FPGA-Based Implementation of Convolutional Neural Networks for Enhanced Physical Security in Data Center Door Access Systems

Bouchra Kouach
Laboratory of Advanced Systems Engineering. ENSA, Ibn Tofail University, Kenitra, Morocco
bouchra.kouach@uit.ac.ma (corresponding author)

Mohcin Mekhfioui
Green Tech Institute (GTI), Mohammed VI Polytechnic University, Benguerir, Morocco
mohcin.mekhfioui@uit.ac.ma

Azzedine El Mrabet
Laboratory of Advanced Systems Engineering, ENSA, Ibn Tofail University, Kenitra, Morocco
azzedine.elmrabet@uit.ac.ma

Rachid El Gouri
Laboratory of Advanced Systems Engineering, ENSA, Ibn Tofail University, Kenitra, Morocco
rachid.elgouri@uit.ac.ma

Received: 27 January 2025 | Revised: 17 March 2025, 3 May 2025, and 11 May 2025 | Accepted: 15 May 2025

 DOI: https://doi.org/10.48084/etasr.10352

ABSTRACT

Digital data security has become a critical topic, whereas the physical security of the hardware on which digital data is stored remains equally critical. The loss of a server is equivalent to the loss of data. Access to the data center rooms is provided using badges assigned to each individual. However, these badges are at risk of being lost or stolen, which can lead to unauthorized access. This represents a risk of the loss of physical equipment and, consequently, of the loss of confidential data. This study proposes an intelligent security system for access doors in data center rooms to ensure optimal security measures, based on CNN image classification applied on a Xilinx Zynq FPGA board using the PYNQ framework. Python was used to develop and train CNN models for image classification, exploiting frameworks such as TensorFlow and Keras. The results demonstrate that Deep Learning (DL) models can be applied on a ZYNQ FPGA board to optimize inference time and highlight Faster R-CNN as the most effective model for image classification, contributing to strengthening the physical security of building access.

Keywords-DL; CNN; FPGA; Xilinx ZYNQ; PetaLinux; PYNQ; Tensorflow; Keras; face classification; Python; door security

I. INTRODUCTION

Today, with the increasing presence of digital technology, data security has become a major issue for companies and individuals everywhere. Advanced security mechanisms are used to protect confidential information against cyberattacks. In the context of continuous advances in cybersecurity measures, an often overlooked but equally crucial aspect is the physical protection of the equipment that contains these precious data. Consider a server that stores years of corporate financial data or sensitive customer information. If this server is physically compromised, the consequences could be catastrophic. Not only could the loss of data be irreversible, but it could also result in considerable financial damage, loss of reputation, and even legal consequences. In this sense, strengthening physical security represents a major challenge. Facial recognition represents an innovative technology for physical security.

Numerous studies have attempted to develop facial recognition technology in projects other than physical security. In [1], the implementation of CNN on a ZYNQ board was presented for object detection using Python. In [2], a system was developed for face detection, facial expressions, and age and gender classification based on a CNN, using an image, a video, or in real-time. In [3], the objective was to detect fake faces using the capability of Generative Adversarial Networks (GANs). In [4], a transformer-based model was used for robust face detection of infants and children in hospitals using RGB

and thermal images. The Detection Transformer (DETR) was pre-trained on the WiderFace dataset to detect faces in USIP, and thermal images were also used to improve the accuracy of face detection. In [5], a CNN was proposed for facial recognition detection to improve precision and processing time. In [6], a comprehensive approach to face occlusion recognition was proposed based on a two-stage CNN. However, these studies did not include the real-time decision-making stage after facial recognition. This is why it is crucial to integrate FGPAs to perform actions in real-time.

