

DESIGN AND DEVELOPMENT OF DEEP LEARNING  
CONVOLUTIONAL NEURAL NETWORK ON AN FIELD  
PROGRAMMABLE GATE ARRAY

LEE YAN QING

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**DESIGN AND DEVELOPMENT OF DEEP LEARNING  
CONVOLUTIONAL NEURAL NETWORK ON AN FIELD  
PROGRAMMABLE GATE ARRAY**

**LEE YAN QING**

**This report is submitted in partial fulfilment of the requirements  
for the degree of Bachelor of Electronic Engineering with Honours**

**Faculty of Electronic and Computer Engineering  
Universiti Teknikal Malaysia Melaka**

**MAY 2018**

BORANG PENGESAHAN STATUS LAPORAN  
PROJEK SARJANA MUDA II

Tajuk Projek : DESIGN AND DEVELOPMENT OF  
CONVOLUTIONAL NEURAL NETWORK ON  
AN FIELD PROGRAMMABLE GATE ARRAY

Sesi Pengajian : 2017/2018

Saya LEE YAN QING mengaku membenarkan laporan Projek Sarjana Muda ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Laporan adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan laporan ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. Sila tandakan (✓):

**SULIT\***

(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

**TERHAD\***

(Mengandungi maklumat terhad yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan.

**TIDAK TERHAD**

Disahkan oleh:

(TANDATANGAN PENULIS)

(COP DAN TANDATANGAN PENYELIA)

Alamat Tetap: 40, JLN  
PULASAN  
3, TMN KOTA  
MASAI, 81700  
PASIR  
GUDANG,  
JOHOR.

Tarikh : 25 May 2018

Tarikh : 25 May 2018

\*CATATAN: Jika laporan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

## DECLARATION

I declare that this report entitled “DESIGN AND DEVELOPMENT OF DEEP LEARNING CONVOLUTIONAL NEURAL NETWORK ON AN FIELD PROGRAMMABLE GATE ARRAY” is the result of my own work except for quotes as cited in the references.

Signature :

Author : Lee Yan Qing

Date : 25<sup>th</sup> May 2018

## APPROVAL

I hereby declare that I have read this thesis and in my opinion this thesis is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering with Honours.

Signature :

Supervisor Name : Dr Wong Yan Chiew

Date : 25<sup>th</sup> May 2018

## **DEDICATION**

To my parents, Lee Kok Hong and Koo Yin Peng, my supervisor Dr Wong Yan Chiew, and my family and friends.

## ABSTRACT

This work presents design and development of Deep Learning Convolutional Neural Network (CNN) on an Field Programmable Gate Array (FPGA). In recent, CNN is a challenging research area in terms both software and hardware. However, software implementations tend to be prohibitively slow due to more of the neural networks run on sequentially operation architecture. The objective of this work is to develop the deep learning CNN on FPGA since hardware implementations perform parallel computation of each neuron in the layers can be made faster. FPGA are construction of programmable logic, which are not only erasable and flexible for design the realize the algorithm like the software, but also have a great speed to operate some kinds of algorithm due to FPGA has parallel execution ability. This work focuses on handwriting recognition where the machines has the ability to receive and interpret intelligible handwritten input from the sources. Researchers all over the world have achieved successful results in handwritten recognition which can be divided into handwritten numeral recognition, character and cursive word recognition. Neural network is the way people used to realize the pattern classification and image recognition. The design in this work utilize filter which acts as feature detectors from the original input image to extracted and recognized the patterns in images. The speed

and the accuracy of the CNN implemented on an FPGA are analysed. Digits and numbers are successfully recognized by the system.



