Homework 5 Programming Assignment

10-605/10-805: Machine Learning with Large Datasets

Due Thursday, November 19th at 1:30:00 PM Eastern Time

1 Introduction

In this homework you will explore methods for neural network pruning as part of a class-wide competition. In particular, you will attempt to compress a simple neural network which has 592,933 parameters in total. It can achieve about 63% test accuracy in a 5-class classification task after training for 50 epochs with the default parameters provided in the notebook. To help you start, we provide a simple example of manipulating the weights, assigning the new weights to the model, and evaluating how that affects the accuracy in the notebook. You can explore various pruning techniques as you like. You might also want to read some references e.g. this or this.

2 Logistics

We will provide a notebook for you for this part which provides data loading helper functions, defines the model architecture, and provides some simple functionalities to save and load model weights for you to start with. To download the notebook for this part, go to this link. Make sure to use your andrew id. This homework is done on Colab.

2.1 Getting the data

Download the compressed training and validation data from this link using your andrew id. Upload it to Colab by going to the `Files` tab in the sidebar and clicking on the upload button. The first few cells in the notebook untar and load the training and validation sets. We will use a hidden test set on gradescope to test the final performance of your model. To avoid uploading the dataset everytime the runtime disconnects, you might want to upload the dataset to drive and mount it from colab.

2.2 Training and pruning the network

You need to first train a network in the notebook until convergence. Starting from this pre-trained full model, you need to explore different strategies to prune the network (i.e., setting some weights to zero) without degrading the accuracy. Your output should be a pre-trained model with the '.h5' extension with the same architecture as the neural network we define in the notebook. Changing

the architecture or designing a new model architecture yourself is not allowed and will cause the autograder to fail.

2.3 Submission

You do not need to submit the notebook, but instead, have to submit the final pruned model with the name **"my_model_weights.h5"**. In the notebook, we specify how to download the model weights to your local work space. You will then upload the model weights to Gradescope for autograding. You can submit an unlimited number of times, and the final ranking will be determined by the last submission.

2.4 Evaluation

Each submission will receive a score, which is a function of the accuracy of the test data and the sparsity level of the model. We assume there is an efficient way to encode the indices of sparse matrices, therefore do not take into account the storage overhead for sparse matrix formats. In particular, the autograder uses the following functions to calculate the score of a model:

if accuracy > 0.6 and sparsity > 0:
    score = (accuracy + num_zero_weights / total_parameters) / 2
else:
    score = 0

This evaluation metric encourages to achieve a good trade off between accuracy and sparsity (the ratio between the number of zero weights and the number of total parameters) while ensuring a reasonable accuracy (accuracy > 0.6). You must obtain an accuracy higher than 0.6 and a total **score higher than 0.36** to receive full credit (though we encourage you to search for even higher scores as part of the competition!)

3 The competition

We will maintain a leaderboard on gradescope and plan to give bonus points to the top 5 submissions in this competition. In such competitions there is always a risk of data leakage but we expect that you will only use the provided train/validation data to train your model. At the end of the competition, we will ask the top 5 participants to submit their code and we will manually check to make sure that they used only the provided train/validation data. Note that you can't use any network pruning libraries in Tensorflow and we will manually verify the top 5 submissions to make sure that's not the case.

We hope that the competition is a fun learning experience for those who want to dive deep into model compression.