Several studies have attempted to use FPGAs to exploit Deep Learning (DL) algorithms. In [7], a DL accelerator was deployed on an FPGA using NVIDIA's Deep Learning Accelerator (NVDLA). In [8], a parallel implementation of a CNN on a multi-FPGA cluster was proposed, developing a PYNQ cluster consisting of cost-efficient Zynq boards, called M-KUBOS. In [9], fast CNNs were implemented on FPGA using HLS4ML. In [10], the main objective was to improve and optimize the performance of asynchronous machine control using the implementation of artificial neural networks on FPGAs. However, these works did not address the use of FPGAs in facial recognition. Some other works tried to use FPGA boards with DL models for facial recognition and image classification. In [11], an innovative recognition system was presented, which integrated GAN and CNN on an FPGA board for a multistage image recovery algorithm, but this study did not use the PetaLinux operating system on the FPGA board. In [12], the FPGA ZC702 SoC board was used to evaluate the performance of the LBP descriptor used for facial feature extraction, using the PetaLinux operating system, but this study did not provide a real hardware implementation. Previous studies have implemented DL models on FPGA boards for various purposes. However, none has fully leveraged the combined advantages of artificial intelligence and the high computational speed of FPGAs to improve intelligent security systems. Thus, there is a literature gap on the exploitation of artificial intelligence, precisely DL, in facial recognition with real-time decision-making by exploiting the potential of FPGAs. In this context, this work proposes the implementation of a system based on image classification by a CNN deployed on an FPGA board to increase the physical security of access doors to data center rooms through real-time face classification. This approach allows the implementation of an intelligent security system that combines artificial intelligence and programmable hardware to offer a robust and effective security solution that provides optimal protection to prevent unauthorized access to entry points of sensitive IT infrastructures. DL frameworks, such as TensorFlow and Keras, and code execution in Google Colab, a cloud service based on Jupyter Notebook, were used with Python. These tools offer a high degree of flexibility and convenience, enabling the design and training of advanced architectures on complex datasets. Python was also used to develop CNN models, as this language, identified as the most popular in 2018 [1], is well suited to this task due to the availability of open-source frameworks.

This study used an FPGA and a GPU to provide the required acceleration, enabling the scaling of current models beyond the limits of contemporary data and model size. However, they depend on High-Level Synthesis (HLS) design tools, which presents a challenge in the deployment of sophisticated algorithms such as face detectors in embedded systems. This issue can be addressed by combining software and hardware design, for instance, by using Python for FPGA design and the Petalinux operating system, an embedded Linux distribution specially designed for systems based on Xilinx ARM processors. PetaLinux simplifies the process of developing and deploying embedded software, providing a reliable platform for running a CNN model in real-time in an FPGA environment, especially the ZNQY board with the ZYNQ framework that is compatible with Python.

The results highlight the significant advantages of implementing a CNN on FPGAs compared to traditional approaches using CPUs or GPUs. First, in terms of energy consumption, FPGAs have greater efficiency than CPUs and GPUs. Implementing a CNN on an FPGA reduces energy consumption compared to traditional architectures, which is particularly important in applications where energy consumption is critical. Second, an FPGA is perfectly suited to the parallel nature of CNN calculations, making it an ideal solution for real-time applications. Unlike CPUs and GPUs that are optimized for limited sequential or parallel computing tasks, FPGAs offer massive parallelization of computations, enabling significant acceleration of data processing. This translates into superior performance and faster response times, which are essential in applications where decisions need to be made in real-time, such as data center security monitoring. The proposed system aims to maximize accuracy while optimizing the performance of the CNN. Moreover, it is easily deployable and based on open-source technologies, providing an accessible and adaptable solution for different security environments.

II. METHODOLOGY

This study focuses on the deployment of a CNN for real-time face detection and access control for server room doors. The proposed system is an open-source embedded solution that can be easily integrated into any door. It also enables adding or blocking access for certain individuals, all managed remotely through a web application. The proposed solution is energy-efficient and includes several stages and subsystems:

- Camera block: This block is used to detect faces and send images to the FPGA board.
- FPGA and CNN block: This block is used to classify faces and decide if the door will be opened or not.
- LED and door control block: The LEDs light up green if the person has access, and red otherwise. The door control takes the action of opening or not opening the door.
- Supervision and control block: The system is connected to a mobile application to send a notification to the system administrator when someone tries to access the room. This application can increase system security.

A. Real-Time Detection

The detection of human faces consists of detecting the presence, location, and size of a human face in a two-

dimensional image, based on criteria such as the detection of facial features without considering other objects. The detection of faces in real-time is a difficult problem because the response of the system must be obtained within a given time. The performance measures of a real-time system are the system's latency, throughput, and bandwidth. For this reason, this study presents the integration of a CNN with an FPGA board to take advantage of the system's parallel computing capabilities. Although effective for image detection and recognition, CNNs are limited by their computational complexity, requiring long processing times on conventional processors. Using an FPGA board, convolution operations can be parallelized, optimizing image processing by distributing calculations over several logic units. This not only improves the execution speed of face detection but also reduces the overall system latency while increasing detection accuracy.

