

Acceleration of convolutional neural network based diabetic retinopathy diagnosis system on field programmable gate array

Meriam Dhouibi, Ahmed Karim Ben Salem, Afef Saidi, Slim Ben Saoud

Advanced Systems Lab, Tunisia Polytechnic School, University of Carthage, La Marsa, Tunisia

Article Info

Article history:

Received Dec 21, 2022

Revised Apr 8, 2023

Accepted Apr 24, 2023

Keywords:

Convolutional neural networks

Diabetic retinopathy

Diagnosis system

Embedded systems

Field programmable gate array

ABSTRACT

Diabetic retinopathy (DR) is one of the most common causes of blindness. The necessity for a robust and automated DR screening system for regular examination has long been recognized in order to identify DR at an early stage. In this paper, an embedded DR diagnosis system based on convolutional neural networks (CNNs) has been proposed to assess the proper stage of DR. We coupled the power of CNN with transfer learning to design our model based on state-of-the-art architecture. We preprocessed the input data, which is color fundus photography, to reduce undesirable noise in the image. After training many models on the dataset, we chose the adopted ResNet50 because it produced the best results, with a 92.90% accuracy. Extensive experiments and comparisons with other research work show that the proposed method is effective. Furthermore, the CNN model has been implemented on an embedded target to be a part of a medical instrument diagnostic system. We have accelerated our model inference on a field programmable gate array (FPGA) using Xilinx tools. Results have confirmed that a customized FPGA system on chip (SoC) with hardware accelerators is a promising target for our DR detection model with high performance and low power consumption.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Meriam Dhouibi

Advanced Systems Lab, Tunisia Polytechnic School, University of Carthage

BP 743, 2078 La Marsa, Tunisia

Email: meriam.dhouibi@ept.rnu.tn

1. INTRODUCTION

The diabetic retinopathy (DR) is a medical condition that causes vision loss due to an increase in blood glucose level. At least 783 million people are predicted to have diabetic issues by 2045 [1]. The increase in blood glucose level causes a hormonal imbalance that produces eye fatigue when blood pressure and glucose levels are high, the capillaries and veins in the retina are damaged, preventing blood flow and resulting in DR. It became blindness when the treatment is not received in a time. According to medical specialists, an early detection of DR helps avoid severe visual loss [2]. DR should be monitored on a frequent basis. Therefore, a basic checkup could be performed twice a year. However, the management of this condition is timely consuming treatment that can be controlled by monitoring blood glucose levels.

The traditional technique for assessment and analysis to discover morphological abnormalities in the eyes makes the experimental procedure more complicated and time consuming. An ophthalmologist diagnoses the color fundus of the patient and then processes the patient condition. Most of the time, patients are not treated on time due to a lack of facilities. Once DR is detected early, the deteriorating process can be controlled, preventing irreparable vision loss. Manual diagnosis of this condition can often result in misdiagnoses.

Automated techniques allow early DR detection, which has a huge public health benefit in terms of preventing blindness. A fundus camera is commonly used in automated systems, which is connected to another device that has the diagnosis algorithm; these devices can be personal computers (PCs) or smartphones (based on central processing units (CPUs) or graphical processing units (GPUs)). By integrating the algorithm within ophthalmic imaging equipment (using field programmable gate array (FPGA) system on chip (SoC)), we present a more performant embedded DR diagnosis system. Figure 1 presents the three different approaches for DR diagnosis.

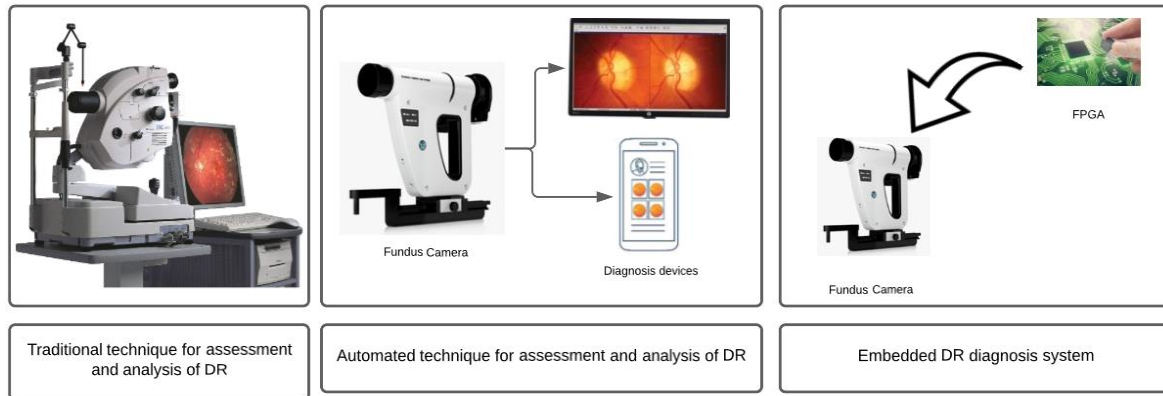


Figure 1. Different DR diagnosis techniques

Deep neural networks (DNNs) especially convolutional neural networks (CNNs), have established themselves as the most popular strategy in the current scenario, with higher performance in a variety of domains, and above traditional machine learning approaches particularly in image analysis and treatment [3]. However, this task may now be completed quickly and easily using advanced deep learning (DL) techniques such as CNNs in a hardware-based solution. Nevertheless, the implementation of CNN on hardware platforms remains challenging. CNNs usually impose large demands on computation and memory consumption given that the number of operations and parameters increases with the complexity of the model architecture. Therefore, the performance will strongly depend on the hardware target resources. This CNN's growing complexity has made researchers think about using optimization strategies to make them more hardware friendly and designing hardware accelerators to enhance their performance on embedded systems.

