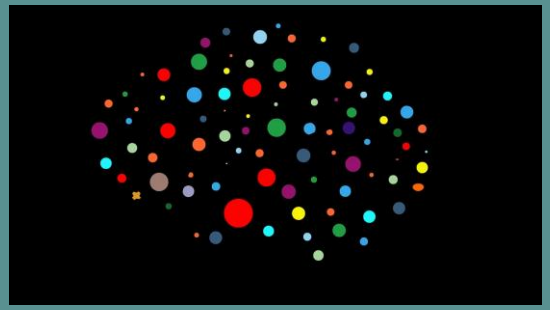
Convolutional Neural Network

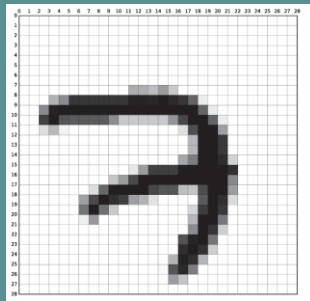
December 9, 2020

ECE 4900 Final Project



Matthew Horvath and Ryan Marten

Project Description



(a) MNIST sample belonging to the digit '7'.

(b) 100 samples from the MNIST training set.

Motivation for Project

- We have worked on simple FCN in hardware before (VHDL)
- Plenty of simple FCN prototypes online for aid
- Wanted to add convolutional layers to further improve digit recognition
- Convolutions offer a great opportunity to implement multithreading and parallelization of software to improve efficiency
- CNNs are the approach of choice for addressing complex image recognition tasks.
- TBB vs Sequential time comparison benefits

Specifications

- MNIST handwritten database used for training/testing
  - 60,000 training samples
  - 10,000 testing samples
  - 28x28 image
- 784 inputs -> 10 outputs (0-9)
- CNN: 2 Layers
  - Conv1: 6@24x24, Pool1 6@12x12
  - Conv2: 24@8x8, Pool 2 24@4x4
  - Uses 5x5 kernel for each convolution
- FCN: 3 Layers (Input, 1 Hidden, Output)
  - Input layer: 384 neurons
  - Hidden Layer: 128 neurons
  - Output: 10 neurons (Represents 0-9 digit options)

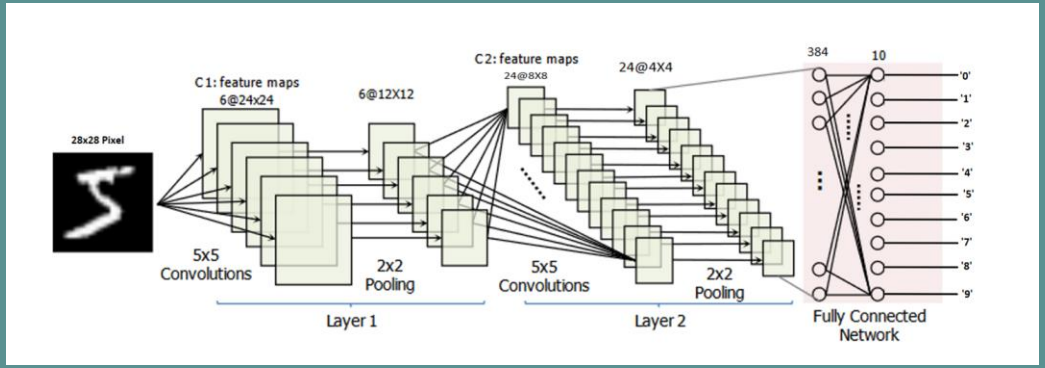
Overview of Implementation (Sequential vs Parallelization)

Sequential

- Straightforward approach with layers of convolution, activation, and pooling with the result being fed into a FCN.