B. Convolutional Neural Networks (CNNs)

A CNN is a DL algorithm composed of neurons that have adjustable weights and biases. Each neuron conducts a mathematical operation, called a dot product, between its input and weights, which is then followed by a nonlinear activation function. CNNs are particularly adept at tasks such as image classification, natural language processing, and recommendation systems because of their ability to learn complex patterns. Furthermore, CNNs can distill the information from raw input images into a single differentiable score, which is achieved through a series of layered operations designed to capture and process relevant features. The convolutional layer is critical to the operation of CNNs. This layer plays a vital role in extracting meaningful features from input data, allowing the network to learn and understand complex visual patterns [13]. Thus, the convolutional layer is simply a convolution of the image of the previous layer, where the weights determine the convolution filter. The input image is given by $I(i,j)$, where each pixel is considered a scalar, the filter is represented as a kernel $K(n,m)$, and the convolved output is given by $h(i,j)$:

$$h(i;j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, i-n)K(m,n) \quad (1)$$

- Pooling Layer: The main objective of the pooling layer is to reduce the dimensions, the number of parameters, and the complexity of the model. The most commonly used pooling techniques are max pooling and average pooling [13].
- Max Pool: This operation is the most commonly used type of pooling and returns the maximum value within the receptive field.
- Min Pool: In this operation the minimum value within the pooling window is selected.
- Average Pool: This function returns the average value within the receptive field.
- Fully Connected (FC) Layer: FC layers form an integral component of CNNs and play a crucial role in object detection. They sit in the back of CNN structures and oversee learning intricate feature combinations and capturing global relationships. As a result, any node in an FC layer is directly connected to every node in the layer above and below it [13].
- Activation Layer: The activation layers are the most integral part that adds nonlinearity to the system, allowing the network to learn highly complex relationships between the feature maps. An activation function determines the output of the model, its prediction accuracy, and computational efficiency during training. These functionalities are realized through dedicated hardware logic in the hardware implementation. Activation functions are chosen based on the statement of the problem.

C. Tensorflow API

TensorFlow is an open-source library dedicated to numerical computation, machine learning, DL, and other statistical processing loads. TensorFlow requires a powerful 64-bit OS for its deployment and is thus implemented on Xilinx Zynq ZCU104 FPGA that can support both 32- and 64-bit operating modes [1]. As shown in Table I, TensorFlow is a good practice for implementing CNNs on FPGAs.

III. ZYNQ HLS: FPGA DESIGN APPROACH

A. ZYNQ SoC

Xilinx, a leader in FPGA technology, has unveiled a new type of device: the Zynq FPGA. Figure 1 presents the high-level model of the Zynq architecture. Zynq comprises two principal parts: a Processing System (PS) built around a dual-core ARM Cortex-A9 processor and Programmable Logic (PL). The connection between the PL and the PS is multiple and programmable, allowing rapid data communication. In contrast, in previous FPGA technologies, the separation between the processor and the PL limited these connections and the speed of data transmission.

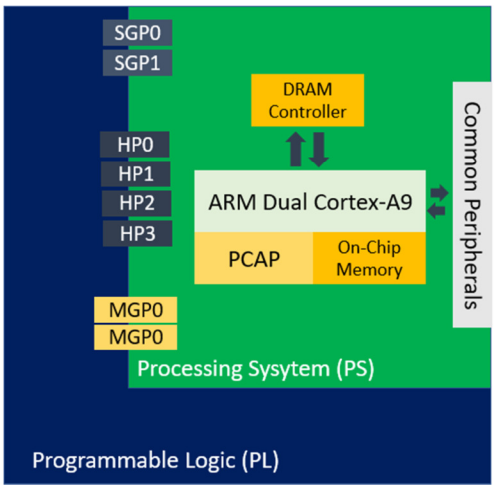

Fig. 1. The architecture of the ZYNQ FPGA board.

TABLE I. ZYNQ 7020 BOARD FEATURES

Device	Logic cells	Look-up tables	Flip-flops	DSP slices	RAM
Zynq 7020	85k	53,200	106,400	220 programmable	560 KB

B. FPGA Design Tools

To develop FPGA applications, Xilinx provides software tools with various functionalities. These tools allow the exploitation of the platform's peripherals and their programming to achieve the desired operation for the application [14]:

- Vivado is an Integrated Development Environment (IDE) for designing the hardware architecture of a system, including the processor, memory, peripherals, external devices, and bus connections [14].
- The SDK is a development environment. C and C++ are used to program the software component of the system. The software supports all IPs provided by Xilinx. The SDK provides tools for compiling, debugging, and running the software application [14].
- High-Level Synthesis (HLS) is a rapid prototyping tool. This tool enables the conversion of C, C++, or System C code into Register Transfer Level (RTL) code. The resulting code can be synthesized and implemented on a Xilinx FPGA or Zynq device (Figure 2) [14].