In the first part of this study, we used transfer learning to design our model. We preprocessed the input data, which is color fundus photography, to reduce undesirable noise in the image. No DR, mild DR, moderate DR, severe DR, and proliferate DR are the five different DR stages detected by the model. Extensive experiments and comparisons with other research work show that the proposed method is effective.

In the second part, to obtain a medical diagnostic device for DR, we implemented the designed CNN model on embedded platform while applying optimization techniques. In addition, we have accelerated our inference model on the FPGA platform using hardware intellectual property (IP). To further demonstrate the advantages of such a hardware architecture we implemented our model on Google GPU and Google tensor processing unit (TPU).

CNNs have widely proved efficient with a higher number of applications related to computer vision tasks [4], [5] and medical applications [6], [7]. The CNN has been used in recent studies to automate the detection and classification of DR [8], depending on the classification method employed. These studies are categorized as binary classification to detect DR and multi-level classification to assess the proper stages.

Xu *et al.* [9] used a tiny architecture of a CNN model to automatically classify the color fundus images in the Kaggle dataset as normal or DR images with an accuracy of 94.5%. According to Hajabdollahi *et al.* [10], the original visual geometry group16 (VGG16) network is updated and used for DR binary classification. The model achieved 93.89% accuracy. Furthermore, a hierarchical pruning strategy is proposed to simplify the CNN structure, resulting in a 1.89% accuracy loss.

Kwasigroch *et al.* [11] offered an automatic detection approach for DR based on CNN and demonstrated that CNNs may be successfully used to handle this kind of difficult task. The obtained model was tested on two tasks based on the examination of retinal images: the first is the detection of DR and the second is the determination of its stage. The model had an accuracy of around 82% in diagnosing DR and 51% in determining its stage. Shaban *et al.* [12] proposed a deep CNN architecture that can accurately

categorize DR into 3 classes: no DR, moderate DR (which includes patients with mild or moderate non-proliferative diabetic retinopathy (NPDR), and severe DR (which includes patients with severe NPDR or proliferative diabetic retinopathy (PDR) in the late stages) with an accuracy of 89%. Automated analysis of retinal color images has such advantages as increased reliability and coverage of screening programs, reduced barriers to access, and early detection and treatment. In the domain of retinal image analysis, all top solutions used CNNs to identify signs of DR in retinal images.

From the hardware perspective, Ghani *et al.* [13] presented an automated retinal fundus image detection approach for glaucoma and DR based on artificial neural network (ANN). The proposed tiny architecture classifier was implemented on the Artix-7 FPGA platform, yielding high performance (100% accuracy and nearly 400 ns of latency). According to Washburn *et al.* [14], the RAPIDS compute unified device architecture (CUDA) machine learning libraries (cuML) based fuzzy c-mean clustering segmentation method was used for the processing and segmentation of input retinal images. The authors proposed a hardware-based system to help ophthalmologists with the mass screening of DR. On the NVIDIA Jetson, the system performed high-speed detection of lesions on input retinal images (300 ns latency).

Pritha *et al.* [15] implemented AlexNet for retinal disease detection on FPGA. The proposed CNN model categorizes retinal images into 4 classes (optical disk cartridge (ODC), diabetic nephropathy (DN), central serous retinopathy (CSR), and normal image) in 2 seconds. Compared to software implementation of solutions related to RD detection, the results on hardware platforms demonstrate the enhancement of performance in terms of latency and energy efficiency with the flexibility of the embedded system. In our work, we implemented a large CNN model with multi-level classification to accurately identify the proper stages of DR with high performance on different platforms. This paper is structured as follows: the proposed approach and the adopted method with different experiments are detailed in section 2. Results are evaluated in section 3. Finally, a summary of this research output and potential future work is presented in section 4.

2. METHOD

The proposed pipeline for designing, implementing, and accelerating a DR system is shown in Figure 2. First, a CNN model dedicated to DR detection has been designed, validated, and trained. In this step, the ResNet50 structure has been updated, and the pre-trained network parameters are fine tuned using preprocessed color fundus photography as input data. Then, optimization techniques have been applied to the model to make it more efficient and hardware-friendly. Finally, it has been deployed and accelerated on the Zynq FPGA using the Xilinx data processing unit (DPU). Different steps of the proposed approach are presented in the following subsections.

