**Write-up -> COP Submission #2**

**Raval Vedant Sanjay: 2017CS10366**

**Shreya Sharma: 2017CS50493**

* The name of our file to be executed is: main.cpp
* There are other files in the zip which also contain the corresponding functions: others.cpp, sourceMKL.cpp, sourcePT.cpp, and sourceOB.cpp. We have also included their header files as well.
* There is a make file which compiles all the programs by the command “make” in the command prompt.
* The program is to be executed by the command: . /library in the command prompt
* We have added all the files containing the main calling function, the required functions, the make file, the readme file, the text files containing the values of the time taken for the different operations obtained by us, the plots, a image.txt file of 512\*512 size for running the program (You can add your own text file in the project folder to test it. There is no constraint on the name of the file taken as the input), GNU plot scripts, and the plots obtained by us for comparing the different modes of matrix multiplication.
* The different functions and their descriptions are given as follow (Be sure that valid existing files are given as inputs):
  + convoWithoutPadding-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2
    - Here, dim\_matrix1 and dim\_matrix2 should be valid integers, else an error will be thrown (For eg. If dim\_matrix1 \* dim\_matrix1 is NOT equal to the total number of floats in matrix1.txt, then an error is thrown.)
    - It implements the Convolution of the two given square matrices, without using the matrix multiplication. Here, no padding is added.
    - Here, matrix1 is the Input and matrix2 is the kernel, thus error if dim\_matrix2 >= dim\_matrix1
  + convoWithPadding-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2-padding
    - Here, dim\_matrix1 and dim\_matrix2 should be valid integers, else an error will be thrown (For eg. If dim\_matrix1 \* dim\_matrix1 is NOT equal to the total number of floats in matrix1.txt, then an error is thrown.)
    - It implements the Convolution of the two given square matrices, without using the matrix multiplication. The program calculated the padding value and returned an error if the size of the kernel is even, since in that case, the size of the output can never be equal to the size of the input, in the previous assignment submission. But in order to make the program more flexible, we decided to take the padding value as an input so that we don’t restrict the output to be the same size as the input.
    - Here, matrix1 is the Input and matrix2 is the kernel, thus error if dim\_matrix2 >= dim\_matrix1
  + multWithoutPadding-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2
    - Works same as the convoWithoutPadding function, the only difference here being the matrix multiplication implemented here.
  + multWithPadding-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2-padding
    - Works same as the convoWithPadding function, the only difference here being the matrix multiplication implemented here.
  + multWithoutPadding\_MKL-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2
    - Works same as the convoWithoutPadding function, the only difference here being the matrix multiplication implemented here using Intel MKL.
  + multWithPadding\_MKL-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2-padding
    - Works same as the convoWithPadding function, the only difference here being the matrix multiplication implemented here using Intel MKL.
  + multWithoutPadding\_OB-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2
    - Works same as the convoWithoutPadding function, the only difference here being the matrix multiplication implemented here using OpenBLAS.
  + multWithPadding\_OB-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2-padding
    - Works same as the convoWithPadding function, the only difference here being the matrix multiplication implemented here using OpenBLAS.
  + multWithoutPadding\_pthread-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2
    - Works same as the convoWithoutPadding function, the only difference here being the matrix multiplication implemented here using pthreads.
  + multWithPadding\_pthread-matrix1.txt-dim\_matrix1-matrix2.txt-dim\_matrix2-padding
    - Works same as the convoWithPadding function, the only difference here being the matrix multiplication implemented here using pthreads.
  + maxPool-matrix1.txt-rows\_matrix1
    - Here, rows\_matrix1 should be a valid integer, else error (For eg. The total no. of floats in matrix1.txt is NOT equal to rows\_matrix1 \* rows\_matrix1, showing that the dimensions of the matrix are wrong)
    - It implements the max pooling function for the given input matrix.
  + avgPool-matrix1.txt-rows\_matrix1
    - Here, rows\_matrix1 should be a valid integer, else error (For eg. The total no. of floats in matrix1.txt is NOT equal to rows\_matrix1 \* rows\_matrix1, showing that the dimensions of the matrix are wrong)
    - It implements the average pooling function for the given input matrix.
  + sigmoid-vector1.txt-dim\_vec
    - dim\_vec should be a valid integer.
    - Implementing sigmoid function for the given vector.
  + softmax-vector1.txt-dim\_vec
    - dim\_vec should be a valid integer.
    - Implementing softmax function for the given vector.
  + relu-matrix1.txt-dim\_matrix1
    - dim\_matrix1 should be a valid integer.
    - Implementing Relu function for the given matrix.
  + tanh-matrix1.txt-dim\_matrix1
    - dim\_matrix1 should be a valid integer.
    - Implementing Tanh function for the given matrix.
  + All the other input formats are invalid and will show an error.
* Explanation for the nature of graphs:
  + We plotted the graphs of MKL, OB and Pthread for the input matrix sizes of 32,64, 128.
  + We kept the size of kernel to be fixed as 4.
  + We measured the time for the overall operation, and not just the multiplication times for the asked comparison.
  + Rather than running the operation once for random matrices, we did the test by running the operation multiple times on the same inputs since each time the program is run, it will take a different amount of time taken which are not different by the orders of magnitude which will also be expected by running the operation on different matrices randomly generated for a single time, if we are not taking the range of the allowed values of the matrices to be too huge and it’s certainly not a good choice since the time taken would differ too much and wouldn’t help in proper comparison.
  + Now since we measured the time of all the operations, we expected only a slight variation in the running times of the functions (This was also posted on piazza but no one responded and so we thought that we should obtain the time taken for the entire operation for the comparison)
  + We observed that the behaviours of MKL and OB were nearly similar while pthread differed more from them and also it turned out to be less efficient as such in all the cases considered.