Efficient Dataloader for Deep Learning Framework

Team members: Bowen Chen

Url

https://vertexc.github.io/efficient-data-loader/

Summary

In this project, we will looking into different levels of parallelism to accelerate dataloader in deep learing framework.

Background

In deep learning, to train the model, we first need to load and preprocess the training and testing dataset. Take image data as an exmaple, we need to

- 1. read image data from some compressed data format into memory and organized as an dataset
- 2. use dataloader to load images and apply different types of transformation (crop, flip)
- 3. we may also want to shuffle the dataset at each training epoch

There are different levels we can potentially apply parallelism, like add more workers (multiprocess, multithreads) to process data, and make transformation operations more efficient (cuda).

Goals and Deliverables

Plan to Achive

We plan to at least achive speed up compared to the baseline, which is a serial data process and transformation implemented in numpy. And compare the performance to pytorch's dataloader and run analysis.

Hope to Achive

We hope we can achive a better performance compared to pytorch.

Platform Choice

The project is going to be implemented in Python, C++, Cuda.

We are going to evalaute the performance on our own desktop. (CPU:Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz, GPU: NVIDIA Corporation GP106 [GeForce GTX 1060 3GB])

The Challenges

First of all, it is non-trivial to write efficient parallel code in python and there are multiple ways needs to explore (either multithread, multiprocess, if multiprocess, fork or spawn).

Secondly, we need to implement multiple cuda kernels, potentially including crop, flip (horizontal and vertical), permutate, which takes time to make sure correctness and may acuqires multiple iterations to optimize it.

Meanwhile, the python level and cuda level parallelism may not be fully independent, we expect there will be potential issues that we need to resolve.

Resources

We are going to use `needle`, which is a toy deep learning framework we developed at `10-414/714 - Deep Learning Systems: Algorithms and Implementation` as backbone code for this project.

Schedule

The project's implementation basically contains two parts, one is python level multi-worker data process, and another is cuda kernels implementation.

- **Week1 (11.8-11.14)**, we will explore different ways to implement mutli-worker data processor in python.
- **Week2 (11.15-11.21)**, we will implement basic cuda kernels and ensure correctness. (Evaluate speedup compared to baseline in milestone report)
- **Week3 (11.22-11.28)**, we will analyze potential bottleneck in current implementation, and try to resolve it.
- **Week4 (12.29-12.5)**, we will focus on optimize cuda kernels.
- **Week5 (12.6-12.10)**, we will wrap up the code, finish the final report and prepare for the poster session.