## ABSTRAK

Kajian ini membentangkan mereka dan membina *Convolutional Neural Network (CNN)* pada *Field Programmable Gate Array (FPGA)*. Pada kebelakangan ini, CNN merupakan kawasan penyelidikan yang mencabar dari segi perisian dan juga perkakasan. Walau bagaimanapun, pelaksanaan perisian cenderung melambatkan perlahan kerana lebih banyak rangkaian saraf berjalan pada senibina operasi secara berurutan. Objektif kajian ini adalah untuk mereka CNN dalam FPGA kerana perkakasan melaksanakan pengiraan selari setiap neuron dalam lapisan boleh dibuat dengan lebih cepat. FPGA merupakan pembinaan logik diprogramkan, bukan sahaja boleh dipadam tetapi juga fleksibel untuk mereka sedaran algoritma seperti perisian, kelajuan untuk mengendalikan beberapa jenis algoritma juga hebat kerana FPGA mempunyai keupayaan pelaksanaan selari. Kajian ini memberi tumpuan kepada pengiktirafan tulisan tangan mesin yang mempunyai keupayaan untuk menerima dan mentafsirkan input tulisan yang boleh difahami dari sumber. Penyelidik di seluruh dunia mencapai keputusan yang berjaya dalam pengenalan tulisan tangan yang boleh dibahagikan kepada pengiktirafan nombor, watak dan juga pengiktirafan perkataan kursif. Rangkaian neural merupakan cara manusia menggunakan untuk menyedari klasifikasi corak dan pengiktirafan imej. Reka bentuk dalam kajian ini menggunakan

penapis yang bertindak sebagai pengesan ciri dari imej input asal untuk diekstrak dan diiktiraf corak dalam imej. Kelajuan dan ketepatan CNN yang dilaksanakan pada FPGA dianalisis. Digit dan nombor Berjaya diiktiraf oleh system.

## ACKNOWLEDGEMENTS

This acknowledgement was written as I completed the work in Universiti Teknikal Malaysia Melaka for the final year project 2018. First and foremost, I would like to sincerely express my gratitude to my supervisor, Dr Wong Yan Chiew who has been guiding me going through this work with patient.

A sincere appreciation I must express to my friend Ibrahim Soliman who always give me guidance in the programming part. He always gives me opportunities to learn and grow stronger.

Next, I also would like to thank lecturers of FKEKK UTeM who had given me advices and comments for this work. Last but not least, my loved ones who has supporting and encouraging me endlessly with enormous love in the entire process. With lots of love and thankfulness, thank you.

## TABLE OF CONTENTS

<b>Declaration</b>	
<b>Approval</b>	
<b>Dedication</b>	
<b>Abstract</b>	<b>i</b>
<b>Abstrak</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>v</b>
<b>Table of Contents</b>	<b>vi</b>
<b>List of Figures</b>	<b>x</b>
<b>List of Tables</b>	<b>xiii</b>
<b>List of Symbols and Abbreviations</b>	<b>xiv</b>
<b>List of Appendices</b>	<b>xv</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>16</b>
1.1 Background	16
1.2 Objectives	18
1.3 Problem Statement	18
1.4 Scope of Work	18

1.5	Significant	19
1.6	Chapter Review	19
<b>CHAPTER 2 BACKGROUND STUDY</b>		<b>21</b>
2.1	Introduction	21
2.2	Convolutional Neural Network (CNN)	21
2.3	Field Programmable Gate Array (FPGA)	23
2.3.1	DE1-SoC	24
2.4	Previous Related Work on CNN and FPGA	25
2.4.1	FPGA Based Real Time Face Recognition System	25
2.4.2	A framework for FPGA-Based Acceleration of Neural Network Inference with Limited Precision via High-Level Synthesis with Streaming Functionality	27
2.4.3	Hardware Design for Convolutional Neural Network as Embedded System	32
2.4.4	XNOR Neural Networks on FPGA	36
2.4.5	A Scalable Deep Learning Accelerator Unit on FPGA	38
2.4.6	Deep Learning Binary Neural Network on an FPGA	40
2.5	Summary	42
<b>CHAPTER 3 METHODOLOGY</b>		<b>45</b>
3.1	Introduction	45
3.2	Flowcharts	45

