Introduction to Computer Vision Final Project

Due Jun. 12 (Wed)

Deep Classification Tasks (Self-supervised, Supervised)

The goal of this project is to help you gain experience with PyTorch, learn how to use pre-trained models provided by the deep learning community, and adapt these models to new tasks and losses. You will use a simple self-supervised rotation prediction task to pre-train a feature representation on the CIFAR10 dataset without using class labels, and then fine-tune this representation for CIFAR10 classification.

- You'll use PyTorch to train a model on a self-supervised task, fine-tune a subset of the model's weights, and train a model in a fully supervised setting with different weight initializations. The CIFAR10 dataset, containing small (32x32) images from 10 classes, will be used. For self-supervised training, you will ignore the labels, but use them for fine-tuning and fully supervised training.
- The model architecture is ResNet18, using PyTorch's implementation, so you don't need to create it from scratch.
- The self-supervised task is image rotation prediction, based on Gidaris et al. (2018). Training images are randomly rotated by 0, 90, 180, or 270 degrees, and the network classifies the rotation using cross-entropy loss. This pre-training helps improve supervised CIFAR10 classification.
- The notebook (FP_rotation.ipynb) guides you through training ResNet for the rotation task and fine-tuning on the classification task. You'll implement the data loader, training, and fine-tuning steps in Pytorch. In detail, you will complete the following tasks:

    1. **Train ResNet18 on the rotation task**: Generate rotated images and labels using the CIFAR10 dataloader. Train ResNet18 on the rotation task, report test performance, and save the model for fine-tuning tasks.
    2. **Fine-tune only the final layers on CIFAR10**: Initializing from the Rotation model or from random weights, fine-tune only the weights of the final block of convolutional layers and linear layer on the supervised CIFAR10 classification task. Report the test results and compare the performance of these two models.
    3. **Train the full network on CIFAR10**: Initializing from the Rotation model or from random weights, train *the full network* on the supervised CIFAR10 classification task. Report the test results and compare the performance of these two models.

Extra Credit

- In Figure 5(b) from the Gidaris et al. paper (2018), the authors show a plot of CIFAR10 classification performance vs. number of training examples per category for a supervised CIFAR10 model vs. a RotNet model with the final layers fine-tuned on CIFAR10. The plot shows that pre-training on the Rotation task can be advantageous when only a small amount of labeled data is available. Using your RotNet fine-tuning code and supervised CIFAR10 training code from the main assignment, try to create a similar plot by performing supervised fine-tuning/training on only a subset of CIFAR10.

Submission instructions: what to hand in:

- Upload the electronic file that includes the report, source code in a single zip format with the name "**ICV_Final Project_yourname.zip**" on the ETL class homepage.
- The report should include the brief description of the problems, results, and discussions

Note: All works should be individual-based. Copy or using existing codes is not allowed.