EECS 498/598

Final Project

Due date: December 4, 2020 11:59pm

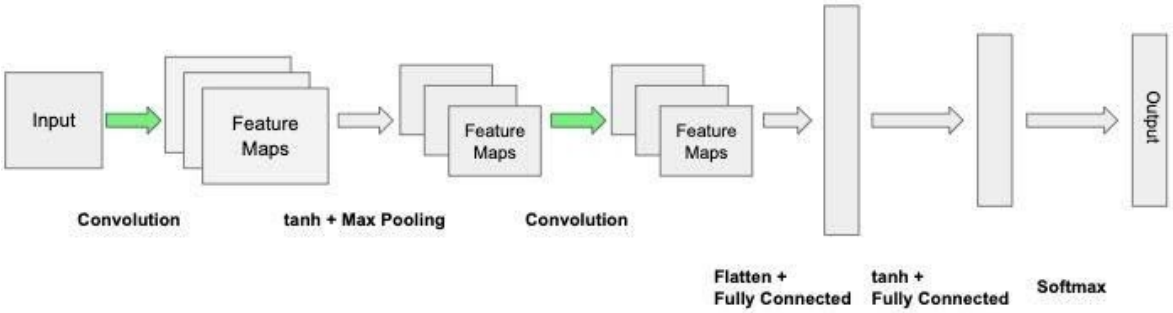
Introduction

- Get practical experience by using, profiling, and modifying MXNet, a standard open-source neural-network framework.

- Demonstrate command of CUDA and optimization approaches by designing and implementing an optimized neural-network convolution layer forward pass.

For a background and introduction to Convolutional Neural Networks we suggest this video.

You will be using a pretrained CNN to predict images from the MNIST-Fashion dataset. The starter code given to you is a correct implementation of the convolution layers and your optimized versions should have identical functionality.



You must implement and document at least **2** optimizations.

Grading

70% correctness and speed

- Correctness: you must have 0.7955 to receive any points on the project

-Speed: you are expected to achieve under 2 seconds for both layers combined

-Your implementation must match the same functionality as the starter code, you will receive 0 points for an incorrect implementation.

30% final project report

-See final report section for what is expected from you.

Set up

You will be developing on great lakes.

Run the following two commands in sequence to pull the code for the final project into your personal workspace and setup (note, this may take a while):
tar -xzvf /scratch/eecs498f20_class_root/eecs498f20_class/shared_data/final_project.tar.gz

python get-pip.py --user

Directory Layout

Once you pull the code into your workspace, there will be a folder named “final_project” with the following items

final_project

-fashion-mnist (d)
-incubator-mxnet (d)
-models (d)
-submit (d)
-new-forward.cuh (f)
-run_student.sh (f)
-run_mxnet_student.sh (f)
-build_student.sh (f)
-get-pip.py (f)

fashion-mnist contains images from the fashion-mnist dataset that your CNN will be classifying.

incubator-mxnet contains the source code for the mxnet library, the code you write will be compiled into this with your custom implementation.

models contains a JSON description of a pretrained CNN that will use your implementation of the forward pass convolution layer to classify the fashion-mnist images.

submit contains a python script that you will use to run your CNN, submission.py, as well as a submit_code script that you will use to submit your code for grading.

new-forward.cuh is the only file that you should be modifying. We have provided scripts to help you with running your code. Run **./build_student.sh** to compile your source code into mxnet, which will allow you to run your implementation by running **./run_student.sh**. The profiling info (including timing) will be available in the generated slurm output.

To submit your code, simply run `**./submit/submit_code <new_forward.cuh>** `**.** This will copy your file into the **all_sub** folder, similar to how your other assignments were submitted.

You will be graded based on what source code you have submitted via the submission script in great lakes. Make sure that you have your final, working version there at the time of the deadline, as only the latest version will be graded

Optimizations

You must implement at least 2 optimizations to the forward pass convolutional layer. You have starter code of an extremely naive but correct implementation in the file new-forward.cuh. You can implement the optimizations separately (i.e. they do not need to build on each other), however you must document each optimizations performance effect within your final report and be sure to keep the functions of each optimization you write within your file for your final submission. (see final report)

You are encouraged to be creative with your optimizations! Here are some suggestions that you could consider, but feel free to implement one not included below.

Suggested optimizations:

- Shared Memory convolution
- Weight matrix (kernel values) in constant memory
- Loop unrolling
- Unroll + shared-memory Matrix multiply
- Kernel fusion for unrolling and matrix-multiplication
- Exploiting parallelism in input images, input channels, and output channels
- Multiple kernel implementations for different layer sizes
- Sweeping various parameters to find best values (block sizes, amount of thread coarsening)

Final Report
As you implement your optimizations, you are required to document their effect on performance.

Create a document and describe each optimization you implemented, including why you selected this optimization, screenshots of profiler output, and the output of each layer's timing with this optimization. Describe in detail how you implemented the optimizations.

Leaderboard and Prizes
Top 3 on the leaderboard (coming soon!) will get prizes sponsored by
https://www.kla-tencor.com/

John McLaughlin
Senior Director, Ann Arbor Site Leader