Handwritten Digits Recognition Using Neural Network

Aakash Sarang, Sassoun Gostantian

Project Advisor: Prof. Morris E Jones

Electrical Engineering Department, San Jose State University, San Jose, California 95192

Introduction

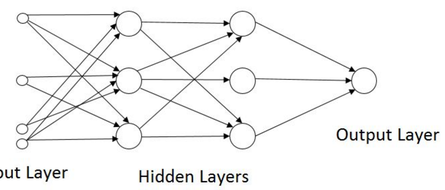
Neural networks are inspired by the brain neurons. It is one of the recently developed technologies already proving its efficiency and importance in the field of image recognition, sound recognition and even biological diagnostics. The parallel processing system of neural networks and its ability to learn from sample inputs makes it suitable for computations which are way more complicated for conventional algorithmic computer systems. Neural networks learn on their own to perform a specific task assigned for the particular network design.

The purpose of this project is to implement a handwritten digit recognition structure using neural network and examine many possibilities to obtain and built the highest accuracy performance by using python and verilog implementations.

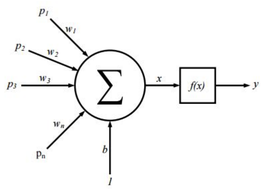
The goal of our work were to study and built different types of neural network systems to predict handwritten images correctly with the best results for the neural network model along with its hardware implementation using FPGA.

Methodology

The figure below shows the data flow structure of the feedforward artificial neural network. As shown in the figure, the network contains four layers of neurons namely input, output and two hidden layers.



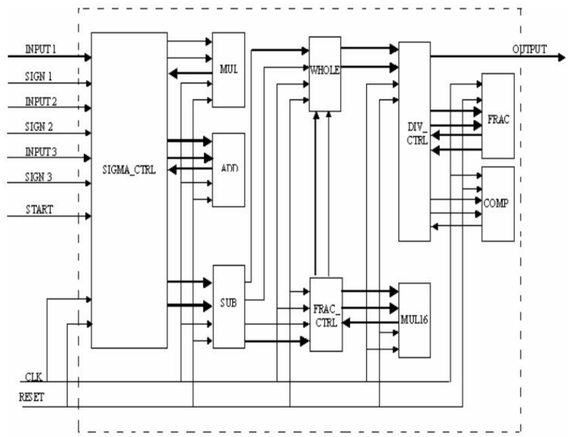
Each neuron is designed as a perceptron as shown in the second figure.The verilog contains basically has three perceptrons as main module.



$$\text{Where, } x = \sum_{i=1}^{n} piwi + b$$

The architecture shown in the next figure has been implemented in verilog for the design of the perceptrons. Three perceptrons are implemented and simulated using quartus ii software.

Methodology



The input images of handwritten digits are converted in the 28x28 pixel format as the design is made according to that size and hence any input images of size greater than that were to be converted and shrinked to the 28x28 pixel size before applying them to the network so that they can be processed easily through computations. The following table shows the possible FPGAs that can be used for implementation of network on hardware.

Hardware:

The possible FPGAs that can be used for implementation:

FPGA	Configurable cells	Other specifications	cost
Virtex-5 SX50T	4080	USB-UART bridge needed	Most expensive
XC3090	1000-7000	144 user i/o pins	Moderate
Altera DE2-70	68416	622 user i/o pins	Comparatively cheaper
Stratix 10	>100000	Low Power with 2x core performance	Expensive
Arria 10	300000	1517 user i/o pins	Moderate

Software:

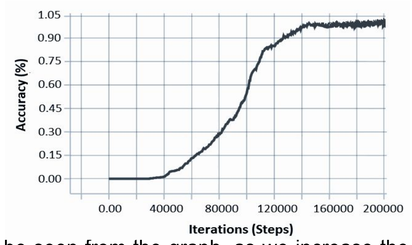
The MNIST dataset is used for training the network model for the handwritten digits recognition. Tensorflow was used for accessing MNIST database and to send it to the python code for neural network for training purpose. The numerical values of weights generated using this training process were then transferred to the verilog code for the perceptron along with the inputs to perform simulations.

Protocol:

The methodology considered in the design is backpropagation algorithm along with parallel processing for the calculation of the neurons simultaneously for a single layer. The layers however work in sequential processing method so as to use the same arithmetic components of hardware for all the layers and hence thereby reducing the requirement of hardware resources. Use of FPGA exploits the advantage of parallel processing for computation of neurons.

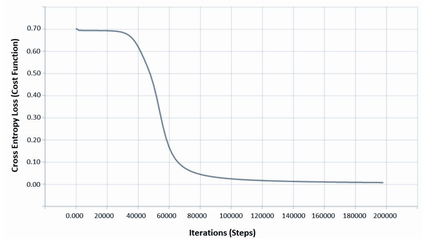
Results

Figure below shows the graph of iterations of training the network v/s accuracy of the CNN network.

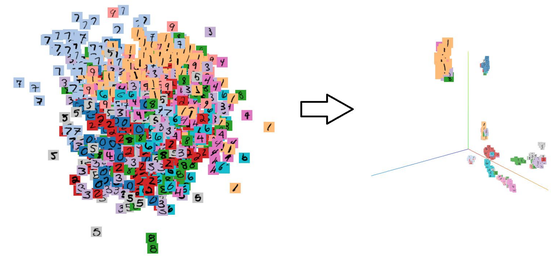


As can be seen from the graph, as we increase the number of iterations in training of the network, the accuracy of the network increases. However, the simulation time increases with increase in number of iterations which makes it impractical to have a very high number of iterations. The extension of graph indicates that we can get up to 98.5 percent accuracy by increasing the number of iterations above 100000 iterations which will take way too long simulation time.

The figure below shows the graph of number of iterations v/s gross entropy.



As can be seen from the graph, the amount of entropy and thereby error in predictions of network decreases with increase in number of iterations. However the window of change in entropy is much narrower than window of change of accuracy with increase in number of iterations. The performance of the network in this project was verifies for 2000 iterations for training as it had sufficient accuracy without compromising simulation times.



The above figure demonstrates how the neural network separates and organizes each digit classes into their own categories. Each of these classes represent embedding process that were learned to classify from fully connected layers while the model was training. Even though we are using only 10 digits classification, the network ends up with more than 10 different classes at the end of final layer which indicates that the network does not have 100 percent efficiency. However, sufficient efficiency was achieved and it was increased furthermore using the convolutional neural network scheme.

Summary

The project successfully deals with the implementation of handwritten digits recognition system using python and the methodology to implement the same on FPGA has been discussed in the project. The comparison of performance of ANN and CNN was performed which proves that higher efficiency of prediction can be achieved using CNN however ANN should be preferred for hardware implementation considering the ease of computation and hardware implementation and satisfactory results with the implementation of perceptron for hardware implementation on FPGA by designing it using Verilog Language.

Key References

[1] Michael A. Nielsen, "Neural Networks and Deep Learning", Determination Press, 2015

[2] UFLDL tutorial by at http://ufldl.stanford.edu/tutorial/

[3] Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[4]Tutorial files available from official tensorflow website at https://www.tensorflow.org/get_started/mnist/mechanics

Acknowledgements

We would like to thank Prof. Morris Jones for his guidance throughout the progress of this project and San Jose State University for providing us resources to carry out the activities related to this project.