EVOLUTION IN ELECTRICAL AND ELECTRONIC ENGINEERING
e-ISSN: 2756-8458

EEEE

Vol. 6 No. 2 (2025) 231-237
https://publisher.uthm.edu.my/periodicals/index.php/eeee

FPGA Implementation for Character of License Plate Recognition System using CNN

Loong Jun Yong[1], Norfaiza Fuad[1*]

[1] *Faculty of Electrical and Electronic Engineering*
Universiti Tun Hussein Onn Malaysia, 86400Parit Raja, Batu Pahat, Johor, MALAYSIA

*Corresponding Author: norfaiza@uthm.edu.my
DOI: https://doi.org/10.30880/eeee.2025.06.02.028

Article Info

Received: 8 July 2025
Accepted: 3 September 2025
Available online: 30 October 2025

Keywords

Character Recognition, License Plate Recognition, CNN, FPGA

Abstract

License plate recognition, which uses optical character recognition (OCR) techniques to identify vehicle registration numbers, has become an essential component of modern intelligent transportation systems. This research focuses on creating a Character of License Plate Recognition (CLPR) system using a Convolutional Neural Network (CNN) model to address applications in parking management, toll automation, traffic monitoring, and law enforcement. The objective is to create a CNN-based character recognition system and implement it on an FPGA for real-time embedded applications. A dataset of 35,500 grayscale, 28 x 28 pixel pictures from European license plates was used for testing and training. With 94% accuracy, the CNN model was created and trained in Google Colab using Python. Verilog HDL in Intel Quartus Prime 23.1 is used to extract and convert the trained weights and biases into decimal and hexadecimal text files for FPGA implementation. The model is validated through an Excel-based simulation, and floating-point parameters are changed to integers to guarantee compatibility. The Altera DE2-115 FPGA board houses the CNN module, which runs at a clock frequency of 50MHz and requires 40μs for character recognition and 8ms for LCD display. With a power consumption of just 137.24mW, the system is ideal for low-power, real-time embedded systems, according to power analysis.

1. Introduction

Character of License Plate Recognition (CLPR) systems play a vital role in modern society, serving as a crucial technology for various applications that enhance security, efficiency and convenience in vehicle management [1]. The significance of CLPR extends across multiple areas, including traffic management, law enforcement, automated toll collection and parking management making it an indispensable tool in contemporary urban infrastructure [2]. In this project, Character of License Plate Recognition (CLPR) systems be constructed by using the Convolution Neural Network (CNN) technique and Field Programmable Gate (FPGA). CNN is a one of the network structures designed in deep learning algorithm. This type of network structure is suitable for image recognition and the task include processing and analysis the pixel data of the image. Convolutional Neural Network (CNN) consists of multiple layers like the input layer, convolutional layer, pooling layer and fully connected layers [3]. Angle, luminance, and resolution all have an impact on CNN's picture classification ability. High-resolution photos aid in the extraction of fine characteristics, different angles enhance generalization, and appropriate lighting lowers noise. Accuracy and robustness are improved through training with a variety of high-quality images [4]. CLPR system designed by using CNN technique, implement and verify on FPGA hardware board.

2. System Design

2.1 Architecture of CNN Model

The training and testing datasets are preprocessed through resizing, grayscale conversion, tensor transformation, and normalization. The CNN model comprises two convolutional layers, two max-pooling layers with ReLu activation, and a fully connected layer. The architecture of the CNN model is illustrated in Fig. 1, while Table 1 summarizes the input and output shapes and the number of filters used in the convolution and max-pooling layers.

