# Lab3 Report

## 1. Introduction

在這次的lab使用了自訂的dataloader，並使用ResNet50、ResNet18預訓練與非預訓練4個model做分類並畫出cunfusion matrix.

## 2. Experiment set up

### A. The detail of model

ResNet使用skip connection的方法，減少model過深帶來的gradient vanishing和gradient explosion問題。

使用torchvision裡的models去使用ResNet18和ResNet50，參數可調整是否Pretrained

```
model_18_pretrain = models.resnet18(pretrained=True)
```

### B. The detail of dataloader

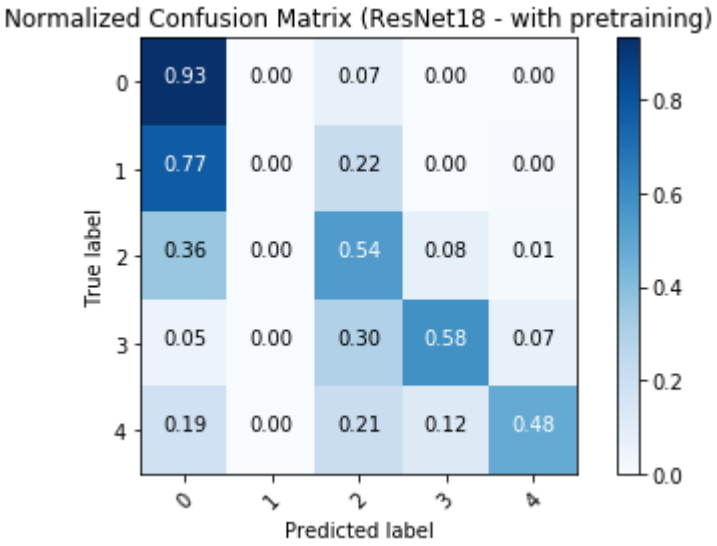
在train的部分，將圖片隨機crop再resize回512，將圖片隨機旋轉並隨機平移，隨機水平翻轉，然後再做normalization

在test的部分，resize成512x512，然後做normalization

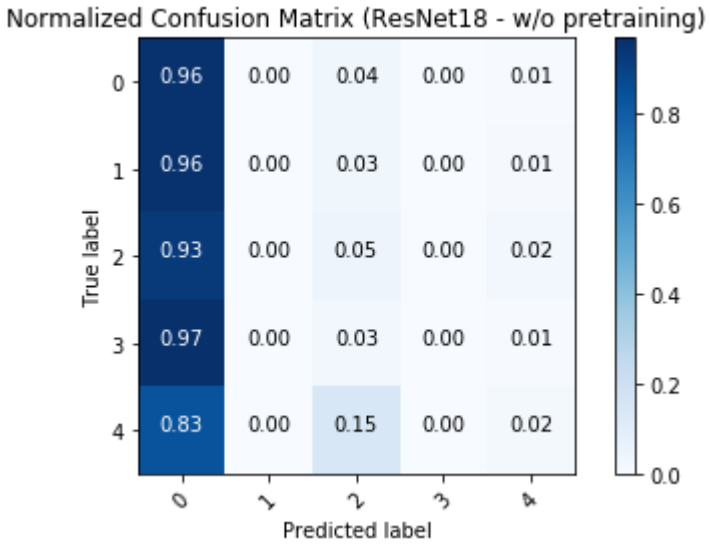
使用PIL讀image，做完data augmentation再轉成trensor返回

```python
class RetinopathyLoader(data.Dataset):
    def __init__(self, root, mode):
        """
        Args:
            root (string): Root path of the dataset.
            mode : Indicate procedure status(training or testing)

            self.img_name (string list): String list that store all image names.
            self.label (int or float list): Numerical list that store all ground truth label values.
        """
        self.root = root
        self.img_name, self.label = getData(mode)
        self.mode = mode
#         print("> Found %d images..." % (len(self.img_name)))
        MEAN = [108.64628601 / 255, 75.86886597 / 255, 54.34005737 / 255]
        STD = [70.53946096 / 255, 51.71475228 / 255, 43.03428563 / 255]
        train_transform = transforms.Compose([
            transforms.RandomResizedCrop(
                size=512,
                scale=(1 / 1.15, 1.15),
                ratio=(0.7561, 1.3225)
            ),
            transforms.RandomAffine(
                degrees=(-180, 180),
                translate=(40 / 448, 40 / 448),
                scale=None,
                shear=None
            ),
            transforms.RandomHorizontalFlip(),
            transforms.RandomVerticalFlip(),
            transforms.ToTensor(),
            transforms.Normalize(tuple(MEAN), tuple(STD)),
        ])
        test_transform = transforms.Compose([
            transforms.Resize((512, 512)),
            transforms.ToTensor(),
            transforms.Normalize(tuple(MEAN), tuple(STD))
        ])
        self.transfrom = None
        if mode == 'train':
            self.transform = train_transform
        else:
            self.transform = test_transform

    def __len__(self):
        """'return the size of dataset"""
        return len(self.img_name)


  def __getitem__(self, index):
      """something you should implement here"""

      """
         step1. Get the image path from 'self.img_name' and load it.
                hint : path = root + self.img_name[index] + '.jpeg'

         step2. Get the ground truth label from self.label

         step3. Transform the .jpeg rgb images during the training phase, such as resizing, random flipping,
                rotation, cropping, normalization etc. But at the beginning, I suggest you follow the hints.

                In the testing phase, if you have a normalization process during the training phase, you only need
                to normalize the data.

                hints : Convert the pixel value to [0, 1]
                        Transpose the image shape from [H, W, C] to [C, H, W]

          step4. Return processed image and label
      """

      path = os.path.join('.', self.root, self.img_name[index] + '.jpeg')
      img = Image.open(path)
      img = self.transform(img)
      label = self.label[index]

      return img, label
```

