License Plate Number Recognition Using FPGA Based Neural Network

Aswathy Poonilavu, SwathiGopala Reddy, and Lili He, *Member, IAENG*

Abstract— **Automatic license plate recognition system has been found convenient in numerous transportation applications. The project aims to develop an FPGA based neural network model that can recognize numbers on vehicle license plates. License plate character recognition becomes challenging when the images have less lighting, or when the number plate is in a broken condition. This would require human intervention for recognizing a character. Hence, a fully automated Number Plate Recognition system needs a better model. In this work, a feedforward neural network that can recognize numbers is implemented on Altera's Cyclone II DE1 FPGA board. The weights required for the system will be obtained by training an NN model on MATLAB using a pool of handwritten digits. The RTL design is synthesized and programmed on FPGA using Quartus II 13.0sp1. The final FPGA model shows an accuracy of 60%.**

***Index Terms*— pattern recognition, FPGA, neural network model**

I. Introduction

Automatic number plate recognition (ANR) is being used in numerous applications including automatic toll systems, parking, traffic law enforcement etc. The need for ANR systems is gearing up with the advent of autonomous vehicles and smart roads. There are about 30 million surveillance cameras on roads that are mostly manually controlled and monitored. Existing ANR systems make use of PC based algorithms to recognize characters from images. A license plate will contain a combination of letters and numbers from the set of 26 English alphabets and 10 decimal numbers. The challenge in developing an accurate fully automated character recognition system originates from the fact that the license plate images obtained for processing could be in different noisy conditions. Neural Network (NN) is one popular approach to address the difficulty in recognition. This project is an attempt to devise a neural network model on a Field Programmable Gate Array (FPGA) that identifies numerical characters on license plate images. [1]

Neural networks have been gaining popularity recently owing to its impressive ability to derive meaning and extract features from a complicated set of data. An Artificial Neural Network (ANN) is a data processing structure inspired by the way a human nervous system works. ANN consists of many processing elements, called the neurons that are inter-connected as shown in figure 1.

The most common structure of a feedforward network where the information propagates forward consists of three layers – input layer, hidden layer, and an output layer. The input layer nodes relay information to the hidden layer nodes. Hidden layer nodes and output layer nodes are active, and hence modify signals according to the weight coefficients in the nodes. ANN learn by reference examples. The neural network is trained for a specific application by literally teaching them through reference patterns. As the network learns, the weight coefficients change in order to compensate for accurate object classification [1].

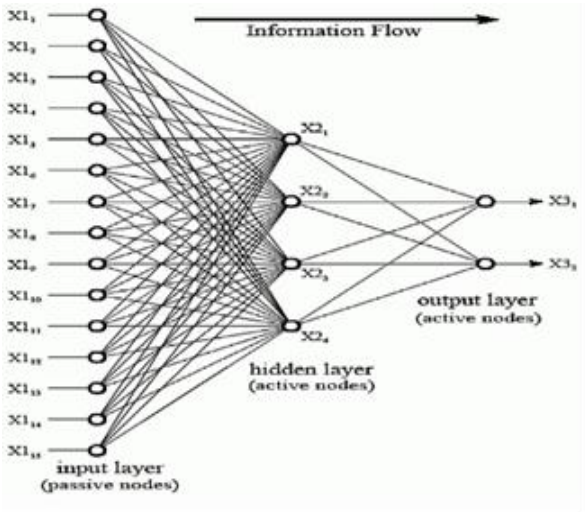

Figure 1. Typical structure of a feedforward neural network [2]. The authors obtained the copyright from the owner.

There are limitations in collecting data samples for the project. Capturing license plate images in the state involves privacy concerns. Even though, the license plate samples available online are plenty, segmenting each of the numbers consumes time and drifts from the scope of the project. Hence, a MATLAB model that trains and recognizes handwritten digits is used to extract the weights associated with hidden and output layers. Since the project focuses

Manuscript received February 22, 2019; revised April 20, 2019.