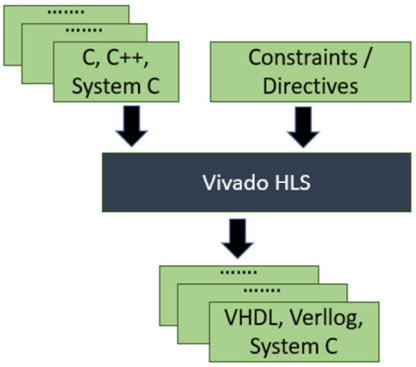

Fig. 2. Vivado HLS tool.

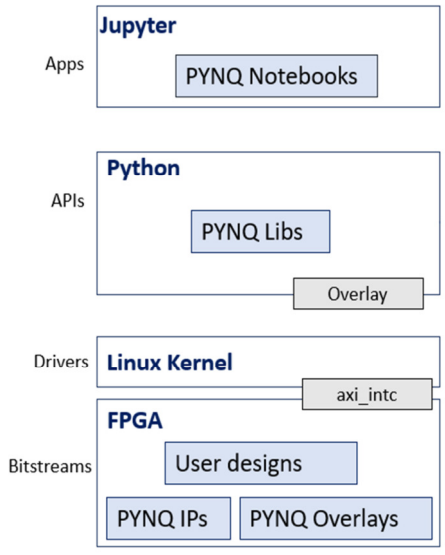

Fig. 3. A simplified overview of PYNQR architecture.

The development environment utilizes hardware libraries or Overlays to create a virtual programmable space on top of the low-level FPGA fabric, without requiring RTL development. Figure 3 provides a simplified overview of the PYNQ development architecture.

IV. EXPERIMENTAL SETUP

A. Implementation Method

The proposed application utilizes a CNN on the ZNYQ FPGA board using the PYNQ framework. The process involves the detection of faces in images captured by a door-mounted camera. Cameras will be installed near access doors to data center rooms to capture real-time images of people attempting to enter. The captured images will be sent to the ZYNQ board with the CNN, which would have been trained to recognize the faces of authorized employees. The CNN will analyze the images to detect and identify people arriving at the doors. Depending on the recognition result, the system will decide on access authorization. If a person is recognized as authorized, the door will be unlocked. If not, an alarm signal will be triggered and appropriate action will be taken, such as recording the incident or sending alerts to security officers. The tools that will be employed include the ZNYQ card with the PYNQ framework, a camera, a motor, the PetaLinux operating system, and Python using TensorFlow and Keras frameworks validated with a Jupyter notebook deployed on the FPGA (Figure 4).

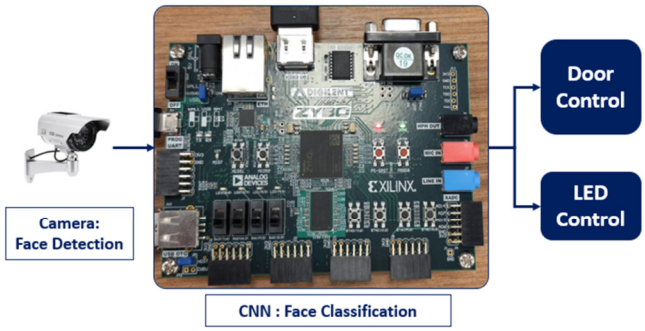

Fig. 4. Implementation of the experiment.

The implementation of a CNN on the ZYBO Zynq-7000 board follows several key steps. First, the PYNQ OS image is flashed onto an SD card, which is then inserted into the board. The board is connected to a display via an HDMI cable, with a keyboard and mouse also connected for direct interaction with the PYNQ environment. Once the board is switched on, the PYNQ interface is accessed to configure and execute tasks. The CNN model is designed and trained on a host using Keras and TensorFlow frameworks and then converted to a format suitable for FPGA execution. A customized hardware block is developed with Vivado HLS to accelerate convolution operations, optimizing calculations for FPGA hardware. This block is integrated into the hardware through Vivado, generating a bitstream file loaded onto the ZYBO board. The interface with the FPGA hardware is realized using the PYNQ framework, enabling data transfer and controlling the execution of the CNN model to infer on access images. This approach

takes advantage of the FPGA's hardware acceleration capabilities using Python for easy and flexible process management.

