Artificial Neural Network using CUDA Python -Implementation from Scratch

Ushasi Chaudhuri & Megh Shukla

ML - Project Presentation
EE769, IIT Bombay

May 2018

Introduction

Outline of the project:

- Implementation of Artificial Neural Network in Python from scratch.
- Used a GPU accelerated computing mechanism (using CUDA Python).
- Verified the working of the model on a Stock market dataset.
- Designed a GUI for making our model user-friendly (No hardcoding), using PyQt.
- Made an executable using PyInstaller.

Artificial Neural Network

Hyper-parameters included in our model:

Weight Decay -

$$\Delta w_i(t+1) = -\eta \frac{\partial E}{\partial w_i} + \alpha \Delta w_i(t) - \eta \lambda w_i \tag{1}$$

- Momentum Factor
- Regularization
- Learning rate

Activation Function (Sigmoid) -

$$s(x) = \frac{\exp(x)}{1 + \exp(x)} \tag{2}$$

- Sigmoidal Gain
- Threshold Value

Graphics Processing Unit (GPU)

Definition (GPU)

A *graphics processing unit (GPU)*[a] is a specialized electronic circuit designed to rapidly manipulate and alter memory accelerate the creation of images in frame buffer intended for output display device.

[a] https://en.wikipedia.org/wiki/Graphics_processing_unit, Accessed on 2.4.2018

Why do we need GPU for this model?

- Accelerates C or C++ applications by updating the computationally intensive portions of the code to run on GPUs.
- Python libraries such as TensorFlow, Keras, implement GPU acceleration using C/C++.
- Not many well defined GPU libraries are there for Python.

Compute Unified Device Architecture (CUDA)

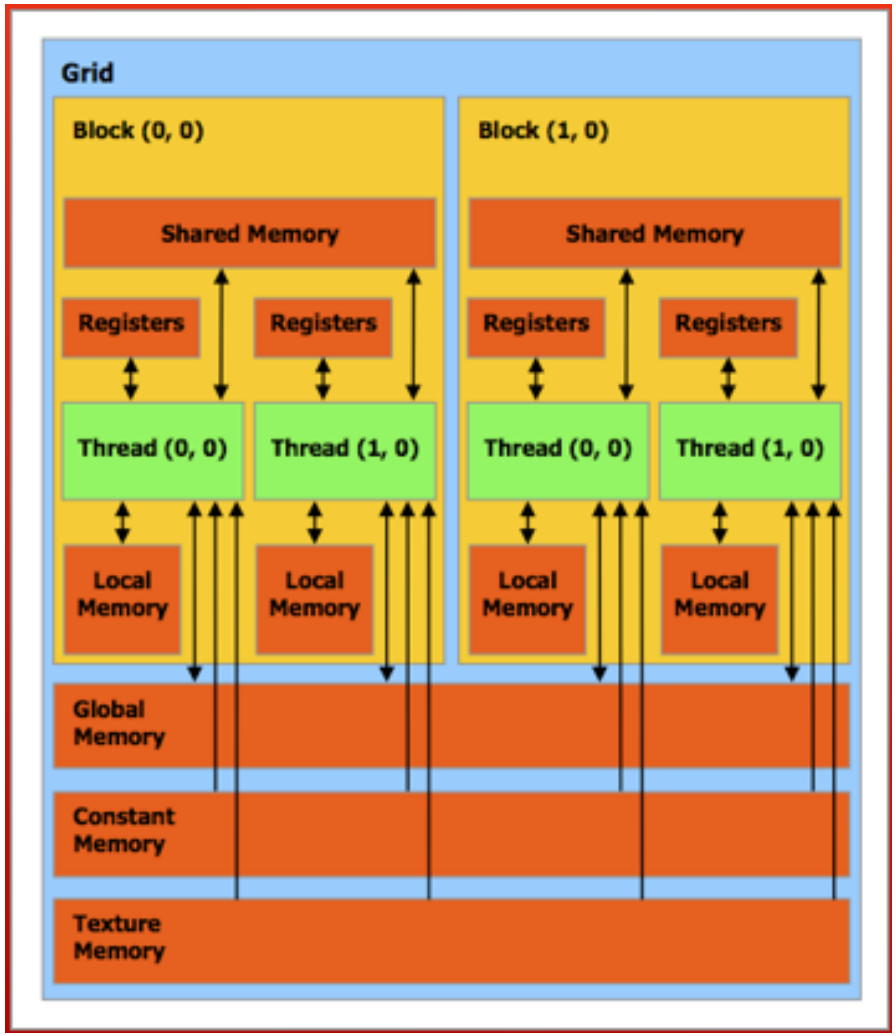
What is CUDA?

CUDA is a parallel computing platform and application programming interface (API) model created by Nvidia.

- Allows software developers to use a CUDA-enabled graphics processing unit (GPU) for general purpose processing.
- CUDA platform is designed to work with programming languages such as C, C++, and Fortran.
- It is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements.
- Third party wrappers are available for Python, hence interfacing CUDA with Python is considered difficult.