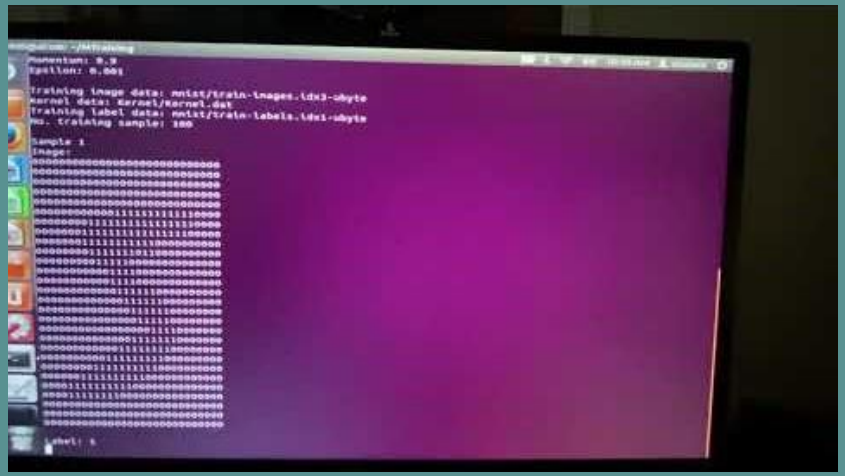
Parallelization

- Implements parallel_for for each set of convolutions
  - 1 for the C1 x 6
    - Additional parallel_for implemented on each epoch for training
  - 1 for the C2 x 24



