Homework 2 Part 1
11-785 Spring 2018

February 12, 2018

1 Introduction

In part 1 of homework 2, we will be training a convolutional neural network on the CIFAR10 dataset. This homework will guide you through loading and preprocessing data, and training and evaluating a model.

We will be building the all-cnn-c model described in `https://arxiv.org/abs/1412.6806`. The model will perform convolutions on the input images to predict one of 10 image classes.

1.1 Autograder Submission

Your solutions will be autograded by Autolab. For this reason, it is important that you do not change the signature of any of the functions contained in the template.

In order to submit your solution, create a tar file containing your code. The root of the tar file should have a directory named hw2 containing your module code and your prediction.txt.

1.2 PEP8 (5 points)

Format your code according to PEP8 guidelines. You might not receive help on poorly formatted code. Modern IDEs should have a shortcut to automatically reformat your code.

In order to check your code for PEP8, install pytest and pytest-pep8 and run the following command.

```
pip install pytest pytest-pep8
pytest --pep8 -m pep8 [path-to-your-code]
```

The autograder will check all PEP8 guidelines except that the maximum line length will be raised to 120.

1.3 Dataset

The dataset is included in the handout and can be loaded using numpy.

The dataset contains 32x32 RGB images. Each file in the dataset contains 10,000 images in a 1000x3072 array. If you reshape the data to 10000x3x32x32, the dimensions are (images, channels, rows, columns). The pixel values are 0-255 unsigned bytes.

2 Preprocessing (40 points)

The first part of this assignment is to complete the functions in `preprocessing.py`. These functions will load and preprocess the CIFAR10 data.

Complete the code in each function without modifying the function signature. Refer to the template code for details. You will receive one point for each function verified by autolab.

- `sample_zero_mean`: subtract sample-wise mean from data
- `gcn`: global contrast normalization (divide by sample-wise standard deviation)
- `feature_zero_mean`: subtract feature-wise mean from data
- `zca`: whiten data using ZCA transformation
- `cifar_10_preprocess`: chain all of the previous functions to process a dataset

Feel free to incrementally submit your results to Autolab to check your progress. Submit your tar file to confirm that you are loading and preprocessing the data correctly before starting work on your model.

3 Building an All-Convolutional Neural Network (30 points)

Build an All-Convolutional Neural Network (ALL-CNN-C) as described in the referenced paper using the hyperparameters described therein. Complete the functions in `all_cnn.py`. Each of these functions will be called by the autograder to test your network.

- `flatten_module`: create a custom module that reshapes data as described in the template
- `all_cnn_module`: create a sequential module containing all 23 layers described in the All-CNN-C Model and the template file

You will create a `torch.nn.Sequential` that takes as input `(samples, 3, 32, 32)` and produces outputs `(samples, 10)` that are the unnormalized pre-softmax activations.

Feel free to incrementally submit your results to Autolab to check your progress. Submit your tar file to confirm that you are building your model correctly.

4 Training the Model (25 points)

Now that you have preprocessed the data and correctly built a model, it is time to train. Parameters described in the paper:

- stochastic gradient descent with momentum of 0.9
- L2 regularization of 0.001 on all weights

Please complete `train_all_cnn.py`. Train your model You should see 70% accuracy within minutes. This vanilla model (no augmentation, smoothing, etc.) should achieve around 90% accuracy.

Use the provided function `write_results` to write your results to a file and include in your Autolab submission. Write your predictions on the test set in order. You will receive credit for reaching at least 70% accuracy.

Submit your predictions as a text file named `predictions.txt` in the root of your tar file.