3.2.1	Operating System and Software Installation	48
3.2.2	Analyse and Enhance the C code	48
3.2.3	Machine Learning and Training	48
3.2.4	Predict	51
3.3	Method Selection	52
3.3.1	FPGA Board Selection	52
3.4	Project Management	54
3.5	Summary	54
<b>CHAPTER 4 RESULTS AND DISCUSSION</b>		<b>55</b>
4.1	Introduction	55
4.2	Interfacing of the CNN and FPGA	56
4.3	Results	63
4.3.1	Accuracy	68
4.3.1.1	Background Color of Testing Images	71
4.3.2	Speed	73
4.4	Summary	76
<b>CHAPTER 5 CONCLUSION AND FUTURE WORKS</b>		<b>78</b>
5.1	Introduction	78
5.2	Conclusion	78
5.3	Future Recommendation	79

<b>REFERENCES</b>	<b>81</b>
<b>APPENDICE A</b>	<b>83</b>
<b>APPENDICE B</b>	<b>85</b>
<b>APPENDICE C</b>	<b>90</b>

## LIST OF FIGURES

Figure 2.1: The architecture of CNN	22
Figure 2.2: The architecture of an FPGA	24
Figure 2.3: DE1-SoC and the package contents	25
Figure 2.4: The overview of complete face recognition system on an FPGA	26
Figure 2.5: The block diagram of face recognition subsystem implemented on Virtex-5 FPGA	27
Figure 2.6: The design of accelerator	29
Figure 2.7: The three abstract layers in the overall system	30
Figure 2.8: The structure of 1-Port ROM	33
Figure 2.9: Initializing a 1-Port ROM	33
Figure 2.10: A MIF file with 25Address Locations	34
Figure 2.11: The architecture to shift and load trained weight values	35
Figure 2.12: The architecture to shift and load input image values	35
Figure 2.13: The setup of hardware	37
Figure 2.14: The hardware implementation of XNOR-net FCC	37
Figure 2.15: The DLAU Accelerator Architecture	39
Figure 2.16: The hardware architecture of the first convolution layer	41
Figure 2.17: The second, third and fourth convolution layer realization	41
Figure 3.1: CNN on FPGA design flow	47



Figure 3.2: The steps of training data	49
Figure 3.3: The process of training data	50
Figure 3.4: The training ended successfully	50
Figure 3.5: The steps of prediction	51
Figure 3.6: The prediction ended successfully	51
Figure 3.7: The DE1-SoC board	53
Figure 3.8: The project planning	54
Figure 4.1: Commands to install python-imaging-library	56
Figure 4.2: Command to change the binary pictures appear as ubyte files	56
Figure 4.3: Coding jpg to MNIST part 1	57
Figure 4.4: Coding jpg to MNIST part 2	58
Figure 4.5: Coding of digit recognition system part 1	59
Figure 4.6: Coding of digit recognition part 2	60
Figure 4.7: The Win32 Disk Imager tool	61
Figure 4.8: Writing the file to the microSD card using Win32 Disk Imager	61
Figure 4.9 Configuring the MODE SELECT (MSEL) switches of the DE1-SoC board	62
Figure 4.10: Determining the Com port of the UART-to-USB connection in Device Manager	63
Figure 4.11: The first training process	64
Figure 4.12: The second training process	64
Figure 4.13: The third training process	65
Figure 4.14: The fourth training process	65
Figure 4.15: The fifth training process and training ended successfully	66

Figure 4.16: Training error against the iteration	67
Figure 4.17: Test error against the iteration	67
Figure 4.18: The first selected testing images in portable network graphics (png) files	69
Figure 4.19: The first results in prediction file in GPU	69
Figure 4.20: The first results in prediction file in FPGA	69
Figure 4.21: The second selected testing images in portable network graphics (png) files	70
Figure 4.22: The second results in prediction file in GPU	70
Figure 4.23: The first results in prediction file in FPGA	70
Figure 4.24: The testing images in portable network graphics (png) files with grey background	71
Figure 4.25: The results in prediction file with grey background images	71
Figure 4.26: The testing images in portable network graphics (png) files with red background	72
Figure 4.27: The results in prediction file with red background images	72
Figure 4.28: Coding to evaluate the speed for processing all images part 1	73
Figure 4.29: Coding to evaluate the speed for processing all images part 2	74
Figure 4.30: The speed for processing all the images in GPU	75
Figure 4.31: The speed for processing all the images in ARM Cortex A9 in DE1-SoC board.	75