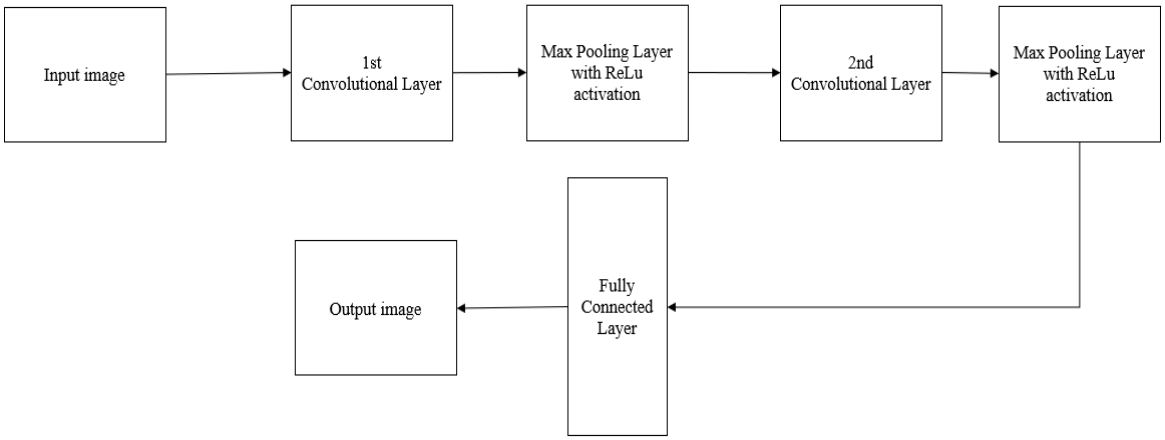

Fig. 1 *Architecture of CNN Model*

Table 1 *Shape of Each Layer of CNN Model*

Layer	No of Filter/Pool	Filter Shape	Pool Shape	Input Shape	Output Shape
Convolution Layer 1	3	(5, 5)	x	(28, 28, 1)	(24, 24, 3)
Max-pooling Layer 1 with ReLu activation	1	x	(2, 2)	(24, 24, 3)	(12, 12, 3)
Convolution Layer 2	3	(5, 5)	x	(12, 12, 3)	(8, 8, 3)
Max-pooling Layer 2 with ReLu activation	1	x	(2, 2)	(8, 8, 3)	(4, 4, 3)
Flatten	x	x	x	(4, 4, 3)	(48, 1)
Fully connected Layer	x	x	x	(48, 1)	(35, 1)

2.2 Design Flow

Fig. 1 shows the design flow of CLPR system. In phase 1, the CNN model was written by using Python code and trained in Google Colab. In phase 2, the CNN model in Excel is designed and configured using floating-point parameters according to the structure of the CNN model built in Google Colab. Then, the parameters of the CNN model times 128 and rounding become integer numbers. The result of the simulation of the fully connected layer of Verilog HDL is the same as the result in Excel, which can verify that the Verilog HDL code is absolutely correct. The MSB is used to reduce the complexity of calculation.

The flowchart of Verilog HDL code functional verification is illustrated in Fig. 3. If the functionality of the CLPR system is passed, then it can be downloaded to the FPGA board, and the functionality of the CLPR system on the FPGA board in phase 3. Then, the input images convert to hexadecimal memories, initialize the file as input of FPGA hardware, and the recognition result will show on the LCD. The CLPR system implemented on the FPGA DE2 115 board [5] would be completed after its functionality is achieved. However, its functionality cannot be achieved; it needs to be debugged in the Verilog HDL module.

Evolution in Electrical and Electronic Engineering Vol. 6 No. 2 (2025) p. 231-237

