學號：b06902006　　系級：資工三　　姓名：王俊翔

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？

   模型架構如下圖:

```
class Classifier(nn.Module):
    def __init__(self):
        super(Classifier, self).__init__()
        #torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride, padding)
        #torch.nn.MaxPool2d(kernel_size, stride, padding)
        #input 維度 [3, 128, 128]
        self.cnn = nn.Sequential(
            nn.Conv2d(3, 64, 3, 1, 1),  # [64, 128, 128]
            nn.BatchNorm2d(64),
            nn.PReLU(),
            nn.Conv2d(64, 64, 3, 1, 1),  # [64, 128, 128]
            nn.BatchNorm2d(64),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),      # [64, 64, 64]

            nn.Conv2d(64, 128, 3, 1, 1), # [128, 64, 64]
            nn.BatchNorm2d(128),
            nn.PReLU(),
            nn.Conv2d(128, 128, 3, 1, 1), # [128, 64, 64]
            nn.BatchNorm2d(128),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),      # [128, 32, 32]

            nn.Conv2d(128, 256, 3, 1, 1), # [256, 32, 32]
            nn.BatchNorm2d(256),
            nn.PReLU(),
            nn.Conv2d(256, 256, 3, 1, 1), # [256, 32, 32]
            nn.BatchNorm2d(256),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),      # [256, 16, 16]

            nn.Conv2d(256, 512, 3, 1, 1), # [512, 16, 16]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.Conv2d(512, 512, 3, 1, 1), # [512, 16, 16]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),       # [512, 8, 8]

            nn.Conv2d(512, 512, 3, 1, 1), # [512, 8, 8]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.Conv2d(512, 512, 3, 1, 1), # [512, 8, 8]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),       # [512, 4, 4]
        )
        self.fc = nn.Sequential(
            nn.Linear(512*4*4, 2048),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(2048, 2048),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(2048, 512),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(512, 11)
        )

    def forward(self, x):
        out = self.cnn(x)
        out = out.view(out.size()[0], -1)
        return self.fc(out)
```

總訓練參數量為 31441240，在 validation set 的準確率為 0.762682。

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？

   模型架構如下圖:

```python
class ClassifierA(nn.Module):
    def __init__(self):
        super(ClassifierA, self).__init__()
        #torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride, padding)
        #torch.nn.MaxPool2d(kernel_size, stride, padding)
        #input 維度 [3, 128, 128]
        self.cnn = nn.Sequential(
            nn.Conv2d(3, 256, 3, 1, 1),  # [64, 128, 128]
            nn.BatchNorm2d(256),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),      # [64, 64, 64]

            nn.Conv2d(256, 512, 3, 1, 1), # [128, 64, 64]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),      # [128, 32, 32]

            nn.Conv2d(512, 512, 3, 1, 1), # [256, 32, 32]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),      # [256, 16, 16]

            nn.Conv2d(512, 512, 3, 1, 1), # [512, 16, 16]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),       # [512, 8, 8]

            nn.Conv2d(512, 512, 3, 1, 1), # [512, 8, 8]
            nn.BatchNorm2d(512),
            nn.PReLU(),
            nn.MaxPool2d(2, 2, 0),       # [512, 4, 4]
        )
        self.fc = nn.Sequential(
            nn.Linear(512*4*4, 2048),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(2048, 2048),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(2048, 512),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(512, 11)
        )

    def forward(self, x):
        out = self.cnn(x)
        out = out.view(out.size()[0], -1)
        return self.fc(out)
```

   總訓練參數量為 30301715，在 validation set 的準確率為 0.760641。

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？

   模型架構如下圖:

```python
class linear(nn.Module):
    def __init__(self):
        super(linear, self).__init__()

        self.fc = nn.Sequential(
            nn.Linear(3*128*128, 640),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(640, 512),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(512, 512),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(512, 512),
            nn.Dropout(0.5),
            nn.PReLU(),
            nn.Linear(512, 11)
        )

    def forward(self, x):
        out = x.view(x.size()[0], -1)
        return self.fc(out)
```