Compute Unified Device Architecture (CUDA)

CUDA Architecture [1]



[1] http://cuda-programming.blogspot.in/2013/01/what-is-constant-memory-in-cuda.html, Accessed on 2.4.2018

CUDA Python

CUDA Python and Numba Library:

- Python - one of the most popular programming languages.
- However, as an interpreted language, it is slow for high-performance computing.
- CUDA Python, using the Numba Python compiler : Targetting both CPUs and NVIDIA GPUs.
- Numba does not yet implement the full CUDA API, so some features are not available.

Dataset

Stock Market Dataset:

For our experimentations, we have chosen a stock market model dataset. The stock market prediction modelling is considered to be a challenging problem.

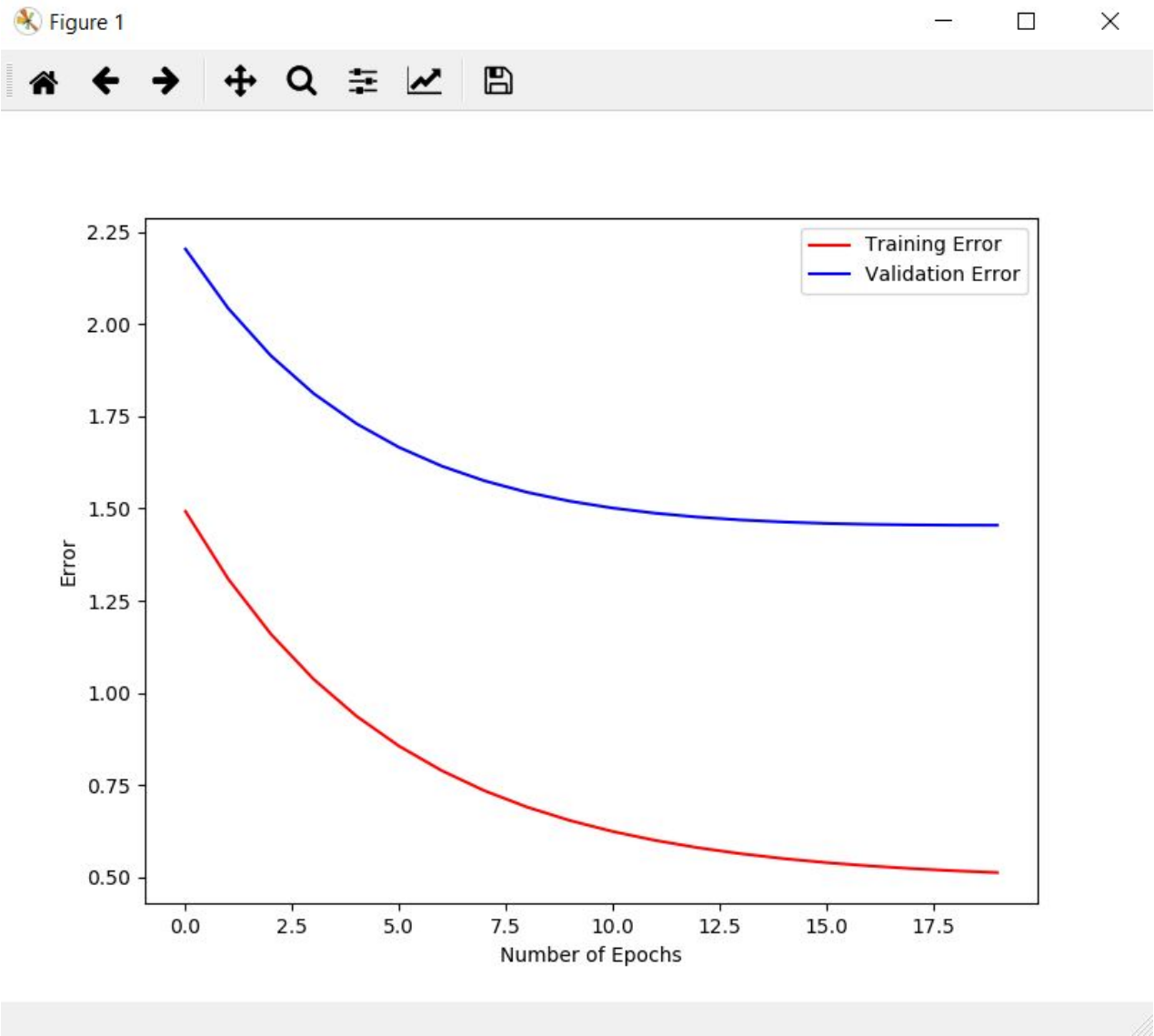
- Stock market price forecasting is one of the challenging tasks due to the difficulty in predicting the non-linear and volatile data.
- Huge volume (about 8lakh entries).
- Small feature set (8 feature entries considered in this dataset).
- Unquantifiable data cannot be included.
- Partially observable Markov processes.