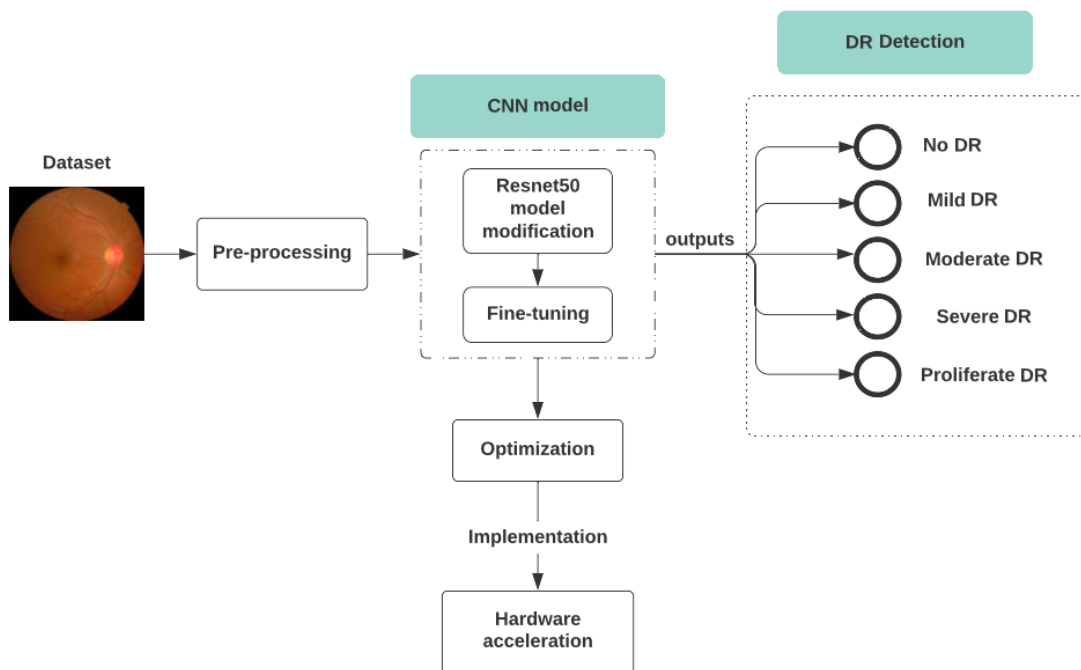


Figure 2. Pipeline of the proposed system for DR detection

2.1. Dataset

Retinal images are acquired using different imaging conditions and equipment, which results in very mixed image quality. Subtle signs of retinopathy at an early stage can be easily masked in a low contrast, blurred, or low resolution image. Analysis of an image of low quality may produce unreliable results when the system labels an image as normal while lesions are present. That is why image quality is a very important factor for DR detection system sensitivity.

For our application, we have used a dataset that consists of a large set of high resolution retina images provided by Kaggle [16]. A clinician has rated each image on a scale of 0 to 4 according to the international clinical diabetic retinopathy (ICDR) severity scale, which is explained: i) 0 is no DR; ii) 1 is mild DR; iii) 2 is moderate DR; iv) 3 is severe DR; and v) 4 is proliferate DR. Figure 3 shows a representative sample of each class. We used 3,662 images for training and 1,992 images for the test. The training data is distributed on the previous scale, as Figure 4.

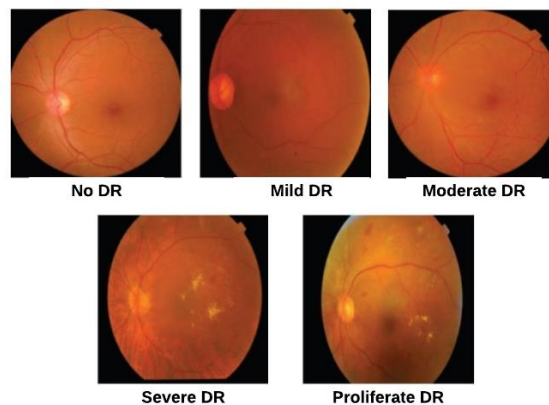


Figure 3. Images from each of the five DR classes

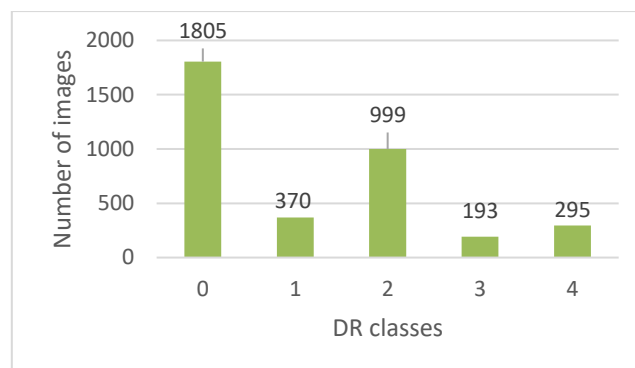


Figure 4. Training dataset distribution

2.2. Model architecture

With the advances in CNNs, it has become possible to design and train models to automatically discover subtle local features without the need for manual annotation of individual lesions. The neural network used in this work is a CNN with a deep layered structure that combines nearby pixels into local features and then progressively aggregates those into global hierarchical features. The model does not explicitly detect lesions; instead, it learns to recognize them based on local features. It was trained on a classification task and produced a continuous score between the five previous classes, which indicates the presence and severity of DR.

The model architecture used in this work originates from the ResNet50 network since it is highly reliable when processing microscopic and complicated images and it solves the vanishing gradient problem. ResNet central concept is to introduce an "identity shortcut connection" that skips one or more layers to minimize information loss Figure 5. Each basic block is a sequence of Conv2D layers followed by a batch normalization layer and a ReLU activation function, all of which is added to an identity shortcut at the end.