The Authors are with San Jose State University, Department of Electrical Engineering, San Jose, CA 95192-0084, USA (corresponding author phone: 408-924-4073; fax: 408-924-3925; e-mail: lili.he@sjsu.edu).

mainly on developing an NN (neural network) on hardware, to recognize only numbers and not alphabets, the handwritten digit approach is perceived reasonable.

II. FPGA IMPLEENTATION

The MATLAB simulations on the final reference model verified the feasibility of the network to be implemented on hardware. The basic block diagram of the FPGA implementation is shown in Figure 2. The design interfaces are mentioned in Table1.

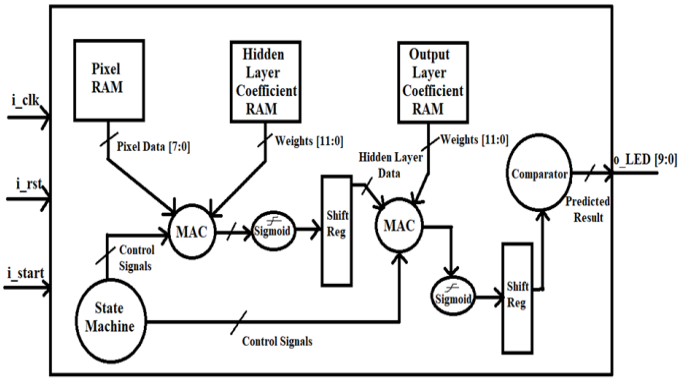

Figure 2. FPGA implementation architecture

A) Interfaces

Table 1. Interface

Interface	IO pins	Function
Inputs	i_clk	The design clock
	i_rst	The reset signal (active high)
	i_start	Trigger signal to initiate the network operations
Output	o_LED[9:0]	Output LED signal to display the recognized digit For example: If the network detects digit 0, o_LED bit [0] is lit

B) Network parameters

The FPGA embedded memory blocks are used to store the input data and the coefficients. Each time, the FPGA is powered on, the RAM blocks get initialized with values fed via mif (Memory Initialization file) files. Cyclone II has 52 M4K RAMs. Input pixels occupy 1M4K, configured as 65x8 bits. Hidden layer and output layer coefficients consume 6 and 1 M4K memory blocks configured as 1300x12 and 210x12 bits respectively. The coefficients for each node are stored one after the other in the RAM locations. In case of hidden layer coefficients' RAM, the first 65 locations contain weights corresponding to node 0 of hidden layer and is followed by node 1 coefficients. The same pattern applies to RAM blocks of the output layer coefficients.

III. METLAB MODEL

The project uses a reference NN on MATLAB for training purpose. Figure 3 contains the original reference model structure.

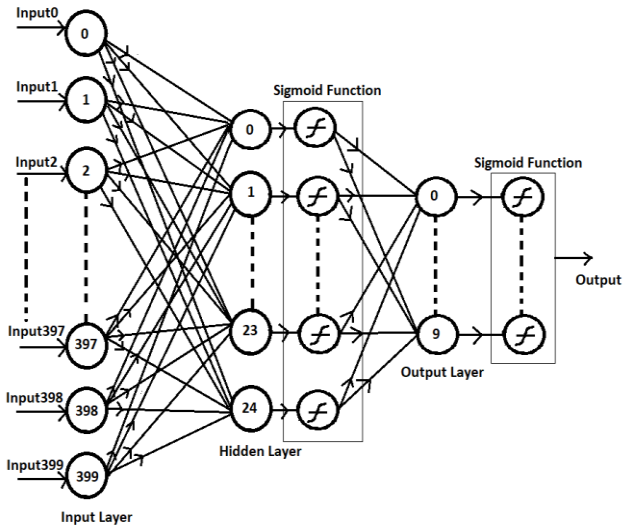

Figure 3. Original reference NN model

The training set includes 5000 images of handwritten digits with a resolution of 20x20 each. The images are converted from color format to grayscale and then normalized to values in the range [-1,1]. All 400 pixels in one sample image is fed via the input layer to the network. Each pixel has 64-bit floating point precision. The network contains only one hidden layer and has a total of 25 nodes. Since there are 10 digits (0-9) to be recognized, the output layer has exactly 10 nodes. Sigmoid function is the choice of activation function for the NN. The activation function limits the output to the range (0,1).