總訓練參數量為 32317071，在 validation set 的準確率為 0.294752。

4. 請說明由 1 ~ 3 題的實驗中你觀察到了什麼？

   以上三題都是在 learning rate 初始為 0.0001，且 epoch 為 150 次，在 30, 55, 75, 95 時會將 lr 變為 0.2 倍，在 1, 2 當中其實發現兩者的準確率雖然前者稍微高一點，但是並沒有相差非常多，且收斂的速度也差不多，不過與純 DNN 做比較時就發現了顯著的差異，純 DNN 很快就到達了收斂，且正確率無法繼續上去，說明了整個 model 架構非常的不合適，且 train 的時間也比前兩者短。

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？

   進行 data normalization 前後的準確率可以說完全不會改變，在 validation set 上都約為 76.5%。

   實作方法:有做的就如助教 code 所寫，ToTensor()就會自動幫你 normalize，沒做的就是取消那一行，並在定義 dataset 要__getitem__ 時在使用 transform.functional.to_tensor。

   Data augmentation 做了許多嘗試:

實作方法都為在助教 code 中的 transform.Compose 添加想要的 augmentation。

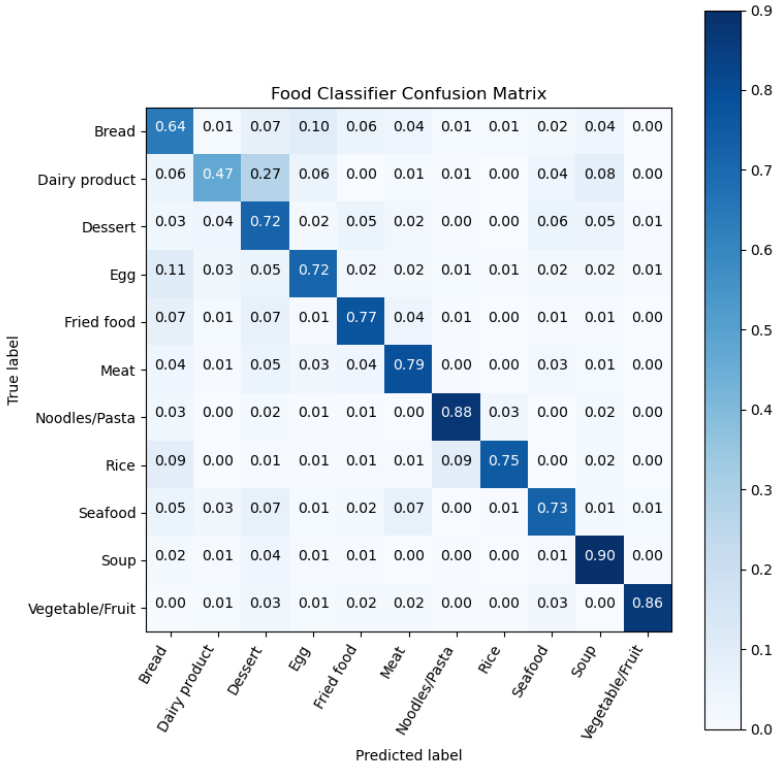
增加 RandomHorizontalFlip 使準確率小幅增加，所以就繼續使用。

把 RandomRatation 中的角度變化，發現在 15-20 度時效果最好，轉太多甚至會造成反效果，沒轉也會減少變化性。

加入 ColorJitter 改變他的亮度飽和度等等，不過怎麼變都沒有原圖好。

一開始讀入變成各種方型大小(>128*128)，然後在做 RandomCrop，但是效果仍不如原本的。

6. 觀察答錯的圖片中，哪些 class 彼此間容易用混？



由上圖可以發現，最容易混淆的就是 Dairy product 及 Dessert，Bread 的部分也不是很理想，尤其會與 Egg 混淆。