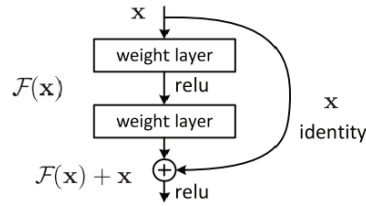


Figure 5. ResNet architecture

2.3. Transfer learning

Transfer learning is a methodology in which a model that has been trained for one task is repurposed for another. Recently, it has gradually become a topic in both academia and industry [17], [18]. Transfer learning focuses on transferring knowledge (features and weights) across domains. Most classification applications based on CNNs are making increasing use of transfer learning technique.

The main idea is to pretrain models on a big dataset and then fine tune models on specific target datasets. The obtained parameters from the first model are used as the initialization of the second model. Yosinski *et al.* [19] demonstrated that transferred parameter initialization is better than random parameter initialization. Since then, transfer learning has been applied to a variety of tasks in different fields. In addition to providing the capability to use already constructed models, transfer learning has several benefits. It speeds up the process of the training and can also result in a more accurate and effective model in most cases. In our study, the transfer learning technique was used to speed up training and improve the model performance. A pre-trained ResNet50 model on the large dataset ImageNet is used as the starting point and then refined on our Kaggle dataset related to retina images.

2.4. Hardware acceleration

After testing and validating the optimized CNN inference on CPU, we are moving to a strictly hardware implementation on FPGA looking for more speedup and energy efficiency. Indeed, we are going to concentrate on accelerating our CNN inference on heterogeneous FPGA, including embedded cores, using a specifically designed hardware accelerator. Such heterogeneous SoCs provide effective platforms for embedded, complex CNN based applications. We are mainly interested in implementing the CNN inference on the Xilinx ZYNQ FPGA using DL DPU. Its architecture exploits the cooperation between the processing system (PS) and the programmable logic (PL) in Xilinx Zynq devices. The Xilinx deep neural network development kit (DNNDK) [20] is also being explored to significantly simplify the implementation of optimized CNN models by automatically mapping these models on FPGA platforms. This tool can accelerate the mapping of the CNN inference phase onto Xilinx hybrid CPU-FPGAs SoC through an easy to use C/C++ programming interface.

2.4.1. Development board

Experiments have been carried out on the Xilinx ZedBoard, illustrated in Figure 6. It is based on Xilinx Zynq-7000 All Programmable SoC. The Zynq architecture provides a PL (FPGA), a PS and a large number of communication interfaces. A dual core advanced risc machine (ARM) Cortex-A9 processor with a variety of peripheral connectivity interfaces is included in the PS. ZedBoard has proven efficient for rapid prototyping and proof of concept development [21].



Figure 6. ZedBoard [21]

2.4.2. Data processing unit overview

Research by Xilinx released DPU [22] as an IP core to improve CNN inference performance on hybrid FPGAs. It is a programmable engine optimized for CNNs and their demanding computing functions. It uses a specialized instruction set and supports a great variety of CNNs. DPU IP can be implemented in the PL as a co-processor connected directly through an AXI interconnect to the main multicore ARM processor of the Zynq@-7000 SoC or Zynq UltraScale+™ MPSoC devices. The CNN model and its massive computations are computed in parallel in the PL. The not supported layer functions of some CNN models are traditionally handled by the processor. A high performance scheduler, an instruction fetch unit, a hybrid computing array, and a global memory pool are all part of the DPU. Figure 7 presents the hardware architecture of the DPU.

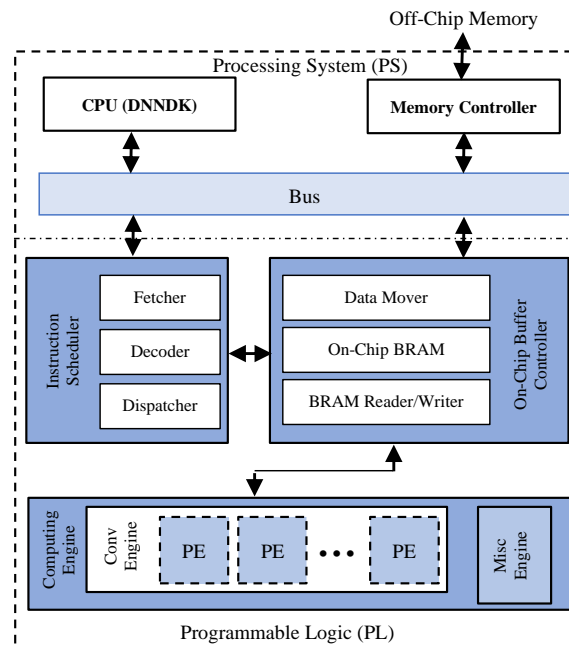


Figure 7. DPU Hardware architecture [22]

The DPU uses quantization and high clock frequency in digital signal processor (DSP) slices. Quantization is required and can be applied using the DNNDK tool. DSP slices are clocked at a very high frequency using a DSP double data rate (DDR) technique [22], which uses a $2\times$ frequency domain to increase peak performance. The DPU requires instructions to implement the model. These instructions are generated by the deep neural network compiler (DNNC), where optimizations have been made.