Sigmoid function, $$f(x) = \frac{1}{1+e^{-x}} \quad (1)$$

The hidden and output layer coefficients are initialized with random values represented in double precision floating point format and inputs are forward propagated through the network. The model uses backpropagation algorithm to train and modify the weights of the neural network. When training the network, it is often convenient to have some metric of how good or bad the network is performing. This metric is called cost function and is calculated using the error between the outputs forward propagated from the network and the expected results. These errors are then backward propagated through the network assigning blame for the error. The model uses gradient descent, first order iterative optimization algorithm for finding the minimum of cost function i.e. by

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

taking steps proportional to the negative of the gradient of the function at the current point and updating the weights as they go. The MATLAB model displays an accuracy of 96% after training [3] and [4].

IV. Results and Analysis

A) Simulation results

The design is simulated and tested using Modelsim simulation software. Before testing with license plate digits, the FPGA model is validated using handwritten digits. The bit position with a value of 1 on interface o_LED, indicates the detected digit.

DIGIT 1:

Signal o_LED which is highlighted in blue shows value 1 on the 1st bit, hence it indicates that the detected digit is 1. Nodeout (highlighted in orange) is the shift register containing all values from output nodes. The register is expanded and displayed in Figure 3 for better viewing. Comparators determine the largest value among all 10 values in 'nodeout'. Since nodeout[1] has the highest value in Figure 8 example, the corresponding bit in o_LED is lit indicating the detection of digit 1.

DIGIT 0:

The simulation result for handwritten digit 0 is obtained. Signal o_LED which is highlighted in blue shows value 1 on the 0^{th} bit, which indicates that the detected digit is 0. Nodeout[0] (highlighted in yellow) has the highest value in the given example and hence corresponding bit 0 in o_LED is lit.

DIGIT 6:

The simulation result for handwritten digit 6 is obtained. Signal o_LED which is highlighted in blue shows value 1 on the 6^{th} bit, which indicates that the detected digit is 6. Nodeout[0] (highlighted in yellow) has the highest value in the given example and hence corresponding bit 6 in o_LED is lit. The results manifested a drop in accuracy on FPGA model. FPGA model could establish only 60% accuracy. Ideally, it should have had the same accuracy rate of 86% as in MATLAB model. The drop occurred due to precision loss before feeding the output of MAC to the sigmoid table.

Application of rounding logic resulted in precision loss leading to the same sigmoid output values. The FPGA design performed poorly on license plate characters. The results ended up being a randomly guessed output. The MATLAB model also showcased poor accuracy for license plate character images. he reduction in accuracy of license plate character recognition can be summarized as follows:

i) The actual aspect ratio of a standard license plate image is ~2.3. While training the network, the image resolution has been cut down to 8x8, i.e, aspect ratio of 1 and this deviated from the original features required for network learning.

ii) Reduction in bit width of coefficients and pixels depreciated the accuracy with which the model can learn. The design runs successfully at 50MHz clock rate. It takes 523 clock cycles to detect a digit, i.e 523 * 20ns = 10µs. The Verilog model is much faster than the MATLAB model.