B. Datasets

A dataset used was obtained from Kaggle [15] and supplemented with personal images of the author. The dataset contains six classes, distributed as follows:

- Five classes from the Kaggle dataset represent celebrities (Chris Evans, Chris Hemsworth, Mark Ruffalo, Robert Downey Jr., and Scarlett Johansson).
- A sixth class comprising personal images of the author was used for real-time testing.

Separating the data into six classes allows the model to learn to recognize different people, which is essential for identifying authorized and unauthorized users. The dataset contains more than 400 images, divided into three sets:

- Training set (60%), used to adjust model parameters.
- Validation set (20%), used to optimize the hyperparameters and avoid overfitting.
- Test set (20%), used to evaluate the final performance of the model.

C. CNN Implementation on FPGA

The ZYBO Xilinx-Zynq7000 development platform [16] is a high-performance, ready-to-use entry-level platform for embedded software and digital circuit development. It is built around the Z-7010, the latest addition to the Xilinx Zynq-7000 family. The Z-7010 is based on the Xilinx AP SoC architecture, which integrates a dual-core ARM Cortex-A9 processor with Xilinx 7-series FPGA logic. Combined with the wide range of multimedia and connectivity peripherals available in ZYBO, the Zynq Z-7010 can support an entire system design. Its onboard memory, video and audio I/O, dual-role USB, Ethernet, and SD card slot simplify the development process by eliminating the need for additional hardware [17]. Additionally, the six Pmod ports offer an easy way to expand the design's capabilities as needed. Table II provides a detailed summary of the features and specifications of the ZYBO Xilinx-Zynq7000 platform.

TABLE II. FEATURES AND SPECIFICATIONS OF THE ZYBO XILINX-ZYNQ7000

Specifications	Features
-Xilinx XC7Z010-1CLG400C. -1GB DDR3L with a 32-bit bus at 1066MHz. -16MB Quad-SPI flash. -Gigabit Ethernet PHY. -USB OTG PHY with host and device support. -PCAM camera connector with MIPI CSI-2 support. -Pmod connectors. -HDMI sink port (input). -HDMI source port (output). -Audio codec with stereo headphone, stereo line-in, and microphone jacks. -Powered by USB or any 5V external power source.	-1× MSPS on-chip ADC. -17,600 look-up tables. -35,200 flip-flops. -270KB RAM. -2× clock management tiles. -5 Pmod ports. -TX port HDMI CEC support. -1× RGB LED.

The experimental setup is based on an FPGA Z-7010 board serving as the main processing platform. A USB HD camera is used to capture images, while a display screen shows real-time results. A servo motor controls the door, opening it when access is authorized. Access status is indicated using LEDs: a green LED lights up for authorized access, while a red one signals denied access. Additionally, a buzzer is triggered to alert unauthorized access attempts. Communication with a web server and a mobile application is facilitated through an Ethernet cable, allowing remote management and seamless integration into a connected system (Figure 5).

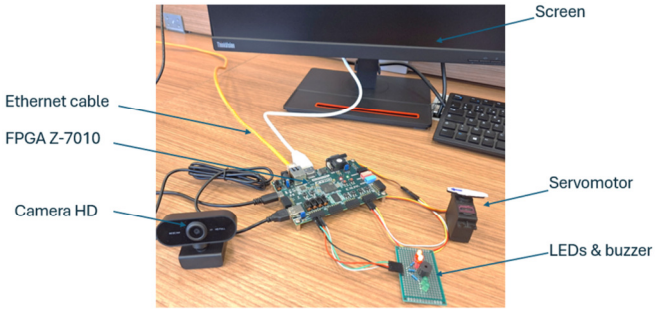

Fig. 5. ZYBO Xilinx-Zynq7000 with the experimental setup.

V. RESULTS AND DISCUSSION

A. Simulation Results

To establish a robust image classification pipeline, a baseline CNN model and advanced versions of R-CNN, Fast R-CNN, and Faster R-CNN architectures were developed.

1) Baseline CNN

The baseline CNN consisted of three Conv2D layers with 64, 96, and 128 filters, respectively, all using 3×3 kernels and ReLU activation. It also included additional components such as four batch normalization layers, three MaxPooling2D layers (2×2), three dropout layers to reduce overfitting, a flatten layer to transition from convolutional to FC layers, and two dense layers for final classification. This architecture demonstrated robust performance, achieving an accuracy of 0.96 after 800 iterations, significantly outperforming an RNN that achieved an accuracy of only 0.85. These results highlight the superior capability of CNNs to capture spatial hierarchies in data, making them more effective for image classification tasks than RNNs.