To control the operation of the computer engine, the DPU fetches these instructions from the offchip memory. For high performance, data is buffered in on-chip memory, and it is reused as much as possible to save memory bandwidth. A deep pipeline design is used for the computing engine. The processing element (PE) in Xilinx devices make full use of fine grained building blocks including multipliers, adders, and accumulators.

2.4.3. Design and implementation flow

The deployment of DNN into Xilinx DPU platforms needs different steps, as illustrated in Figure 8. In this part, the Xilinx Vivado design flow steps have been followed. To configure the PL part of the Zynq-7000 and integrate the DPU IP, the design block is designed first. We need mainly the PS and the DPU IP. Other IPs are instantiated to meet the hardware requirements for the proper functioning of the system. Some configurations on the DPU are required, in order to respond to CNN architecture.

After connecting all the IP components, the very high hardware description language (VHDL) files are created so that the Vivado synthesizer can interpret and manage the design block. After the implementation phase, the design was completely routed on the FPGA. A Bitstream is generated that will be exported to be used as the base hardware for software development. The AXI4 Interconnect serves as a vital communication connection between the PS and the PL for bidirectional data transfer. The PS accesses the

input images stored on the SD card and then inputs them to the DPU during inference. Data input can be visualized on external displays connected directly to PS (universal asynchronous receiver transmitter (UART) and secure shell (SSH)).

In the second part, we used Petalinux to generate a custom Linux image that incorporates the DL aspect of the project and the Xilinx DNNDK tool. DNNDK is an integrated framework that offers a complete range of toolchains for inference with the DPU, including compression, compilation, deployment, and profiling. Among the components of DNNDK are the deep compression tool (DECENT) and the DNNC. The quantizer DECENT is used for the model quantization with the precision of INT8. The compiler DNNC generates the DPU instructions related to the implementation of the model and memory access. This results in an executable linker file (ELF), which is then compiled together with other custom C/C++ program instructions that control other tasks of the DL algorithm, such as image loading, visualization, and other preprocessing tasks. The CPU and DPU elf files are then compiled into a single file that is loaded onto the board. Then, the CNN inference tasks are initialized in ARM and assigned to DPU. Figure 8 presents the flow for hardware and software implementation of CNN on FPGA using DPU IP.

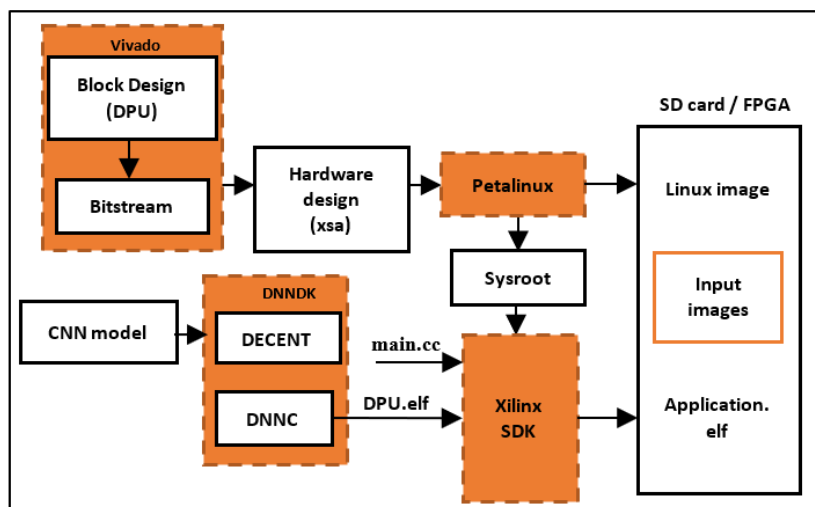


Figure 8. Hardware and software implementation flow

2.4.4. Data processing unit configuration and hardware resource utilisation

In this section, we aim to evaluate different configurations of DPU to provide a more comprehensive analysis. A variety of DPU architectures are available and can be easily adapted to be supported on different embedded boards. It can be configured with a variety of convolution architectures based on the convolution unit parallelism. From B512 to B4096 are the DPU IP architectures [22]. The DPU convolution architecture has three dimensions of parallelism: pixel parallelism, input channel parallelism, and output channel parallelism. Each architecture requires different programmable logic resources, and larger architectures with more resources can achieve higher efficiency. The B4096 architecture, for example, makes use of all available hardware resources in terms of block RAMs (BRAMs) and DSPs, resulting in a peak performance of 4096 operations per cycle.

Some configurable parameters in the DPU IP can be used to optimize resource utilization and customize different features. Depending on the amount of programmable logic resources available, different configurations for DSP and BRAM utilization can be selected. In addition, a single DPU IP may have a maximum of three cores. To achieve higher efficiency, multiple DPU cores can be used. As a result, more programmable logic resources are consumed.

2.4.5. Implementation on data processing unit

To implement a CNN inference on an FPGA using DPU IP and Xilinx tools, we need to follow the steps presented in Figure 9. Firstly, the model is trained and prepared on a PC using the TensorFlow (TF) framework. Secondly, the Xilinx DNNDK is used to quantize and compile the model and to generate the DPU instructions. Then, the CNN inference tasks are initialized and assigned to the DPU by the PS.

