch



3. (1%) 請嘗試 data normalization, data augmentation,說明實作方法並且說明實行前後對準確率有什麼樣的影響？
   (Collaborators:)

   答：對以下三種 case 進行實驗

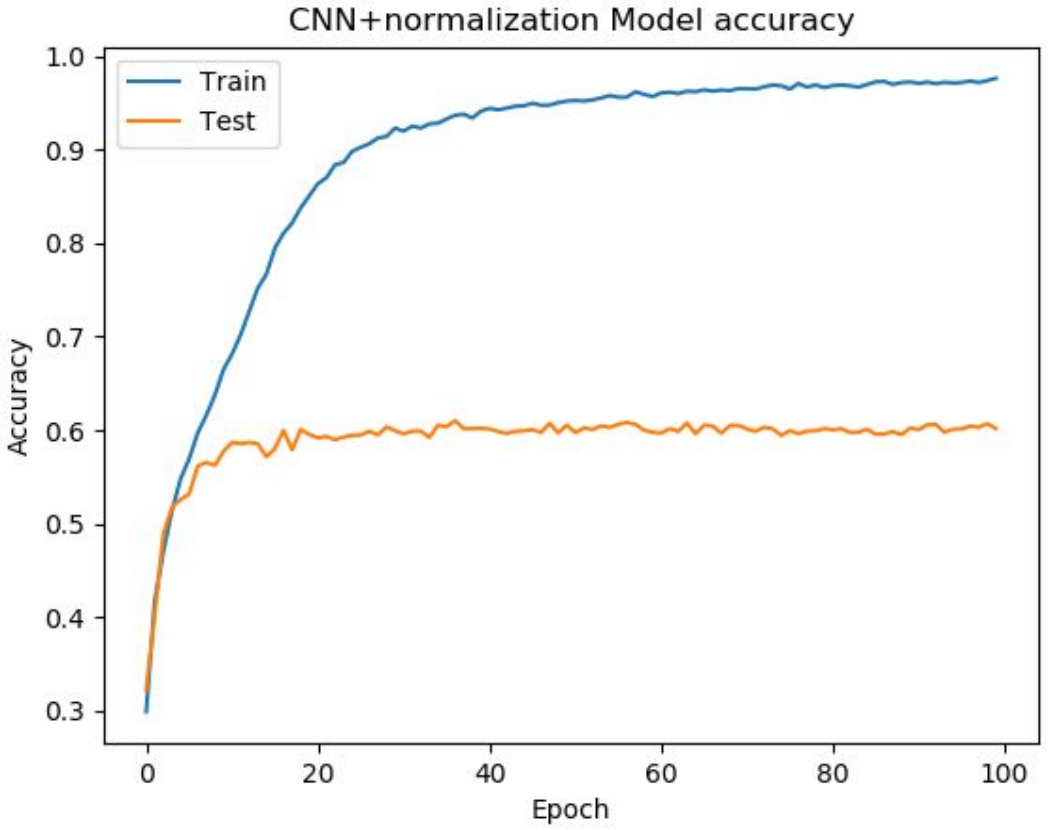
(1) raw：即為 1. 的 CNN model

準確率：0.6077960068584797

(2) normalization：對 48*48*1 個輸入資料的維度分別做 standardization。

準確率：0.6017347658200004

對最終準確率影響不大，但可看出訓練過程中的準確率曲線較為平滑。



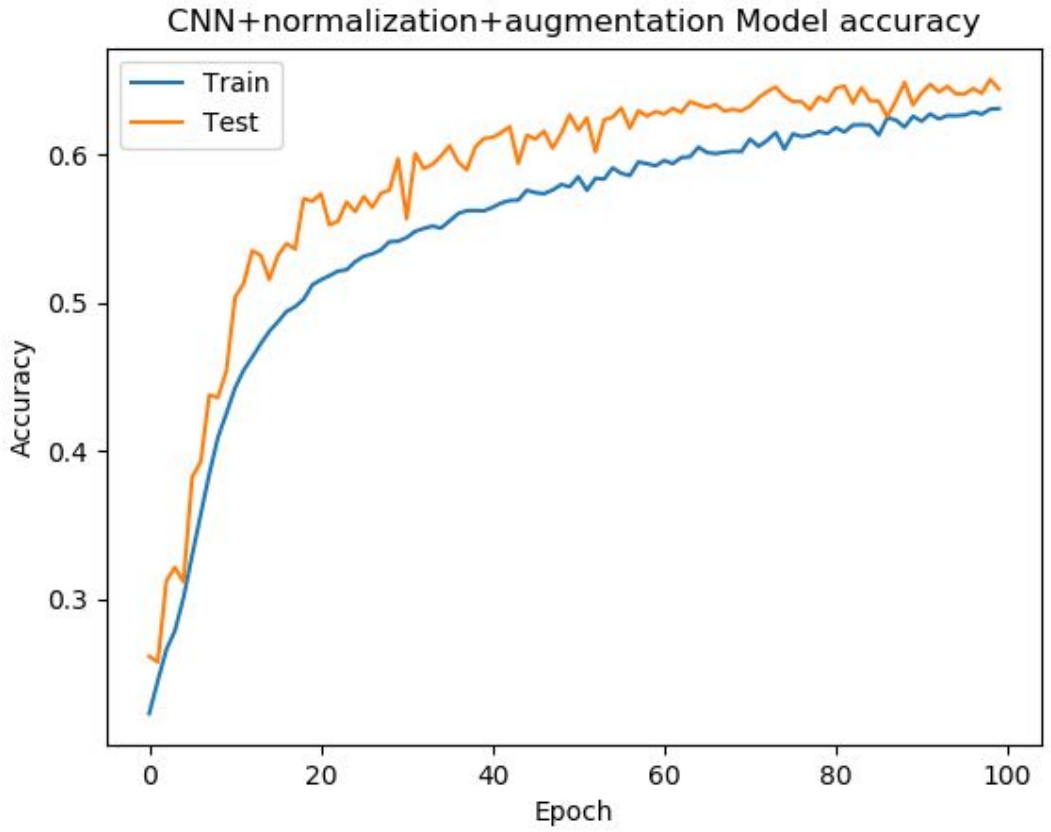
(3) normalization + augmentation：做完(2)後，使用 Keras 的 ImageDataGenerator 來產生資料，並用 fit_generator 進行訓練。

準確率：0.644163444456056

```
datagen = ImageDataGenerator(
        rotation_range=40,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest')
model.fit_generator(
        datagen.flow(x_train,y_train,batch_size=100),
        validation_data=(x_test,y_test),
        steps_per_epoch=len(x_train)//100+1,
        epochs=100,
        workers=10)
```

