Homework 6: Convolutional Neural Network

November 30, 2025

Due Date: December 14, by 23:59

Introduction

In this assignment, you will implement and test various convolutional neural networks for image classification on the CIFAR-10 dataset. **If the machine you're experimenting on is out of memory, you can attempt to decrease the width and depth of your models.** The goals of this assignment are as follows:

- Implement Data Augmentation to reduce overfitting.
- Implement Batch Normalization for training deep networks.
- Implement Dropout to regularize networks.
- Implement three types of convolutional neural networks for image classification: VGG, ResNet, and ResNeXt.
- Train and test VGG, ResNet and ResNeXt.

Here are some supplementary materials that may help you:

- Transforming and augmenting images
- VGG
- ResNet and ResNext

1 Data augmentation (10 pts.)

Choose at least **2** ways to augment your data to improve the generalization of your models and put the methods in your report.

(Hint: you can learn how to use pre-defined functions in **torchvision.transforms** to implement.)

2 Implement VGG for image classification (20 pts.)

The **VGG** class in **models.py** will walk you through implementing VGG. Besides, you are required to apply **Batch Normalization** and **Dropout** to the model. The VGG16 architecture shown in Figire 1 can be referred to implement your network.

3 Implement ResNet for image classification (20 pts.)

The **ResBlock** class in **models.py** will walk you through implementing residual block. Further, the **ResNet** class in **models.py** will walk you through implementing a residual network based on residual block. You are required to apply **global average pooling** to the model.

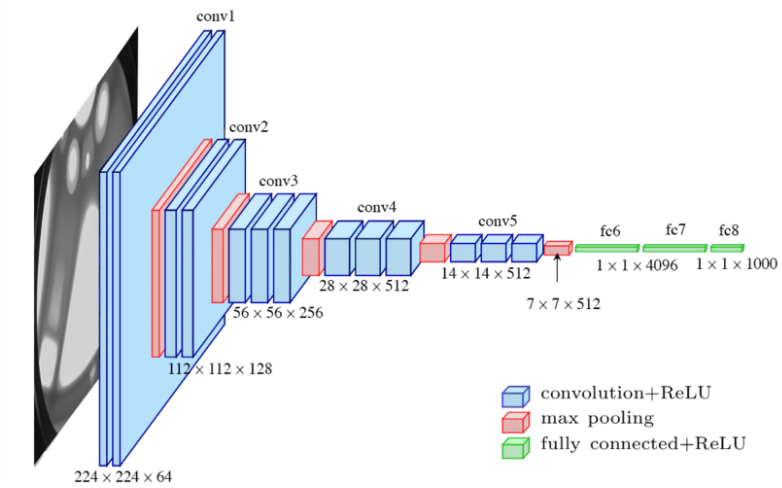

Figure 1: VGG16 architecture.

4 Implement ResNeXt for image classification (20 pts.)

The **ResNextBlock** class in **models.py** will walk you through implementing the ResNext block. Further, the **ResNext** class in **models.py** will walk you through implementing ResNext based on the ResNext block. You are also required to apply **global average pooling** to the model.

5 Train and test (30 pts.)

Add your code to **train()** and **test()** functions in **main.py** to train and test your models (VGG, ResNet, ResNeXt). You need to make sure that your networks are not overfitting. If that occurs, you can try some regularization techniques to address it. Finally, put all of the training and test loss and accuracy curves and the final classification accuracies in your report.

(Hint: For a quick check to make sure you are on the right track, empirically the final classification accuracies should be around or surpass 80%.)

6 Report

You have now completed the entire process of this project. Put all the visualizations and results in your report. More experiments and discussions are encouraged.

7 Submit

Be sure to zip your code and final report. Name it as **StudentID_YourName_HW6.zip.**