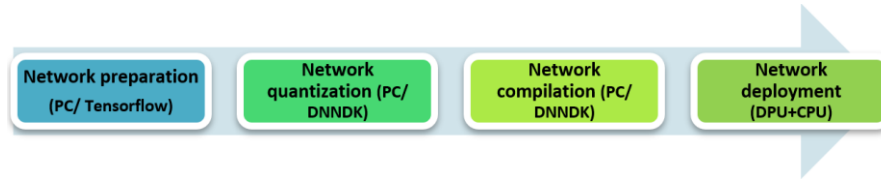


Figure 9. Workflow for CNN implementation on DPU

3. RESULTS AND DISCUSSION

3.1. Model performance

In our experimentation, we used transfer learning to design our model based on state of the art CNN architectures such as VGG, Xception, and ResNet50. The last model produced the best results after the training process, as shown in Table 1, we designed a CNN model based on a modified version of the ResNet50 to detect DR and assess its stage. We fitted our model with the training dataset and it got to 98% training accuracy and 92.90% validation accuracy, which means that the model is well trained for prediction and well generalized. At this point, we can evaluate our model by running the inference phase. Our model performs 120 s on an Intel Core i7-6700 U CPU quad core at 2.60 GHz with 16 GB RAM.

Table 1. Different model performance for DR classification

Model	Accuracy (%)
ResNet-50	92.9
Xception	92
VGG-16	88.15

Comparisons with other research studies are presented in Table 2 to better evaluate the performance. Our proposed model outperforms other works in terms of accuracy with multilevel classification. High accuracy is critical in practically all medical applications, especially in detecting DR and assessing its stage. A misdiagnosed DR can lead to a lack of therapy and further progression of the illness, which can cause blindness.

Table 2. Performance comparison of different DR classification methods

Method	Classification type	Accuracy (%)
Inception V3 [23]	Multi-level (5 class)	63.23
VGG-D [11]	Binary (DR/NO DR)	81.70
VGG-D [11]	Multi-level (5 class)	51
CNN [24]	Multi-level (5 class)	74.04
CNN [12]	Multi-level (3 class)	89
VGG16 [10]	Binary (DR/RDR)	93.80
ResNet50 (Ours)	Multi-level (5 class)	92.90

Furthermore, we run our DR detection model on the Google colaboratory GPU and TPU platforms. The GPU is an NVIDIA TESLA K80 with 2 CPU cores, 12 GB of RAM, and up to 4.1 teraflops (TFLOPS) of performance. While the TPUs were TPUv2 (8 core) with an Intel Xeon (4 core) CPU and 16 GB of RAM. As the results show in Table 3, our model performed well on the Google GPU computational platform. Furthermore, the GPU performed better than the TPU in every case. This is due to the fact that TPU setup takes some time when compiling the model and distributing the data in the clusters, so the first iteration always takes time.

Table 3. DR detection model performance summary

	Accuracy (%)	Latency (s)		
		CPU	Google GPU	Google TPU
Our model	92.9	120	0.75	0.89

3.2. Hardware acceleration

The inference phase of the DR detection system based on a complex CNN model has been implemented successfully on a limited resource FPGA, the Xilinx Zynq-7000 SoC. In order to test and

validate the impact of deploying a hardware Xilinx DPU accelerator on the inference phase performance, different possible DPU architectures supported on the board, have been tested looking for good performance with less HW resource utilization. The best peak of performance is achieved with the B1152 architecture.

The CNN model with 8-bit quantization embedded in the Zynq-7000 predicts DR and evaluates its stage on a single input image in 26.2 ms while consuming only 3.237 watts. More details are presented in Table 4. With the use of the B1024 DPU architecture, we achieved comparable performance with more area saving. However, results showed that using B800 and B512 DPU types decreased the performance by 50% (nearly the GPU performance).

Furthermore, the results presented in Table 5 show that our model inference on hybrid FPGA device outperforms the software implementation on CPU (PC), on Google GPU and on Google TPU with an average loss in accuracy. FPGA SoC has become a beneficial hardware device to replace the GPU for complex CNN implementation thanks to its powerful parallel processing and low power consumption. Hence, the hybrid Zynq-7000 FPGA device remains a viable candidate for implementing the complex CNN model used for the DR detection system. The obtained results show the efficiency of our system in detecting DR stages. Despite the high performance, our system cannot be compared to other related work, taking into consideration the difference in the task, the DL used model, and the hardware platform Table 6.

Subsequently, the whole system performance could be improved with some enhancements in DPU configuration. Therefore, we tested the use of two DPU cores with the four possible DPU architectures (B1152, B800, B1024, and B512). We have noticed that the use of hardware resources (DSP and RAM) has been doubled, which causes a resource issue that is not supported on our ZedBoard. The utilization of the DPU is limited by the computational capability and hardware resource capacity of the board. To improve performance and create a more powerful embedded DR detection system, this work can be further developed and implemented in a larger-resource FPGA SoC that supports multiple DPU core utilization.