Challenges

The following were the main challenges faced for this project:

- **Debugging** the CUDA interfaced parts.
  1. Accessing variables in GPU memory.
  2. Ordered display of variables.
- **Erroneous functions**
- **Dynamic parallelization:** Parent child launching problem. (In case it was supported, a boost of 20 to 30 times time-efficiency could have been achieved.)

Error Convergence Graph



Training and Validation error plots.

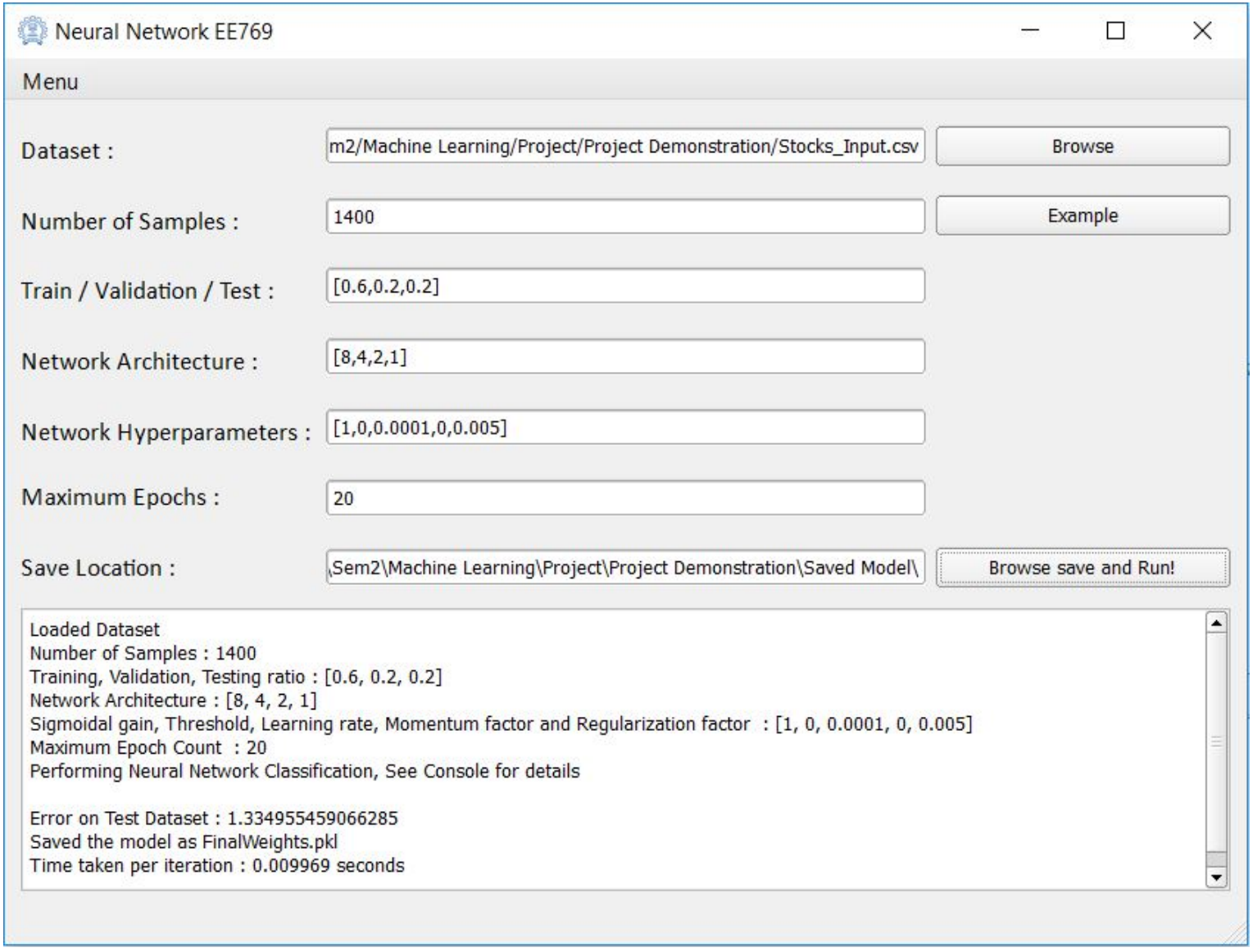
Graphical User Interface (GUI)

To make the model more User-friendly, we developed a GUI on PyQt.

- Hard-coding has been avoided.
- The following fields are required for the user to input.

1. Choosing the dataset.
2. Number of data samples.
3. The training, validation, testing dataset split.
4. Number of neurons in input, hidden(s) and output layer.
5. Sigmoidal gain
6. Threshold value
7. Learning rate
8. Momentum factor
9. Regularization factor

Graphical User Interface (GUI) - Demonstration



Executable File

Created an executable file (.exe) of the complete model, using PyInstaller.

Challenges while making the executable-

- Hidden imports.
- Numba dependencies.
- Platform dependent.

Thank You