準確率提高了，從訓練過程中可看出 training accuracy 明顯降低了，可見augmentation

改善了 overfitting 的問題。

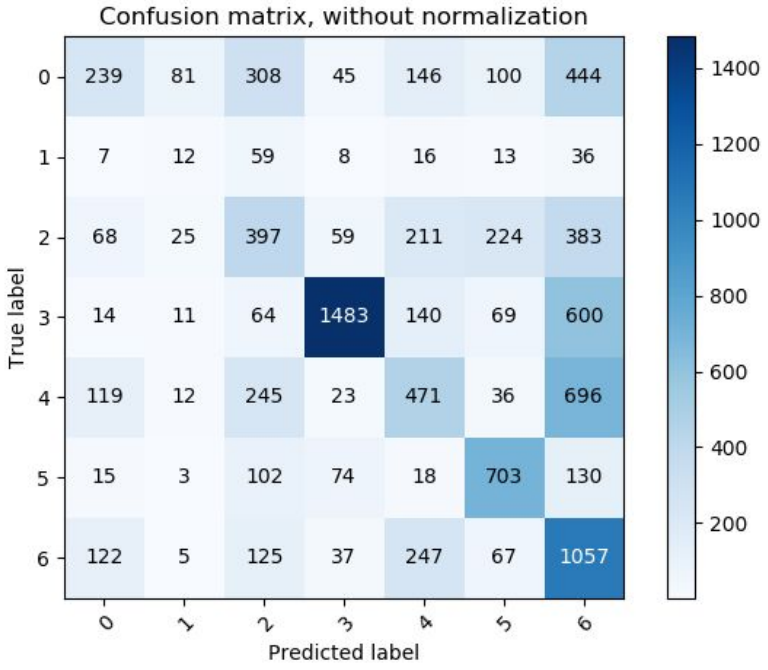


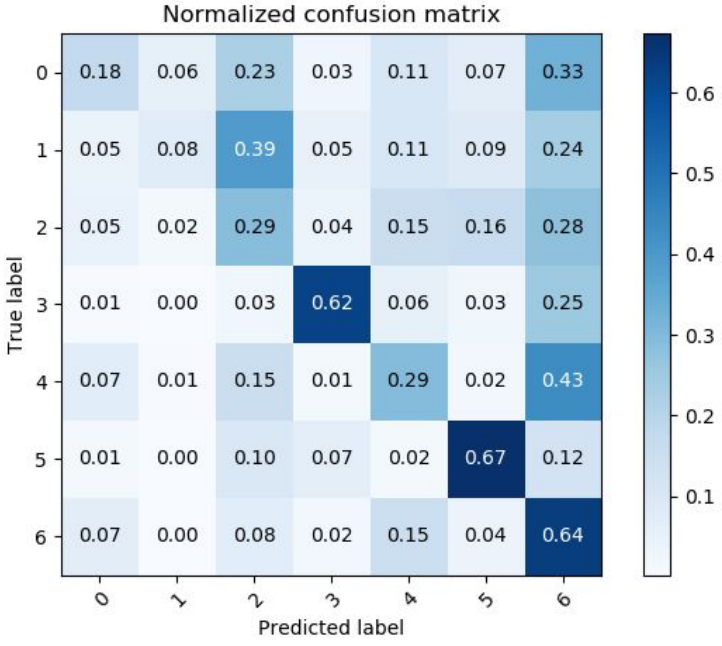
4. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]
(Collaborators:)

答：

(a) 0 跟 1 的準確率相當低，僅有 0.18 跟 0.08，大多數的 0, 1 都被 predict 成 2 或 6。

(b) 0 跟 4 最多的比例被判成 6。1,2,3,5 第二多的比例是 6。可見幾乎所有表情都容易被誤判成 6，然後 6 (1660 張)並不是最多的資料，最多的是 3 (2381張)，可能是因為 6 是中立的表情，如果表情做的不夠誇張會被誤判當成中立。


Confusion matrix, without normalization
True label
Predicted label
0 239 81 308 45 146 100 444
1 7 12 59 8 16 13 36
2 68 25 397 59 211 224 383
3 14 11 64 1483 140 69 600
4 119 12 245 23 471 36 696
5 15 3 102 74 18 703 130
6 122 5 125 37 247 67 1057
0 1 2 3 4 5 6
1400 1200 1000 800 600 400 200


Normalized confusion matrix
True label
Predicted label
0 0.18 0.06 0.23 0.03 0.11 0.07 0.33
1 0.05 0.08 0.39 0.05 0.11 0.09 0.24
2 0.05 0.02 0.29 0.04 0.15 0.16 0.28
3 0.01 0.00 0.03 0.62 0.06 0.03 0.25
4 0.07 0.01 0.15 0.01 0.29 0.02 0.43
5 0.01 0.00 0.10 0.07 0.02 0.67 0.12
6 0.07 0.00 0.08 0.02 0.15 0.04 0.64
0 1 2 3 4 5 6
0.6 0.5 0.4 0.3 0.2 0.1