TABLE III. ACCURACY COMPARISON

DL models	CNN	RNN
Accuracy	0.96	0.85

2) R-CNN Models

Three advanced CNN architectures, an R-CNN, a Fast R-CNN, and a Faster R-CNN, were developed and optimized to compare their performance with the CNN model. The R-CNN model utilized 96, 192, and 256 filters in its three convolutional layers, with kernel sizes of 3, 5, and 5, respectively, and included a dropout rate of 0.2 in the first dropout layer and 0.4

in the second, along with 320 dense units and a learning rate of 0.001. The Fast R-CNN model improved upon this by employing 128, 256, and 128 filters with kernel sizes of 3, 5, and 3, respectively, and included dropout rates of 0.2 and 0.5, 384 dense units, and the same learning rate of 0.001. Last, the Faster R-CNN model introduced further optimizations, utilizing 128, 128, and 384 filters with kernel sizes of 5, 5, and 3, respectively, and included dropout rates of 0.3 and 0.5, 256 dense units, and a higher learning rate of 0.01. Table IV shows the best parameters for these models.

TABLE IV. BEST HYPERPARAMETER VALUES FOR R-CNN, FAST R-CNN, AND FASTER R-CNN

	R-CNN	Fast R-CNN	Faster R-CNN
Filters_1	96	128	128
Kernel_size_1	3	3	5
Filters_2	192	256	128
Kernel_size_2	5	5	5
Filters_3	256	128	384
Kernel_size_3	5	3	3
Dropout_1	0.2	0.2	0.3
Dense_units	320	384	256
Dropout_2	0.4	0.5	0.5
Learning_rate	0.001	0.001	0.01

3) *Performance Metrics*

The performance of the CNN model highlights its strengths and limitations in face detection tasks. Its superior performance compared to the RNN is attributed to its ability to capture spatial hierarchies in image data, making it more effective for image classification tasks. Consequently, the focus was on the CNN-based architecture. The R-CNN model provided a solid foundation for face detection, but its sequential approach to region proposal and classification led to slower processing times but achieved an accuracy of 0.89. Fast R-CNN improved upon this by integrating region proposal generation directly into the model's structure, resulting in a more efficient pipeline, faster convergence, and a slightly higher accuracy of 0.95. The Faster R-CNN model introduced a significant breakthrough by incorporating a Region Proposal Network (RPN) to streamline the generation of regions of interest, dramatically improving both speed and accuracy. With an accuracy of 0.96, Faster R-CNN emerged as the best-performing model, particularly suited for real-time face detection applications. Contrary to the initial claim, the accuracy results show that the Faster R-CNN achieved the highest accuracy, followed by Fast R-CNN, with R-CNN performing the least effectively. These results align with the architectural advancements of each model:

- R-CNN: Accuracy of 0.89, limited by its slower and less efficient region proposal process.
- Fast R-CNN: Accuracy of 0.95, benefiting from the integrated approach that combines region proposal and classification.
- Faster R-CNN: Accuracy of 0.96, leveraging an RPN for faster and more accurate predictions.

TABLE V. COMPARISON OF ACCURACY BETWEEN R-CNN, FAST R-CNN, AND FASTER R-CNN

Models	R-CNN	Fast R-CNN	Faster R-CNN
Accuracy	0.89	0.95	0.96

Based on these metrics, the Faster R-CNN is recommended for face classification tasks due to its superior accuracy and computational efficiency. This model was deployed on an FPGA board for real-time performance.

B. *Implementation Results*

The proposed system utilizes strategically placed cameras near the data center access doors to capture real-time images of individuals attempting to enter. These cameras isolate the individual's face from the background, filtering out irrelevant elements such as trees, walls, or other objects. The captured images are then transmitted to an FPGA board, where a CNN model processes them. The CNN analyzes the images, detecting and classifying individuals to determine access permissions. Access is granted or denied based on the classification results. The system is designed to continuously improve by incorporating new data and updating its parameters, ensuring optimal accuracy over time. Additionally, the system features robust database management, allowing for the addition of new authorized individuals by capturing multiple facial images from various angles and enabling the removal of individuals. The proposed implementation, illustrated in the following figures, consists of an FPGA board where the Faster R-CNN is implemented, a camera for face detection, a motor for door opening, and LED indicators. The green LED lights up when access is granted, whereas the red LED indicates denial. The figures also show the state of the LEDs and the servo motor depending on access authorization. If access is denied, the red LED is turned on and the servo motor remains in its initial position at 0° (door closed), as shown in Figure 6. In contrast, when access is granted, the green LED lights up and the servo motor rotates to 180° (door open), as shown in Figure 7. Additionally, data, including the captured image, date, and time, is transmitted to the web server to be displayed in the mobile application and logged for history management.