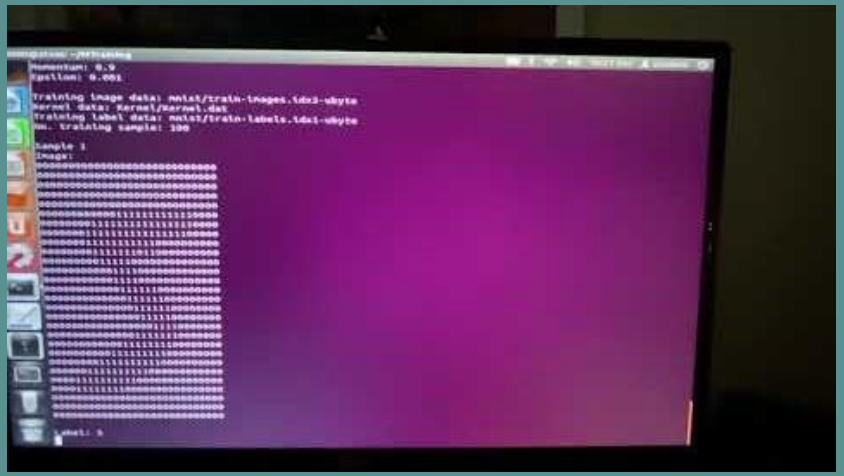
Training Sequential vs TBB

Sequential



https://youtu.be/L1DCfD82Y0I

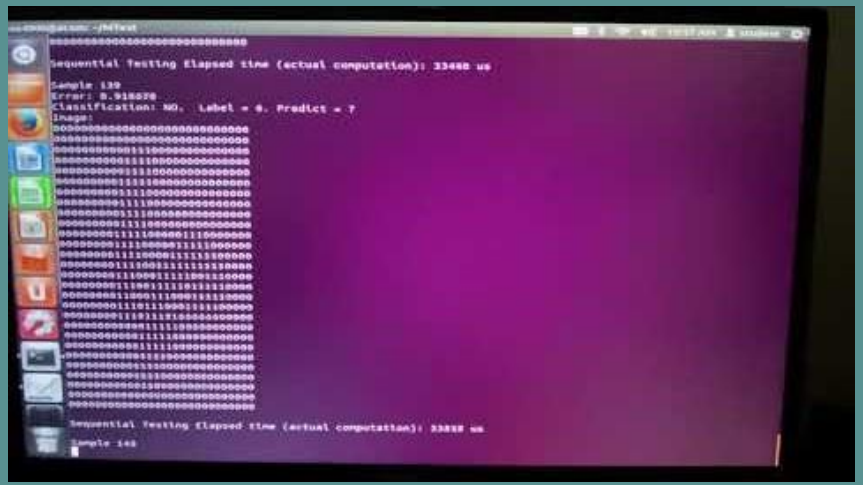
TBB



https://youtu.be/yYyX8eImCl8

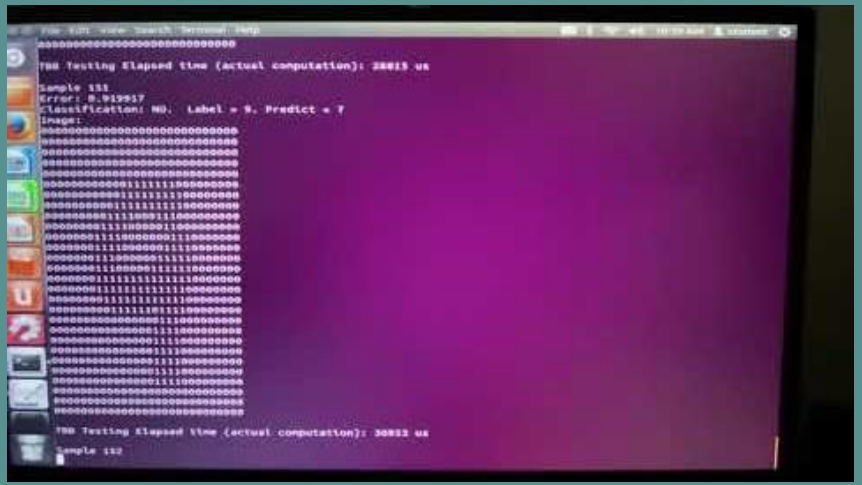
Testing Sequential vs TBB

Sequential



https://youtu.be/i2h7w4CZJuw

TBB



https://youtu.be/QnPNsxg0nwM

Results

Sequential

- Trained 100 Images at 512 Epochs
  - CNN Time: ~ 2.86 s/epoch
  - Total Time: ~ 3.42 s/epoch
  - Total Time: ~ 17.53 s/image
- Testing 1000 Images
  - CNN Time: ~ 3.12 s
  - Total Time: ~ 3.306 s

```
Sequential Training Timings
Average CNN Time: 28668 us
Average FCN Time: 1824 us
Average Back Propagation Time: 3739 us
Average Time for each computation (Per Epoch): 34250 us
Average Time for each computation (Per Training Image): 17536053 us
```

```
Sequential Testing Timings
Average CNN Time: 31202 us(C1: 6753, C2:24433)
Average FCN Time: 1860 us
Average Time for each computation: 33067 us
Number of correct samples: 99 / 1000
Accuracy: 9.90
```

TBB

- Trained 100 Images at 512 Epochs
  - CNN Time: ~ 0.98 s/epoch
  - Total Time: ~ 1.80 s/epoch
  - Total Time: ~ 9.24 s/image
- Testing 1000 Images
  - CNN Time: ~2.80 s
  - Total Time: ~2.97 s

```
TBB Training Timings
Average CNN Time: 9892 us
Average FCN Time: 2411 us
Average Back Propagation Time: 5882 us
Average Time for each computation (Per Epoch): 18049 us
Average Time for each computation (Per Training Image): 9241524 us
```

```
TBB Testing Timings
Average CNN Time: 28010 us(C1: 8003, C2:19992)
Average FCN Time: 1771 us
Average Time for each computation: 29785 us
Number of correct samples: 99 / 1000
Accuracy: 9.90
```

Notes

The CNN time improvement seen on the training is not the same for the testing. Why?

When training only only 1 epoch, the CNN improvement falls off to almost zero.

```
tbb::parallel_for (int(0), int(epochs), [&] (int z) {
    tbb::parallel_for (int(0), int(6), [&] (int i) {
            conv1(i);} );
```

```
TBB Training Timings
Average CNN Time: 24717 us
Average FCN Time: 1794 us
Average Back Propagation Time: 3749 us
Average Time for each computation (Per Epoch): 31031 us
Average Time for each computation (Per Training Image): 31031 us
```

Conclusion

Improvements

- Parallel_pipeline and parallel_reduce for both the testing and training of the data
  - Optimize Parallel_for loops
  - Deeper embedding of parallelization
- Add in back propagation for kernel values
  - Currently trains only the weight values for the FCN
  - Randomized kernel values were used for the training, then brought over for use in the testing
- An increase in accuracy through longer training iterations
  - Trained single batch of 15k
  - Increase average training size from 500