Image Processing Library

Gohil Dwijesh, Sahil Dahake

2019 February

Computer Science and Engineering
Indian Institute of Technology, Delhi

iddwijesh@gmail.com
sahildahake@gmail.com

1 Creating a framework

- Convolution
  - Implemented two functions that takes two square matrices as an input and returns a matrix which is convolution of the two matrices. The first function uses standard formula to convolve two matrices and the second function uses matrix multiplication to convolve two matrices.
- Some other functions, like Relu, tanh, max pooling, avg pooling, softmax and sigmoid function.
- We will use these functions in building convolutional neural network architecture.

2 Accelerating Matrix Multiplication

2.1 Linear Algebra Libraries

- Openblas
  - OpneBLAS is open source library for Basic Linear Algebra Subprograms. It provides efficient functions to implement matrix operations(official site[5]).
  - To squeeze the most power of the CPU is to go to the lower level from the programmer's perspective: assembly. By writing in assembly we limit ourselves to specific CPU architecture and to specific CPU features like AVX(Advanced Vector Extension) that boost things up. All the algorithms in OpenBLAS are rewritten to specific CPU family resulting in very efficient and fast matrix operatinos. [1].

- MKL(Math Kernel Library)
  - Intel MKL offers highly-optimized and extensively threaded routines which implement many types of operations[2].
  - Intel MKL provides several routines for multiplying matrices. The most widely used is the dgemm routine, which calculates the product of double precision matrices[4].

2.2 Acceleration with pthread[6]

- In order to take full advantage of capabilities of threads, we will use threads for matrix multiplication.
- We have used four threads(bcz only four threads can work concurrently in my pc), deviding the the first matrix into four parts and concurrently multipling these matrices with the second matrix.
- Memory location corresponding to answer matrix is shared by all four threads, so we synchronized the answer matrix and as a subsequence only one thread can modify at perticular memory location in the answer matrix at a time.

2.3 Performance comparision

size of the matrices	Time taken by **simple impl.**	Time taken by **pthread impl.**	Time taken by **opneblas impl.**
3	6.00E-07 s	2.99E-04 s	4.90E-07 s
10	1.60E-05 s	1.15E-04 s	2.30E-06 s
30	2.59E-04 s	2.49E-04 s	9.51E-06 s
100	9.33E-03 s	4.49E-03 s	4.70E-04 s
300	2.46E-01 s	1.14E-01 s	1.69E-03 s
500	1.21E+00 s	5.56E-01 s	6.30E-03 s

Table 1: Comparing avg. time taken by different implementations to multiply two matrices of same size n*n.

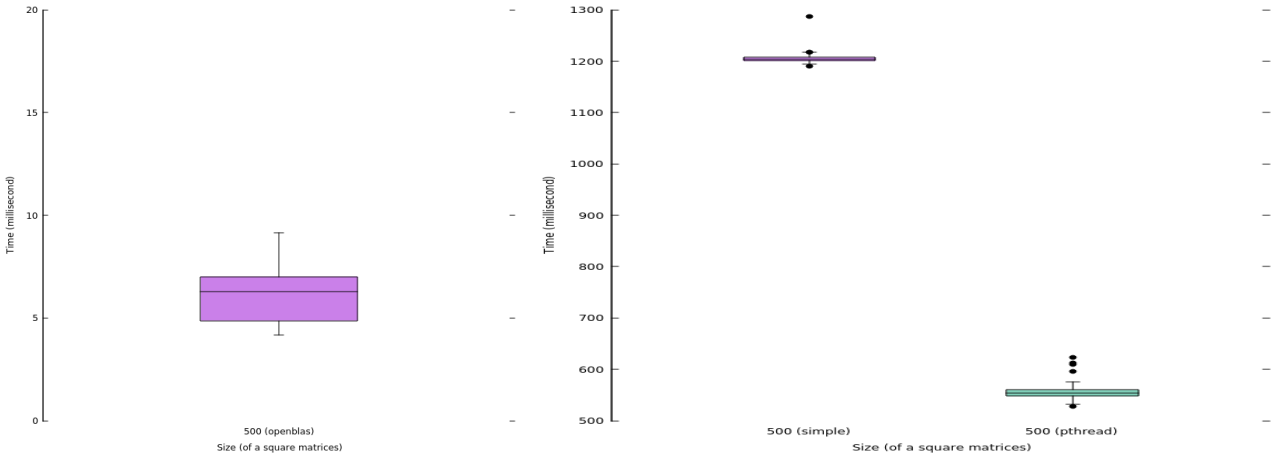

Figure 1: Gnuplot[7] to compare three implementations for size of matrices = 500. First fig. shows openblas impl., second fig. shows comparision b/w pthread and simple impl.

2.4 Camparision results

- It is clear from the table 1 and figure 1 that Openblas is the best out of three implementations for matrix-matrix multiplication.
- Pthread implementation takes almost half the time taken by simple implementation for large value of n(size).

3 LeNet-5 – A Classic CNN Architecture[3]

- The order in which different functions are called is what defines a neural network architecture.
- LeNet-5 uses convolution, max pooling and softmax functions to map an image to correspodning probabilities.
- We have implemented the CNN with six hidden layers, one input and one output layer.
  - first layer: convolution layer.
  - second layer: max pooling layer.
  - third layer: convolution layer.
  - forth layer: max pooling layer.
  - fifth layer: convolution layer (fully connected).
  - sixth layer: convolution layer (fully connected).
- We used Pthread impl. to convolve two matrices.

- We have a python program that converts image in text form to preprocessed image data and then we use this preprocessed data as an input to the input layer.
- Now to automate the process we have a main c++ program that invokes the python3 interpretor and runs that python file and the python file stores preprocessed data to some file. After this, c++ program closes interpretor and uses that preprocessed data from the file to map images to corresponding probabilities.
- So we have a c++ program that takes file names of image and weights as an input and returns corresponding probability values on std Output.

References

[1] *How does it work and OpenBLAS matrix multiplication example*. URL: https://www.konda.eu/c-openblas-matrix-multiplication/.

[2] *Introduction to the Intel® Math Kernel Library*. URL: https://software.intel.com/en-us/mkl-tutorial-c-introduction-to-the-intel-math-kernel-library.

[3] *LeNet-5 – A Classic CNN Architecture*. URL: https://engmrk.com/lenet-5-a-classic-cnn-architecture/.

[4] *Multiplying Matrices Using dgemm*. URL: https://software.intel.com/en-us/mkl-tutorial-c-multiplying-matrices-using-dgemm.

[5] *Openblas: An optimized BLAS library*. 2017. URL: http://www.openblas.net/.

[6] *POSIX Threads Programming*. URL: https://computing.llnl.gov/tutorials/pthreads/.

[7] *Understanding and interpreting box plot*. URL: https://www.wellbeingatschool.org.nz/information-sheet/understanding-and-interpreting-box-plots.