Assigned: 28 August

Homework #0

EE 641: Fall 2024

Due: Wednesday, 04 September at 15:00. Submission instructions will follow separately on canvas.

1. Construct a residual MLP for image classification on the CIFAR-100 dataset (`https://www.cs.toronto.edu/~kriz/cifar.html`) using PyTorch. Train your model to at least 68% accuracy.

   Review the original ResNet paper:

   > K. He, X. Zhang, S. Ren and J. Sun, *Deep Residual Learning for Image Recognition,* 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.

   Implement the residual network described in Table 1. Use a *block*-architecture to define the convX layers. Combine blocks to construct the end-to-end network. Experiment with different regularization, optimizers, and learning rate schedules and apply data pre-processing and augmentations to improve model performance and generization.

layer name	
conv1	$3 \times 3, 32$, stride 1, padding 1
	BatchNorm, ReLu, Dropout
conv2_x	$\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times 2$, stride 1, padding 1
conv3_x	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 4$, stride 2, padding 1
conv4_x	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$, stride 2, padding 1
conv5_x	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$, stride 2, padding 1
output	max-pool, 100-d fc, softmax

Table 1: Residual architecture. Building blocks are shown in brackets with the numbers of blocks stacked. Downsampling is performed by conv3_x, conv4_x, and conv5_x with a stride of 2.

Use a validation set for hyperperameter tuning (*e.g.,* grid search, random search). Record final loss and accuracy values for each hyperparameter configuration that you evaluate. Plot the training and validation classification accuracy as a function of epoch on separate figures for at least three hyperparameter configurations. Apply your best trained model to the test set and report the final loss and accuracy.

You may refer to the PyTorch ResNet reference implementation but you must implement the model *from scratch*.

https://github.com/pytorch/vision/blob/main/torchvision/models/resnet.py

What to submit

a. Tested hyperparameter configurations with final loss and accuracy values.

b. Accuracy and loss plots for at least three hyperparameter configurations.

c. Determine and report the output sizes for each `convX` layer.