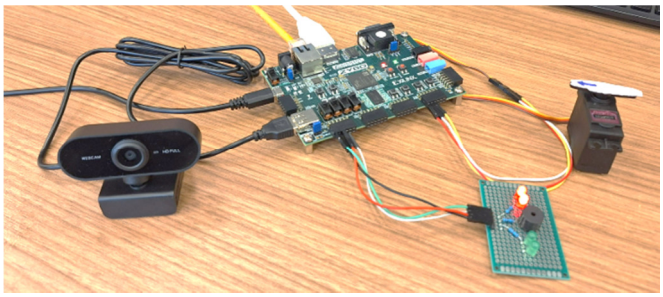

Fig. 6. The circuit when access is blocked.

To ensure real-time monitoring and ease of use, a cross-platform mobile application was developed using Flutter and Dart [18], compatible with both Android and iOS devices. The application features a robust user authentication system that requires a username and password to secure access. Once

authenticated, users are directed to the home page, which serves as the central hub for user management. Here, administrators can add or remove users and manage access groups. The application categorizes individuals into two groups: authorized users, who have access to the rooms, and blocked users, who are denied entry. When an unauthorized or blocked user attempts access, an error message is prominently displayed on the home page. For administrative purposes, the app includes a dedicated settings page for modifying account details and managing user roles. Additionally, the application settings allow for customization of language, theme, and display preferences, enhancing user accessibility and providing a personalized experience. This user-friendly design ensures effective real-time monitoring and streamlined management of the access control system. Figure 8 shows the user interface of the application, showcasing its clean and intuitive design. Key features such as user group management and error notifications ensure seamless operation and effective access control oversight.

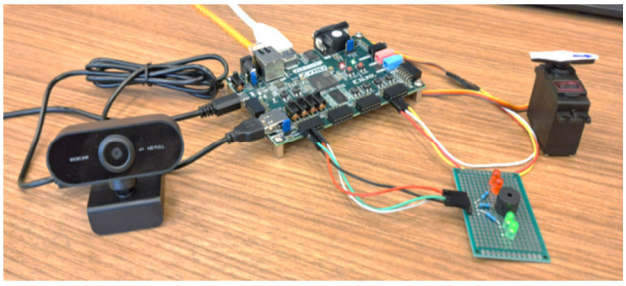

Fig. 7. The circuit when access is authorized.

Fig. 8. Real-time monitoring application.

This implementation improves security with real-time facial recognition, ensuring that only authorized personnel can access data centers. It is scalable, allows updates to the database, and features a user-friendly mobile app to manage users and settings. Real-time monitoring enables administrators to track access attempts and make instant adjustments. Overall, the proposed system combines AI and programmable hardware for a robust, efficient, and adaptable security solution.

VI. CONCLUSION

This work demonstrated the development and deployment of a robust security system leveraging a CNN on a Xilinx Zynq-based ARM embedded system, implemented using the Xilinx PYNQ framework. Through a detailed analysis and optimization of both software and hardware components, the system effectively balances the speed of inference and accuracy, making it highly suitable for real-time applications in embedded environments. The implementation of the physical door security system shows the practical application of CNN-based image classification for real-time face detection. The results indicate that the CNN achieves superior performance, with an accuracy of 96%, far surpassing the RNN's accuracy of 85%, thus solidifying its suitability for image classification tasks. Further analysis of advanced CNN architectures, including R-CNN, Fast R-CNN, and Faster R-CNN, highlights the superiority of Faster R-CNN, which achieved the highest accuracy of 96%. This model's streamlined architecture, incorporating an RPN, allowed for enhanced computational efficiency and real-time capabilities.

The system's scalability and adaptability were demonstrated through its ability to continuously update the database by adding new individuals or removing existing ones. Additionally, the integration of a user-friendly mobile application for real-time monitoring and management further enhances its practicality for administrators. Future work may focus on improving the deep learning modules by using a larger dataset, including multiple classes and images of individuals captured from various angles and positions. Furthermore, future research could explore the implementation of these modules on alternative electronic boards, such as the Raspberry Pi, to identify the most suitable platform for real-time decision-making to enhance the physical security of departments.

REFERENCES

[1] A. Sharma, V. Singh, and A. Rani, "Implementation of CNN on Zynq based FPGA for Real-time Object Detection," in *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kanpur, India, Jul. 2019, pp. 1–7, https://doi.org/10.1109/ICCCNT45670.2019.8944792.

