ASSIGNMENT 3

CS5304 - TRANSFER LEARNING IN CONVOLUTIONAL NEURAL NETWORKS

1. Transfer Learning

From http://cs231n.github.io/transfer-learning/:
In practice, very few people train an entire Convolutional Network from scratch (with random initialization), because it is relatively rare to have a dataset of sufficient size. Instead, it is common to pretrain a ConvNet on a very large dataset (e.g. ImageNet, which contains 1.2 million images with 1000 categories), and then use the ConvNet either as an initialization or a fixed feature extractor for the task of interest.

We'll explore two different types of Transfer Learning in this assignment. The first approach is using a pre-trained CNN as a fixed feature extractor. In this technique, all layers of the CNN is frozen except for the last fully-connected layer. This last layer is changed to suit the task at hand. In this part of the assignment, you will take a pre-trained model in PyTorch and replace the last fully connected layer which classifies images into 1000 classes into a new classifier that is adapted to classify images into 5 classes.

The second approach is to fine-tune the entire pre-trained network rather than just the final layer. Once again, you'll replace the final fully connected layer of the network with a 5-class classifier. However, you'll not freeze the rest of the weights of the CNN.

In both cases, you will take a small 2500 image dataset comprising 5 classes and train (rather fine-tune) the CNN on this small dataset. In your assignment submission, you'll be giving us your trained models. You should train for at least 50 epochs.

We highly recommend that you train these models on GPUs. Both AWS and GCP offer GPU instances. Another great place to get quick access to GPUs cheaply is Papersource (look for the **fast.ai** public image there. It comes pre-loaded with PyTorch, making your setup process easy).

We'll post a script on Slack that will download a small imagenet dataset and create training and validation partitions on a portion of that dataset. You should train using the images in the **train** sub-directory of **tiny-imagenet-5** directory.

You should take a look at the PyTorch transfer learning tutorial at
http://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

For your submission, we'll expect the following in CMS.

- Your Transfer Learning PyTorch code in a file called **assign3.py** that clearly shows that you tried both approaches of Transfer Learning and that you trained these models for about 50 epochs (5 points)

- A text file called **model.txt** containing a string from one of **vgg16, resnet50, resnet34** indicating that you chose vgg16, resnet50, or resnet34 pre-trained model from Torchvision. Choose only one network for your submission and it should correspond to what is in model.txt. Otherwise, it will break our autograder and you'll lose points.
- Your two model files. These should be saved using the torch command (5 points for each working model)
  torch.save(yourmodel.state_dict(),yourmodelfile). Call the first saved model as **model_ft** and the fully tranfer-learned model as **model_cnn**.