Table 4. Implementation results on Zynq-7000 FPGA

	ResNet50	ResNet50 for DR detection
Workload (GOPS)	7.71	2.527
FPGA		Zynq-7000
DPU architecture		B1152
Memory usage (%)		83.93
DSP usage (%)		88.18
Latency (us)	100.636	26.200
Throughput (GOPS/s)	76.6127	96.45038

Table 5. Implementation results on Zynq-7000 FPGA and other platforms

Platform	ResNet-50 for DR detection		
	DPU	Google GPU	Google TPU
Latency (us)	26.200	52.000	57.000
Throughput (GOPS/s)	96.45038	48.5962	44.3333

Table 6. Performance comparison of similar implemented research work related to DR

Method	Classification type	Data precision	FPGA Platform	Accuracy (%)	Latency (s)
Detection of retinal diseases (AlexNet) [15]	4 classes: Optical disk cartridge (ODC) Diabetic nephropathy (DN) Central serous retinopathy (CSR) Normal image	NP	SPARTAN3AN	Training 100	2
Our proposed DR detection system (ResNet-50)	5 classes: No DR Mild DR Moderate DR Severe DR Proliferate DR	8-bit	ZedBoard	80.27	0.0262

4. CONCLUSION

This paper has demonstrated the feasibility of obtaining an embedded DR instrument diagnostic system by implementing a complex CNN model on FPGA SoC to detect DR stages. Using color fundus photography as input data, a CNN model for the detection of DR and its stages depending on severity is

presented. Our model performs well, with a 92.9% accuracy for five-class classification on the kaggle dataset. The model has been optimized and deployed on embedded platforms to obtain a medical diagnostic device. Our system can reduce the time of the manual diagnosis and be more effective than automated solutions that use a fundus camera linked to AI models running on CPUs or GPUs to diagnose DR.




The emphasis of this paper has also been on accelerating the complex and heavy CNN inference phase that has been adapted for DR application. In this work, the DPU accelerator has been used for the hardware (HW) implementation of our DR detection system on FPGA based SoCs. The implementation process of our CNN model on DPU has been studied and tested and it shows that, thanks to the Xilinx DPU IP and related tools, the model can be optimized, deployed, and tested easily on the FPGA SoC. The possible architectures of DPU and hardware resources have been investigated on the Zynq-7000 FPGA to improve the inference performance of our CNN model. Our embedded system, in a tiny Smart SoC, outperforms the GPU in terms of latency and power consumption. This work can be further developed and deployed in a larger UltraScale FPGA SoC using multiple DPU cores in order to achieve better performance and to obtain a more efficient embedded DR detection system.

REFERENCES




- [1] H. Sun *et al.*, “IDF Diabetes Atlas: Global, regional and country-level diabetes prevalence estimates for 2021 and projections for 2045,” *Diabetes Research and Clinical Practice*, vol. 183, Jan. 2022, doi: 10.1016/j.diabres.2021.109119.
- [2] A. Rakhlin, “Diabetic Retinopathy detection through integration of Deep Learning classification framework,” *bioRxiv*, 2018, doi: 10.1101/225508.
- [3] D. Mane, N. Londhe, N. Patil, O. Patil, and P. Vidhate, “A Survey on Diabetic Retinopathy Detection Using Deep Learning,” *Data Engineering for Smart Systems*, Springer, Singapore, 2022, pp. 621–637. doi: 10.1007/978-981-16-2641-8_59.
- [4] Y. Pi, N. D. Nath, and A. H. Behzadan, “Convolutional neural networks for object detection in aerial imagery for disaster response and recovery,” *Advanced Engineering Informatics*, vol. 43, pp. 1–14, Jan. 2020, doi: 10.1016/j.aei.2019.101009.
- [5] D. Konstantinidis, V. Argyriou, T. Stathaki, and N. Grammalidis, “A modular CNN-based building detector for remote sensing images,” *Computer Networks*, vol. 168, pp. 1–29, Feb. 2020, doi: 10.1016/j.comnet.2019.107034.
- [6] J. W. Sanders *et al.*, “Deep learning application engine (DLAE): Development and integration of deep learning algorithms in medical imaging,” *SoftwareX*, vol. 10, pp. 1–12, Jul. 2019, doi: 10.1016/j.softx.2019.100347.
- [7] E. Trivizakis *et al.*, “Extending 2-D Convolutional Neural Networks to 3-D for Advancing Deep Learning Cancer Classification With Application to MRI Liver Tumor Differentiation,” *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 3, pp. 923–930, May 2019, doi: 10.1109/JBHI.2018.2886276.
- [8] W. L. Alyoubi, W. M. Shalash, and M. F. Abulhair, “Diabetic retinopathy detection through deep learning techniques: A review,” *Informatics in Medicine Unlocked*, vol. 20, pp. 1–11, 2020, doi: 10.1016/j.imu.2020.100377.
- [9] K. Xu, D. Feng, and H. Mi, “Deep Convolutional Neural Network-Based Early Automated Detection of Diabetic Retinopathy Using Fundus Image,” *Molecules*, vol. 22, no. 12, pp. 1–7, Nov. 2017, doi: 10.3390/molecules22122054.
- [10] M. Hajabdollahi, R. Esfandiarpour, K. Najarian, N. Karimi, S. Samavi, and S. M. R. Soroushmehr, “Hierarchical Pruning for Simplification of Convolutional Neural Networks in Diabetic Retinopathy Classification,” in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Jul. 2019, pp. 970–973. doi: 10.1109/EMBC.2019.8857769.
- [11] A. Kwasigroch, B. Jarzembinski, and M. Grochowski, “Deep CNN based decision support system for detection and assessing the stage of diabetic retinopathy,” in *2018 International Interdisciplinary PhD Workshop (IIPHDW)*, IEEE, May 2018, pp. 111–116. doi: 10.1109/IIPHDW.2018.8388337.
- [12] M. Shaban *et al.*, “A convolutional neural network for the screening and staging of diabetic retinopathy,” *PLOS ONE*, vol. 15, no. 6, pp. 1–13, Jun. 2020, doi: 10.1371/journal.pone.0233514.
- [13] A. Ghani, C. H. See, V. Sudhakaran, J. Ahmad, and R. Abd-Alhameed, “Accelerating Retinal Fundus Image Classification Using Artificial Neural Networks (ANNs) and Reconfigurable Hardware (FPGA),” *Electronics*, vol. 8, no. 12, pp. 1–17, Dec. 2019, doi: 10.3390/electronics8121522.
- [14] P. S. Washburn *et al.*, “WITHDRAWN: Hardware-based identification of bright lesions to diagnose diabetic retinopathy,” *Materials Today: Proceedings*, pp. 1–5, Apr. 2021, doi: 10.1016/j.matpr.2021.03.335.
- [15] N. Pritha, T. Sujitha, S. S. Hivani, and Sharmila, “Detection of retinal diseases and implementation using FPGA,” *International Journal of Creative Research Thoughts*, vol. 10, no. 5, pp. e697–e709, 2022.
- [16] Kaggle, “Diabetic Retinopathy Detection,” *kaggle*, 2023. <https://www.kaggle.com/c/diabetic-retinopathy-detection> (accessed May 14, 2022).
- [17] F. Zhuang *et al.*, “A Comprehensive Survey on Transfer Learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.
- [18] C. Cai *et al.*, “Transfer Learning for Drug Discovery,” *Journal of Medicinal Chemistry*, vol. 63, no. 16, pp. 8683–8694, Aug. 2020, doi: 10.1021/acs.jmedchem.9b02147.
- [19] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in Neural Information Processing Systems*, 2014, pp. 3320–3328.
- [20] Xilinx and Inc, “DNNDK User Guide,” *Unknown*, vol. 1327, pp. 1–147, 2019.
- [21] Avnet, “ZedBoard,” *Avnet*. <http://zedboard.org/product/zedboard> (accessed Jan. 04, 2021).
- [22] XILINX, “DPU for Convolutional Neural Network v3.0 DPU IP Product Guide,” vol. 338, 2019.
- [23] X. Wang, Y. Lu, Y. Wang, and W.-B. Chen, “Diabetic Retinopathy Stage Classification Using Convolutional Neural Networks,” in *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, IEEE, Jul. 2018, pp. 465–471. doi: 10.1109/IRI.2018.00074.
- [24] H. S. Sharma, A. Singh, A. S. Chandel, P. Singh, and P. A. Sapkal, “Detection of Diabetic Retinopathy Using Convolutional Neural Network,” *SSRN Electronic Journal*, pp. 1–8, 2019, doi: 10.2139/ssrn.3419210.