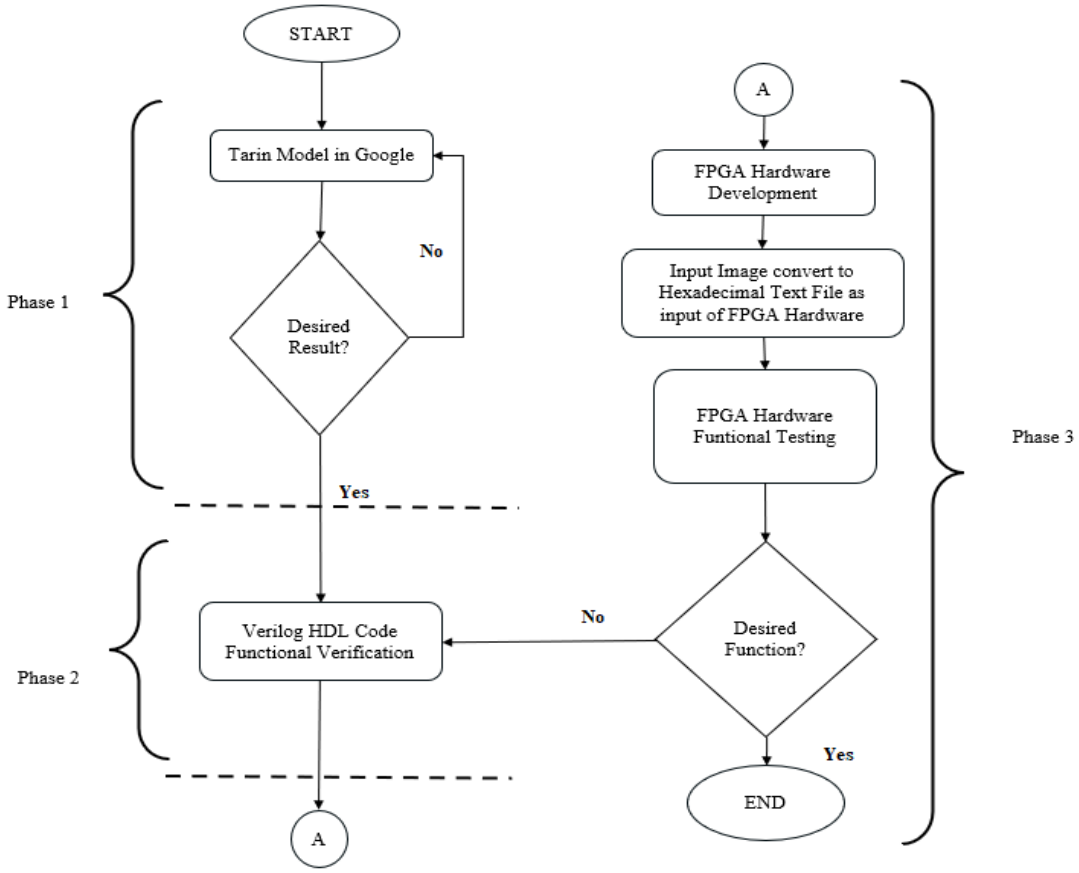

Fig. 2 *Design flow of CLPR system*

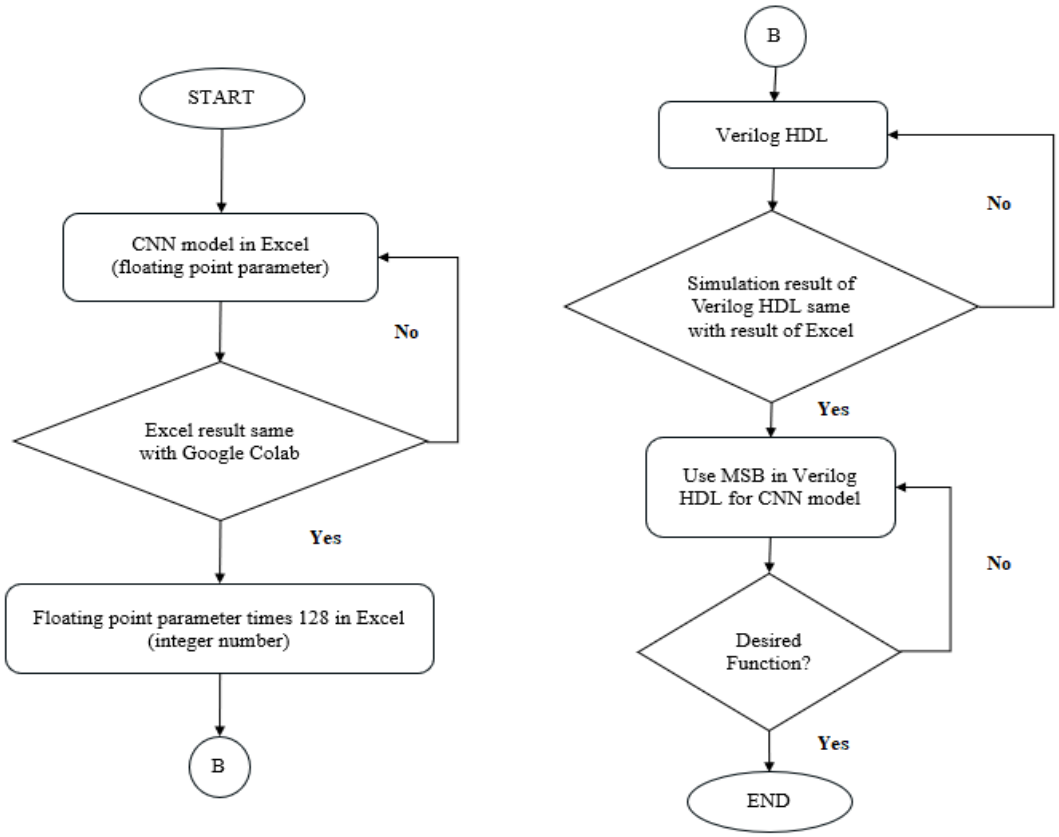

Fig. 3 *Verilog HDL Code Functional Verification*

2.3 Dataset for Training and Testing

The CLPR database for the training and testing process is prepared. This dataset is taken from Kaggle, which consists of 35.5k images about the black character and number of European license plates. All the images are separate, 28x28 pixels in grayscale, and have been categorized into 35 classes. The dataset from Kaggle, according to the Europe license plate rule, would not consist of the character "O" since it is similar to the number "0". The input image would transform to 28x28 pixels and be normalized for training in Google Colab. However, the input images would be extracted to a hexadecimal text file for the Verilog HDL module in phases 2 and 3, since the Verilog HDL module can read number systems only.

3 Results and Discussion

The accuracy of the trained CNN model achieves 94%, while its loss is 0.2474 in Google Colab. The RTL design of the CNN model is shown in Fig 4.

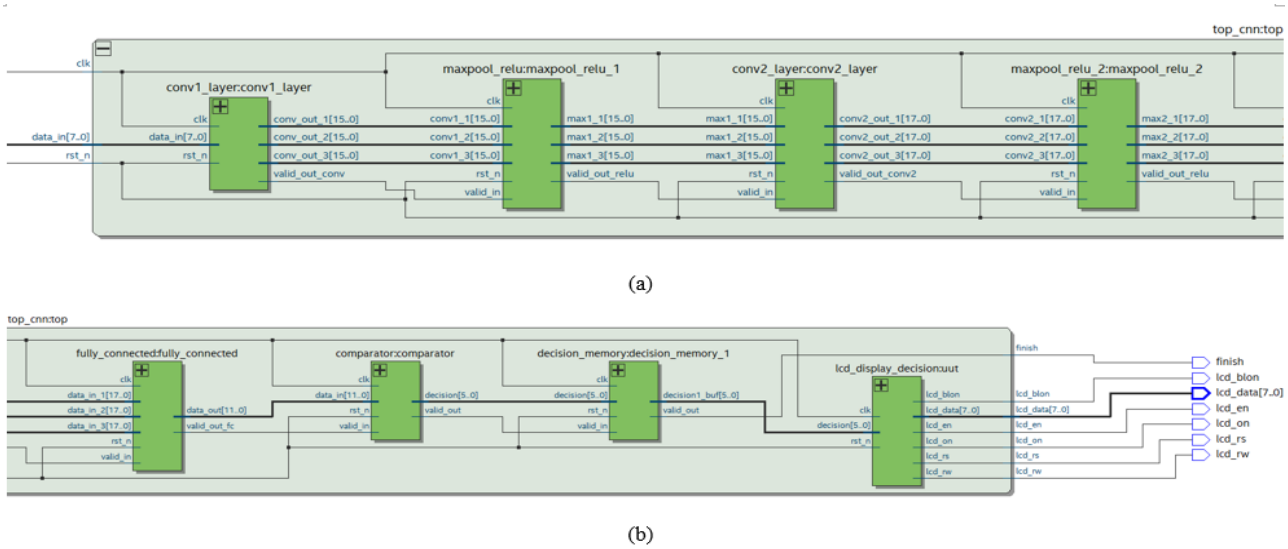

Fig. 4 *The RTL design of CNN model in Verilog HDL (a) First part (b) Continued design from the first part*

Then, the image of the number "1" as an input example after preprocessing becomes the input of the CNN model with floating-point parameters for Google Colab and Excel to verify the correctness of the CNN model in Excel. The results of fully connected layers are shown in Fig. 5 and Fig. 6.

```
fc1 output:
[[-10316.807     5555.915     1834.6586
    2532.3792    1717.4169    3698.634
    4584.7437   -3352.6023   -4106.4385
    1616.0502   -1256.4027    -665.79755
    3623.615     -621.09515    247.46165
   -5675.359     1388.6333     518.0407
    3411.93     -4430.586     3157.0508
Predicted class: 1
```

Fig. 5 *The highest value is number "1" is detected in Google Colab*

0	1	2
-10316.8076	5555.9141	1834.6582

Fig. 6 *The highest value is number "1" highlighted with green in Excel*

Evolution in Electrical and Electronic Engineering Vol. 6 No. 2 (2025) p. 231-237

Next, the image of character "Z" as an input example after preprocessing becomes the input of the CNN model with integer parameters in Excel and Verilog HDL, which results in fully connected layers that are the same in the CNN model to verify the correctness of the CNN model and are shown in Fig. 7 and Fig. 8.

