ML Hw3 report

1. **請說明你實作的 CNN 模型(best model)，其模型架構、訓練參數量和準確率為何？(1%)**

   我的CNN 架構是由7層的convolution layer 最後再接上4096 X 1024 X 512 X 11的fully connected feedforward layer。每層convolution layer 完都會讓他再經過batch normalization 跟 ReLU function，唯一不同的是第一層是由原本的3個channel 經由 convolution 拉到32個 channel ，而這一層我是沒有做maxpooling 的，使用的activation function 為PReLU。每層 fully connected feedforward layer 之後都會經過ReLU 和 dropout，dropout 比率為0.5。訓練的參數量約為2517,7868，epoch數量定為60，而在validation set的準確率約為0.752。

2. **請實作與第一題接近的參數量，但 CNN 深度（CNN 層數）減半的模型，並說明其模型架構、訓練參數量和準確率為何？(1%)**

   此減半層數的模型架構與原本的CNN模型相似，僅拿掉了前四層的convolution layer，並把原本第五層的nn.Conv2d(216, 512, 3, 1, 1) 改為 nn.Conv2d(3, 512, 3, 1, 1)，訓練參數量與第一題相近，為2362,2667，而在validation set 的準確率約為0.635。

3. **請實作與第一題接近的參數量，簡單的 DNN 模型，同時也說明其模型架構、訓練參數和準確率為何？(1%)**

   此DNN模型相較於第一題原本的CNN模型，將convolution layer 全部拿掉，並且DNN的架構為：

   ```
   self.fc = nn.Sequential(
               nn.Linear(3*128*128, 512),
               nn.Dropout(0.5),
               nn.ReLU(),
               nn.Linear(512, 128),
               nn.Dropout(0.5),
               nn.ReLU(),
               nn.Linear(128, 11)
           )
   ```

   訓練的參數量與第一題相去不遠，參數量為2523,3419，而在validation set的準確率大約為0.294。

4. **請說明由 1 ~ 3 題的實驗中你觀察到了什麼？(1%)**

   從以上三題的實驗中，可以很清楚的看見當CNN的層數越多，對於這樣一個分辨圖片的問題在準確度的表現上會有相當明顯的差異。在第二題的模型中，光是減去一半的convolution layer就足以在performence上與原本的相差0.1以上，而當第三題拿掉了所有的CNN層數，僅依靠DNN來訓練模型的時候，我們可以發現甚至可以說是train不起來，僅比用"猜"的表現還好上一些。這也說明了雖然上面三題的訓練參數量是一樣的，但是顯然在這樣的圖片分類問題中，使用完整的CNN的模型會好上非常多。

5. **請嘗試 data normalization 及 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？(1%)**

   實作方法:

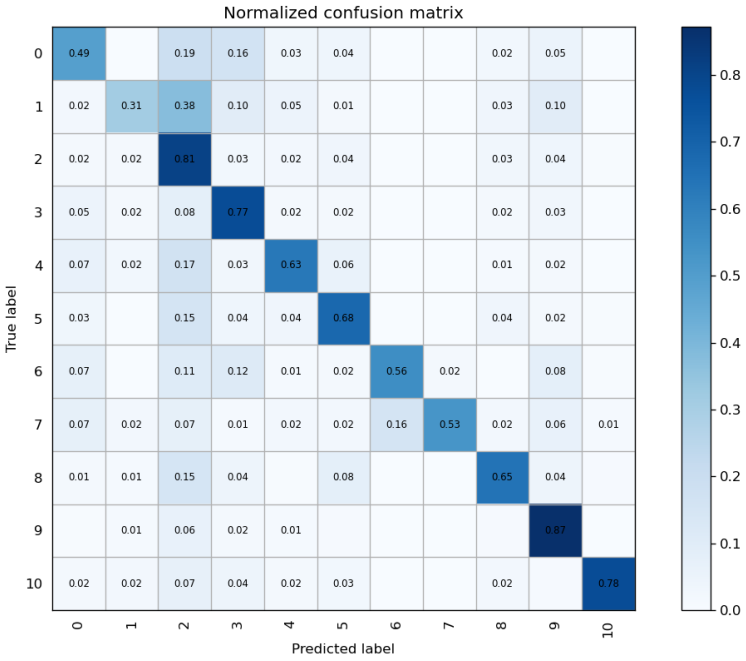
- Data normalization : 在torchvision.transforms可以實作，利用transforms.normalization 將圖片的數據轉為0~1之間的數據。
- Data augmentation : 利用 torch的torchvision.transforms可以實作，可以利用 transforms.rotate, transforms.CenterCrop 等等來增加 data 的數量。

以下實驗皆為**batch_size=64, learning rate= 0.001 , epoch = 25 , CNN layer = 5 :**

	Do nothing	data normalization	data augmentation	Both
Validation Set Acc	0.522	0.619	0.631	0.694

從上表可知，無論是 data normalization ，還是做data augmentation，都對準確率有很大的影響。而且兩個一起做的時候，準確率相較於原本的模型有顯著的提升。

6. **觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]**



此為使用助教一開始給的code，再將epoch調至60，learning rate調整到0.0001後所繪製的confusion matrix。從圖中我們可以發現第一類的跟第二類的似乎訓練的成效沒有很好，容易判斷成其他的類別，而尤其是第一類的圖片中，判斷出第二類的比率竟然比正確的比率更高。由圖中可以大致發現容易搞混的class有：**0 & 2; 0 & 3 ;1 & 2 ; 4 & 2 ; 5 & 2 ; 6 & 2 ; 6 & 3 ; 7 & 6 ; 8 & 2**。可以發現這些class 之間認錯的比率都 > 0.1 ，尤其是幾乎每一個class 都會跟第二個class 搞混。