EEP 596 Computer Vision

HW #6

The purpose of this homework is to continue exploring computer vision concepts using PyTorch. You are free to work with fellow students but should turn in your own work. Specifically, you should be able to explain anything in your assignment if asked.

You should turn in your assignment as **"Assignment6.py" and "Report.pdf,"** in Gradescope. **Please also turn in your "Report.pdf," in Canvas.** Use the CIFAR10 dataset.

The questions are as follows.

1. **(15 points) Number of parameters.** Write a function to compute the number of trainable parameters in a model, the function takes as input the model, e.g., ResNet-34. Return the estimated number of parameters. *The question will be automatically graded.*

2. **Global average pooling (30 points). Please briefly revisit HW4 before starting. Reuse the CIFAR10_dataset_a() implementation from the previous assignment for the dataset.** Create a network with the following parameters:

```
GAPNet(
  (conv1): Conv2d(3, 6, kernel_size=(5, 5), stride=(1, 1))
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
  (conv2): Conv2d(6, 10, kernel_size=(5, 5), stride=(1, 1))
  (gap): AvgPool2d(kernel_size=10, stride=10, padding=0)
  (fc): Linear(in_features=10, out_features=10, bias=True)
)
```

   Train the network for 10 epochs with learning rate =0.001 and momentum=0.9. Use Stochastic Gradient Descent as the optimizer. Evaluate the model and save the weights as Gap_net_10epoch.pth and submit it on Gradescope.
3. **Backbones (10 points).** Download resnet18 (pretrained on ImageNet). Remove the final fully connected layer (classifier). Return the features for "cat_eye.jpg".
4. **Transfer learning (30 points).** Download resnet18 (pretrained on ImageNet), modify the last layer to output 10 classes. Freeze all the weights except the last layer. (`for param in model.parameters(): param.requires_grad = False` and so forth. See here.) Train the network for 10 epochs with learning rate =0.001 and momentum=0.9. Use Stochastic Gradient Descent as the optimizer. Evaluate the model and save the weights as Res_net_10epoch.pth and submit it on Gradescope.

5. **(25 points) DenseNet.** Write PyTorch code to implement **MobileNet.** Derive your class from nn.Module, and implement both __init__() and forward() methods. Be sure to include batch norm and ReLU. Run your network on an image to **be sure that all the dimensions are correct**; you do *not* have to check that the output makes sense. (It will not, since the network is not trained.) *The question will be automatically graded.*