29	30	31	32	33	34
U	V	W	X	Y	Z
-5137212870	-4888401538	-11658549306	-7104652625	-939474049	13752162269
FECDCC5A3A	FEDCA0E97E	FD4918A7C6	FE58879AAF	FFC800C37F	333B157DD
-14	-32	11	6	-19	33
24	15	45	-87	-38	-6

Fig. 7 *The highest value is character "Z" highlighted with green in Excel*

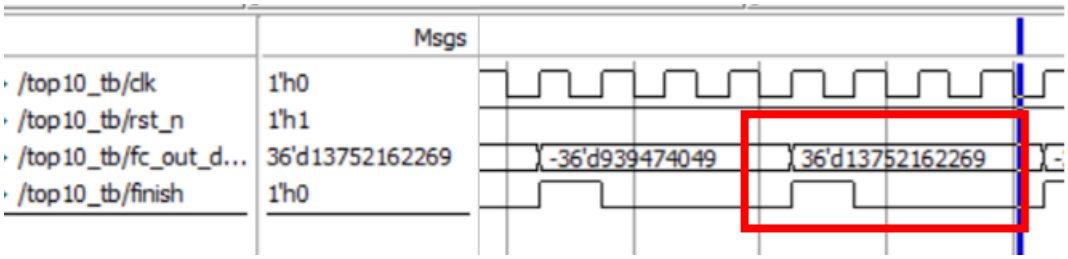

Fig. 8 *Output result in Excel is the same as the CNN model in Verilog HDL*

Reducing power consumption in a Verilog HDL CNN model is achieved by limiting each layer's computation to its most significant bit (MSB). This minimizes bit-width usage, reducing switching activity and logic utilization. The dynamic power is lowered, making the design more efficient for FPGA implementation without sacrificing model accuracy. Then, the most significant bit (MSB) used for each layer in the Verilog HDL CNN model was listed in Table 2. After that, the function of the CNN model in Verilog HDL is to remain and complete implementation on the FPGA board, which is shown in Fig. 9.

Table 2 *MSB taken from each layer*

Output Data Bits for Each Layer	Data Bits
Convolution Layer 1	16
Maximum Pooling Layer 1	16
Convolution Layer 2	18
Maximum Pooling Layer 2	18
Fully Connected Layer	12
Comparator	6

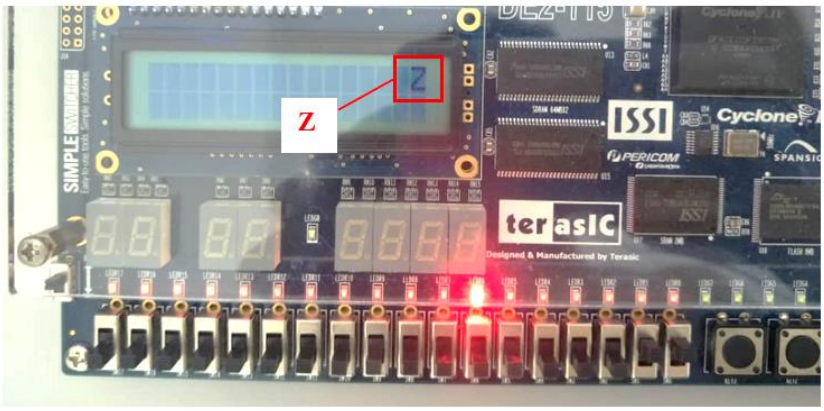

Fig. 9 *Character "Z" is detected on the FPGA board*

In Fig. 10, the number "4" which was taken from a Malaysian license plate, is shown. It was converted to hexadecimal format and inverted from black and white and is shown in Fig. 11. Then, the number "4" is detected and displayed on the LCD of the FPGA, which is shown in Fig. 12.