[2] A. Manel, "Système de détection de visage, d'émotion, de sexe et classification d'âge en Deep Learning," University BBA, 2023.

[3] A. Yahiaoui, "Système de Discrimination Visages / Faux Visages par Réseaux de Neurones Convolutifs (CNN)," M.S. Thesis, Université de Guelma, 2021.

[4] T. Bouras, "Modèle basé sur le Transformer pour une détection robuste du visage des nourrissons et des enfants hospitalisés en utilisant des images RVB et thermiques," M.S. Thesis, Université du Quebéc, 2024.

[5] Y. Said, M. Barr, and H. E. Ahmed, "Design of a Face Recognition System based on Convolutional Neural Network (CNN)," *Engineering, Technology & Applied Science Research*, vol. 10, no. 3, pp. 5608–5612, Jun. 2020, https://doi.org/10.48084/etasr.3490.

[6] W. Zhe *et al.*, "A Research on Two-Stage Facial Occlusion Recognition Algorithm based on CNN," *Engineering, Technology & Applied Science Research*, vol. 14, no. 6, pp. 18205–18212, Dec. 2024, https://doi.org/10.48084/etasr.8736.

[7] S. Ramakrishnan, "Implementation of a Deep Learning Inference Accelerator on the FPGA.," M.S. Thesis, Lund University, 2020.

[8] Y. Fukushima, K. Iizuka, and H. Amano, "Parallel Implementation of CNN on Multi-FPGA Cluster," *IEICE Transactions on Information and Systems*, vol. E106.D, no. 7, pp. 1198–1208, Jul. 2023, https://doi.org/10.1587/transinf.2022EDP7175.

[9] T. Aarrestad *et al.*, "Fast convolutional neural networks on FPGAs with hls4ml," *Machine Learning: Science and Technology*, vol. 2, no. 4, Dec. 2021, Art. no. 045015, https://doi.org/10.1088/2632-2153/ac0ea1.

[10] A. Faid and N. E. Ferhat, "Commande DTC du moteur asynchrone par réseaux de neurones artificiels implémenté sur FPGA," M.S. Thesis, University of M'sila, 2022.

[11] X. Li and L. Zhang, "Wearable damaged clothing fabric image recognition system based on image restoration algorithm," *Systems and Soft Computing*, vol. 6, Dec. 2024, Art. no. 200140, https://doi.org/10.1016/j.sasc.2024.200140.

[12] M. Ouloul, Z. Moutakki, A. Amghar, and K. Afdel, "Low-cost embedded facial recognition system based on overlapped local binary pattern," *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 11, Mar. 2025, Art. no. 100924, https://doi.org/10.1016/j.prime.2025.100924.

[13] T. D. S. Ramos, "Implementation of Convolutional Neural Networks on a Versal Device," M.S. Thesis, Universidade do Porto, 2023.

[14] M. S. Azzaz, A. Maali, R. Kaibou, I. Kakouche, M. Saad, and H. Hamil, "FPGA HW/SW Codesign Approach for Real-time Image Processing Using HLS," in *2020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP)*, El Oued, Algeria, May 2020, pp. 169–174, https://doi.org/10.1109/CCSSP49278.2020.9151686.

[15] "Avengers Faces Dataset." Kaggle, [Online]. Available: https://www.kaggle.com/datasets/yasserh/avengers-faces-dataset.

[16] A. Benahmed and Z. Guennoun, "FPGA based Hardware Co-Simulation Implementation for RealTime Image Blind Separation using ICA Algorithms," *International Journal of Emerging Technology and Advanced Engineering*, vol. 12, no. 10, pp. 75–81, Oct. 2022, https://doi.org/10.46338/ijetae1022_09.

[17] M. Mekhfioui, R. Elgouri, A. Satif, M. Moumouh, and L. Hlou, "Implementation Of Least Mean Square Algorithm Using Arduino & Simulink," *International Journal of Scientific & Technology Research*, vol. 9, no. 4, pp. 664–667, 2020.

[18] M. Mekhfioui, A. Benahmed, A. Chebak, R. Elgouri, and L. Hlou, "The Development and Implementation of Innovative Blind Source Separation Techniques for Real-Time Extraction and Analysis of Fetal and Maternal Electrocardiogram Signals," *Bioengineering*, vol. 11, no. 5, May 2024, Art. no. 512, https://doi.org/10.3390/bioengineering11050512.