## LIST OF TABLES

Table 1 : The resources usage of the final implementation on the Arria V SoC FPGA device	31
Table 2 : The resources usage of the final implementation on the Cylone V SoC FPGA device	31
Table 3: The summary of the literature review.	42

## LIST OF SYMBOLS AND ABBREVIATIONS

For examples:

CNN	:	Convolutional Neural Network
FPGA	:	Field Programmable Gate Array
IC	:	Integrated Circuit
DSP	:	Digital Signal Processing
AI	:	Artificial Intelligence
SoC	:	System on Chip
HPS	:	Hard Processor System
DNN	:	Deep Neural Network
GPU	:	Graphics Processing Unit
HDL	:	Hardware Description Language
ROM	:	Read-Only Memory
FSM	:	Finite State Machine
UART	:	Universal Asynchronous Receiver Transmitter
png	:	portable network graphics (png)

## LIST OF APPENDICES

Appendix A	83
Appendix B	85
Appendix C	90

# CHAPTER 1

## INTRODUCTION

This thesis proposes the design and development of Convolutional Neural Network (CNN) on Field Programmable Gate Arrays (FPGA). This chapter discusses about the project background and motivation, objectives, problem statement and the scope of the project.

### 1.1 Background

Deep Learning CNN is a challenging research region in terms both software and hardware. CNN consists of one or more convolutional layers and then followed by one or more fully connected layers in a multilayer neural network. The architecture of a CNN is designed to take advantages of the two-dimensional structure of an input image with local connections and tied weights followed by some form of pooling which results in translation invariant features. The advantages of CNN also include

they have many parameters compared to the fully connected networks with the same number of hidden units. CNN are easy to train. The feedforward structure of CNN is classified in three layers which are subsampling layer, convolutional layer and the fully connected layer.

Machine recognition, description, classification and image processing are critical problems in variety of scientific disciplines and engineering such as biology, marketing, medicine, psychology, artificial intelligence and computer vision. Handwriting recognition is the capability of the machines that receive and interpret intelligible handwritten input from the sources. Neural network is the way people used to realize the pattern classification and image recognition.

FPGA are the construction of programmable logic, which are not only erasable and flexible for design and realize the algorithm like the software, but also have a great speed to operate some kind of algorithm especially running parallel algorithm due to FPGA has parallel execution ability.

Deep learning CNN on an FPGA can be applied to many applications such as handwritten digits recognition and handwritten documents recognition. It also can be used as facial recognition system on chip which the design methodology can be used to integrated entire components of a target system into single chip so that it can applied to one chip implementation of face recognition for wearable or mobile applications with compact size and weight. Facial recognition on chip for wearable or mobile application can allow users to authenticate themselves by looking at the camera, allowing financial transactions. Besides that, a policeman wears the device around the neck and by accessing a registered database, can automatically check the information of the person in front of him.

## 1.2 Objectives

1. To identify the key parameters of the implementation CNN on FPGA platform.
2. To develop the deep learning CNN on FPGA platform.
3. To analyse the performance of the CNN implemented on an FPGA board.

## 1.3 Problem Statement

Nowadays, software implementations incline to be prohibitively slow due to more of the neural networks run on sequentially operation architectures. Unfortunately, it is impossible to compute or ‘multiple and adds’ or multiple connections synchronously. A fully software implementation is more flexible and easier debugging. However, it usually requires higher hardware capabilities such as a large amount of memory and powerful processor. Therefore, hardware implementations of neural networks are perform, the ‘multiple and adds’ can be performed synchronously and The parallel computation of each neuron in the layers can be made faster. The hardware implementation is characterized by its high speed, lower power consumption and concurrency.

## 1.4 Scope of Work

This work focuses on design and development of Deep Learning CNN on an FPGA. The developed CNN codes will be analysed and enhanced. The simulation of codes will be proceeded in the Linux Ubuntu 16.04 LTS. The DE1-SoC Linux image will be configured to send and receive text. The programs will be written and compiled on the host computer and then transfer the resulting executable onto Linux file system which is microSD. The performance of CNN will also be analysed.