Fig. 10 *Number "4" from Malaysia License Plate*

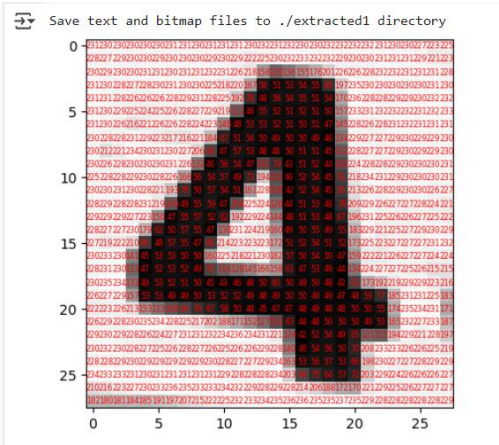

Fig. 11 *Result after inverting colour*

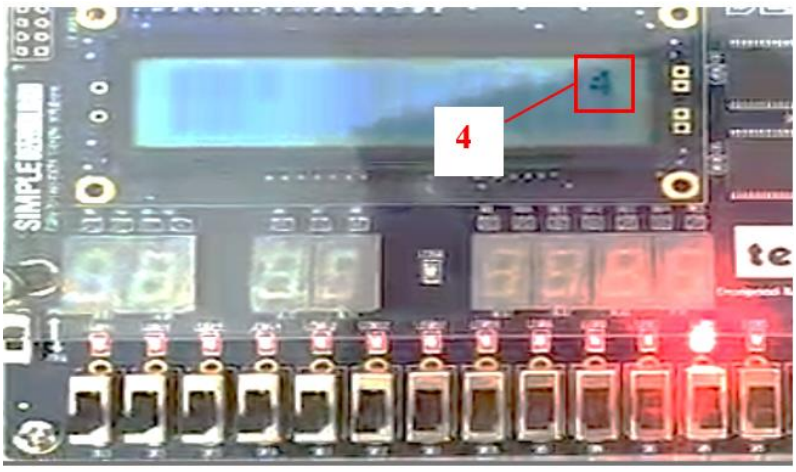

Fig. 12 *Number "4" is detected on FPGA board*

Table 3 summarizes the resource utilization, time for character detection and result display on LCD, and total consumption. With a 50MHz clock signal, the power consumption is 137.24mW, and from the perspective of resource utilization, the complete design of the CNN model occupies 55,739 logic elements, 16 I/O pins, and 8,192 block memory. The time for character detection and LCD display is 40us and 8ms separately.

Table 3 *Resource utilization, time detection and total power consumption*

Resource Utilization	Logic Elements	55,739 / 114,480 (49%)
	I/O Pins	16 / 529 (3%)
	Block Memory	8,192 / 3,918,312 (<1%)
Character Detection Time	40us	
LCD Display Time	8ms	
Power Consumption	137.24mW	

4 Conclusion

In conclusion, the Character License Plate Recognition (CLPR) system using a Convolutional Neural Network (CNN) was successfully trained using Google Colab and implemented on a Field Programmable Gate Array (FPGA) hardware board. The CNN model was trained using European license plate characters. Malaysian license plate characters can also be used for detection, but their black and white colors must be inverted before being used as model input. The character detection function of the CNN model has been validated, demonstrating efficient performance with high detection speed and low power consumption.

Acknowledgement

The author would like to thank the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussien Onn Malaysia for support.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Loong Jun Yong, Norfaiza Fuad;* ***data collection:*** *Loong Jun Yong;* ***analysis and interpretation of results:*** *Loong Jun Yong, Norfaiza Fuad;* ***draft manuscript preparation:*** *Loong Jun Yong. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] N. Das, U. Mahbub, and S. M. Mahbubur Rahman, "Automatic number plate recognition (ANPR) in smart cities," Cities, vol. 131, p. 103930, 2022, doi: 10.1016/j.cities.2022.103930.

[2] Shulman, L., Averbuch, A., & Tal, Y. (2006). Towards a Multinational Car License Plate Recognition System. Machine Vision and Applications, 17, 173–183.

[3] IBM, "Convolutional Neural Networks," Oct. 6, 2021. [Online]. Available: https://www.ibm.com/think/topics/convolutional-neural-networks

[4] Z. E. Vargoorani, "License Plate Detection and Character Recognition using Deep Learning and Font Evaluation," Doctoral dissertation, Concordia University, 2024.

[5] Terasic Technologies. (2013). DE2-115 user manual (Rev. 1.6). Available https://www.terasic.com.tw/cgi-https://www.terasic.com.tw/attachment/archive/502/DE2_115_User_manual.pdf