UC Riverside

Final Project Report

**Implementing a Neural Network**

**in**

**CUDA C**

Report by,

Rajat M. Jain

rjain032@ucr.edu

December 3rd, 2021

Aim: To implement a single hidden layer perceptron in CUDA C.

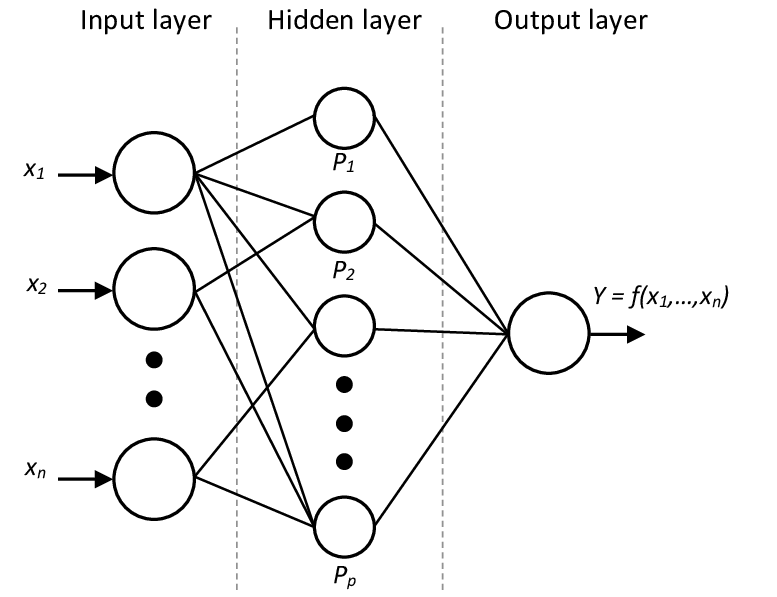
1. Motivation:

While studying Neural Networks in my undergraduate degree, I always wondered how we could speed up the process of training a Neural Network since it has multiple linear algebra operations to be done on the input matrix. The only conventional way to execute it earlier was with the help of loops, where looping through each element of the matrix and then operating on them consumed a lot of time. With the knowledge of GPU and Parallel Programming, this serial approach could be transformed into a parallel programming approach harnessing the suitable architecture of the GPU.

1. Description:

Implementing a neural network on the very low level is just operating functions like multiplication and subtraction on matrices. Taking my understanding from Lab 3 I define several kernels which perform different kinds of matrix operations like multiplication, dot product, subtraction and even performing the Sigmoid activation function. Since it is a single hidden layer perceptron my code even facilitates backpropagation and updating of weights. In my approach I do not use any biases.

1. Graphical representation of a single hidden layer perceptron:



1. Planned tasks:
   1. 11/22/21-11/29/21: Understanding foundations of Neural Networks, and research on implementation methods and selecting a data set.
   2. 11/29/21-12/10/21: Implementing a Neural Network using the selected data set.
2. Tools:
   1. CUDA C
   2. Visual Studio Code
   3. Iris dataset
3. Expected outcome:
   1. Understanding of parallelizing the implementation of a neural network.
   2. Predicting the values and calculating the error function.

Conclusion: Parallelizing the implementation of a single hidden layer perceptron increases the speed of obtaining predictions from the binary classifier and we observe better efficiency due to appropriate memory management. Complex concepts of Neural Networks can be broken down into rudimentary matrix operations.

References:

1. <https://docs.nvidia.com/cuda/>
2. <https://stackoverflow.com/questions/27590166/how-to-compile-multiple-files-in-cuda>
3. <https://www.youtube.com/watch?v=gAgZkdTF4KQ&t=1s>
4. <https://developer.nvidia.com/blog/even-easier-introduction-cuda/>
5. https://github.com/UCR-CSEE217/lab3-matrixmultiply-rajatjain007