CSC 180-01
Mini-Project 3
Friday October 25 2019

Computer Vision using GPU and Transfer Learning

By
Warren Quattrocchi (Team Leader) ID# 219714665
Janus Kwan ID# 212910387
Andrew Wright ID# 218771424

(1) Problem Statement

In this project, the goal is to practice with image classification using Google GPU and transfer learning. The Google GPU is utilized with Google Colaboratory, a cloud-based Jupyter notebook environment that is free to use and requires no setup. Transfer learning is a machine learning technique where pre-trained models are used as the starting point on computer vision and natural language processing tasks. Transfer learning is an optimization that allows rapid progress or improved performance when modeling the second task. The project is twofold. In the first part, a CNN model is trained and tested on the Google GPU without transfer learning. In the second part, a CNN model is trained and tested on the Google GPU with transfer learning.

This project uses the CIFAR-10 dataset, which is preinstalled with Tensorflow. The CIFAR-10 dataset consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

(2) Methodology

The methodology for this project was simple. The base of the project was provided with empty sections in which we had to fill in code to get it all working. The data for the test and train set was all provided, the most we had to do was change the shape of the dataset. First step was to add in a CNN model for regular model training. We used the sample code provided in a previous lab and put it here. Tweaked the parameters a little to see if we could get an improvement in the accuracy, which remained around 70%. We kept in early stopping as instructed, but took out check pointing as it was throwing some errors and we could not figure out why. For the assisted learning we changed the y train and y test data shape, and applied one hot encoding to them. We then applied every single layer from the VGG16 model, besides for the top layers. We then freeze the model as it currently is then add on dense layers to improve the training. Next step was to add in the code for early stopping and train/test of the data. After going through extremes in the parameter values for the dense layers, we found that changing it did little, at most 2-3% change to the accuracy.

(3) Experimental Results and Analysis

We were told to skip Parameter Tuning.

We still attempted some, however the results would be a difference of 1-2% in either direction for both assisted learning and non assisted. One thing we did find was that it was better to do this on our own GPU rather than google's. For example, it takes an average of 50 seconds per epoch with assisted learning on google colab, on my GPU at home it takes about 30 seconds. One improvement we found was that using adagrad as our optimizer resulted in a model with .74 accuracy, but also produced inconsistent results with some models giving <.70, so we decided to stick with Adam for consistency.

(4) Task Division and Project Reflection

Similarly to Project 1 and 2, we created an initial working model of the normal CNN to make sure we could get everything working before working on the Transfer Learning model . This project had much less freedom than the previous two, with the dataset being given to us test-ready. So we didn't spend any time on data pruning and went straight to model training. Dividing the work in this Project proved more difficult than the previous two, but we all tried to different parts of the project whenever possible with Andrew doing the regular CNN model, Warren with the Transfer Learning model, and Janus with Parameter Tuning and code clean-up.

The biggest problem we had with this project was getting poor results from both models (~70%). We spent a large amount of time parameter tuning to get a better accuracy/f1 score but in the end we decided it was due to the low resolution of the photos in the dataset, and getting better results wasn't realistic using the cifar-10 dataset.