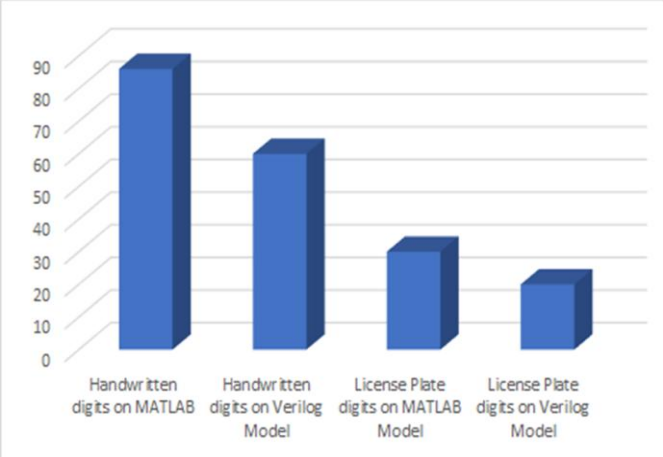

Figure 4. Test results

B) Resource utilization

The hardware model is synthesized and programmed on FPGA using Altera's Quartus II software tool. The resource utilization on Cyclone II DE1 FPGA board is obtained. The number of embedded multipliers used in the design is shown in Table 2.

Table 2. Embedded Multipliers

Modules	Multiplier count
Sigmoid.	6 multipliers (18-bit elements)
Hidden layer MAC	1 multiplier (18-bit elements)
Output Layer MAC	1 multiplier (18-bit elements)

Fitter Summary

Fitter Status	Successful - Sat Nov 24 15:17:35 2018
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	LicensePlate
Top-level Entity Name	nn_top
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	1,261 / 18,752 (7 %)
Total combinational functions	1,261 / 18,752 (7 %)
Dedicated logic registers	465 / 18,752 (2 %)
Total registers	465
Total pins	13 / 315 (4 %)
Total virtual pins	0
Total memory bits	18,640 / 239,616 (8 %)
Embedded Multiplier 9-bit elements	16 / 52 (31 %)
Total PLLs	0 / 4 (0 %)

Figure 5. Resource Utilization on Cyclone II

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The utilization of memory bits and logic elements are minimal owing to reduced parameter precision and pipeline free design.

V. Conclusion

In this project, a FPGA based feed forward neural network has been designed. Due to the lack of license plate characters, a reference MATLAB neural network has been trained using samples of handwritten digits. Handwritten digit recognition worked decently well on FPGA model. Whereas, license plate recognition worked poorly on both FPGA and MATLAB Neural Network models trained using handwritten digits.

The results prove that the accuracy of a neural network is dependent on how well the model is trained using samples that has features closer to the actual objects to be recognized. The precision of parameters also plays a key role in enhancing the recognition accuracy. More the precision, better the detection rate is. When it comes to hardware implementation, precision and accuracy always have a trade-off relation. It requires memory to store network parameters. Memory blocks on hardware are very expensive. Proper analysis is crucial to design an efficient NN model that satisfies both budget and precision demands.

The future work of this project may include the following: The features of license plate digits are different from that of a handwritten digit. The same NN on MATLAB, if trained using license plate images with original aspect ratio of ~2.34 , will guarantee better accuracy results. The design can also be extended to make changes in the precision of inputs to the sigmoid table for more accuracy. Further, the network can be upgraded to train and recognize all 36 characters possible on a license plate image.

References

[1] "Neural Networks", Doc.ic.ac.uk. (2018). Neural Networks. [online] Available at: https://www.doc.ic.ac.uk/~nd//surprise_96/journal/vol4/cs11/report.html[Accessed 26 Nov. 2018].

[2] C. Choo, EE278 Course work, Topic: "Digital Design for DSP and AI/DNN.", Charles Davidson College of Engineering, San Jose State University, San Jose, Sep, 2017. The authors obtained the copyright from the owner.

[3] Hallström, E. (2018). Backpropagation from the beginning. [online] Medium. Available at: https://medium.com/@erikhallstrm/backpropagation-from-the-beginning-77356edf427d [Accessed 3 Dec. 2018].

[4] Donges, N. (2018). Gradient Descent in a Nutshell – Towards Data Science. online] Towards Data Science. Available at: https://towardsdatascience.com/gradient-descent-in-a-nutshell-eaf8c18212f0 [Accessed 3 Dec. 2018].

ISBN: 978-988-14048-6-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)