C. Describing your evaluation through the confusion matrix




在有pretrained時，predict的label較為分散




在沒有pretrained時，大部分predict的label是0

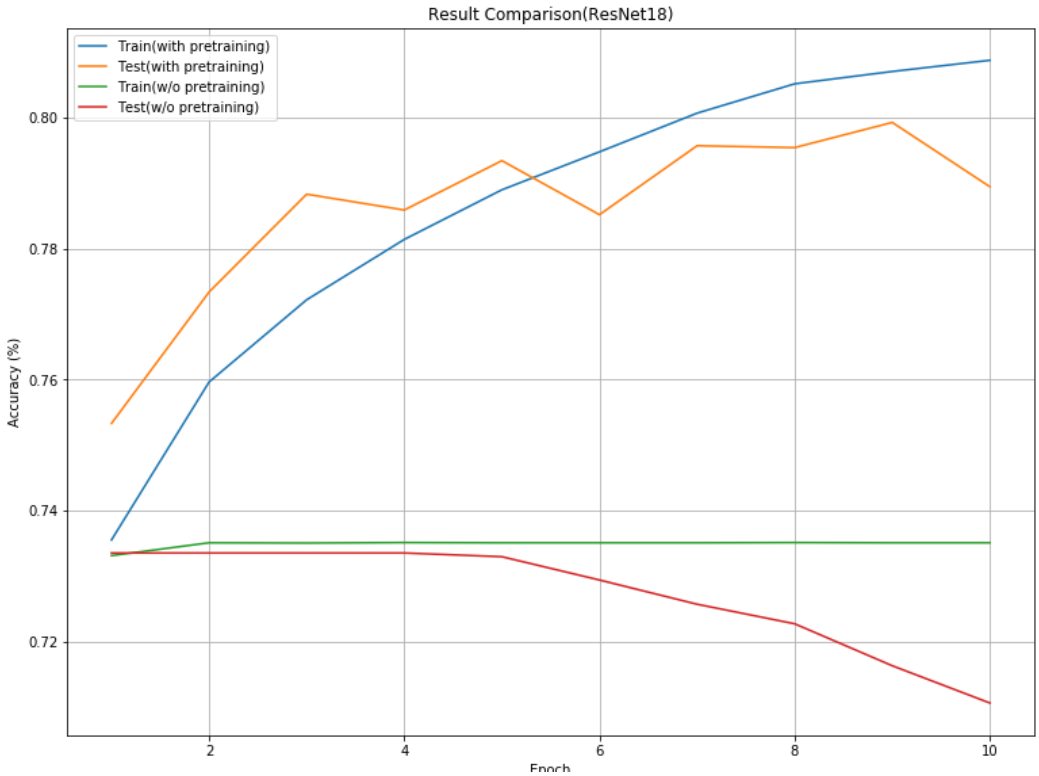
# 3. Experiment Results

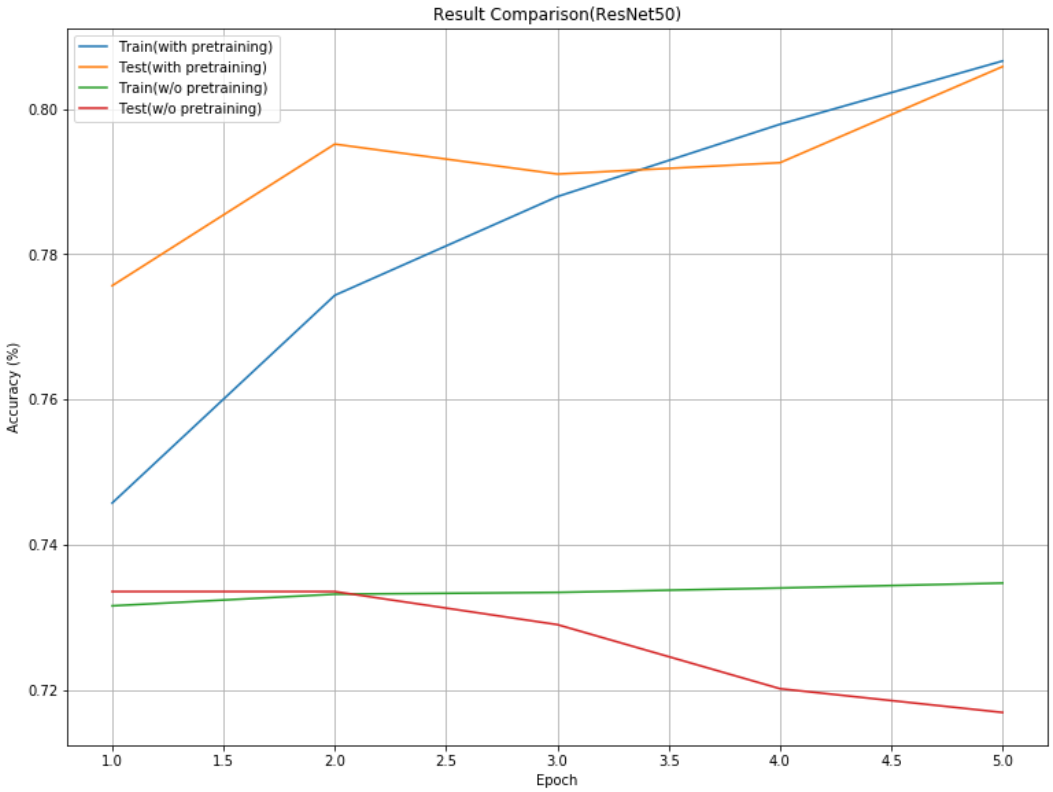
## A. The highest testing accuracy

```
                ResNet18 with pretraining

Test set: Average loss: 0.1580, Accuracy: 5546/7025 (79%)

================================================================
                ResNet18 w/o pretraining

Test set: Average loss: 0.2337, Accuracy: 4992/7025 (71%)

================================================================
                ResNet50 with pretraining

Test set: Average loss: 0.1535, Accuracy: 5661/7025 (81%)

================================================================
                ResNet50 w/o pretraining

Test set: Average loss: 0.2332, Accuracy: 5036/7025 (72%)

================================================================
```

Without pretrained，在train時每個epoch的loss跟accuracy都不太會動，效果相當不好，理由應該是初始給定的weight很重要

# B. Comparison figures



## 4. Discussion

這次的作業開始要train比較久了，而且要修改model就要再重train幾次，需要更早開始。從比較圖跟數據都可以看出是否pretrain對結果的影響很大。