BIOGRAPHIES OF AUTHORS






Meriam Dhouibi    (1989) a Ph.D student at the Department of Electrical Engineering at the National Engineering School of Carthage. She is doing her research work at the Systems Lab Advanced (LSA) in Tunisia Polytechnic School. In 2018 she started Teaching design of Embedded Systems and Advanced on chip architecture at the Department of Industrial Information Technology at the Higher Institute of Information and Communication Technologies. In 2014 she had her first master's degree in Embedded Electronic Systems and Medical Equipments. In 2017 she completed her second master's degree in Biophysics. Both from the Higher Institute of Medical Technologies of Tunis (ISTMT). She can be contacted at email: meriam.dhouibi@ept.mu.tn.






Ahmed Karim Ben Salem    received the engineer and master degrees in electrical engineering (industrial computer and automatic) from the National Institute of Applied Science and Technology (INSAT), in 2004 and the Ph.D degree in the same field and specifically in embedded systems in 2011 from INSAT. Since 2006, he has been a professor assistant in the Electrical Engineering Department of Higher Institute of Medical Technologies of Tunis (ISTMT), in Tunisia. Currently, he is pursuing his research at advanced system laboratory in Tunisia Polytechnic School (EPT). His research interests include embedded systems, real time system on chip co-design and performance evaluation. He is also working on hardware accelerators for artificial intelligence algorithms. He can be contacted at email: ahmed.bensalem@istmt.utm.tn.



Afef Saidi    Ph.D. student in Electrical Engineering at the University of National Engineering School of Carthage. She is a research member at Laboratory of Advanced Systems of Polytechnic School of Tunisia. She has a master degree in Biophysics and Medical Imaging in Higher Institute of Medical Technologies of Tunis. Since 2018, she started as teaching assistant in the Higher Institute of Information and Communication Technologies of Tunisia at the Department of Industrial Information Technology. Her research centers on the solutions for the implementation of classification techniques based on parallel architectures and reconfigurable platforms. She can be contacted at email: afef.saidi@ept.rnu.tn.



Slim Ben Saoud    received the electrical engineer degree from the High National School of Electrical Engineering of Toulouse/France (ENSEEIH) in 1993 and the Ph.D. degree from the National Polytechnic Institute of Toulouse (INPT) in 1996. He joined the department of Electrical Engineering at the National Institute of Applied Sciences and Technology of Tunis (INSAT) in 1997 as an Assistant Professor. He is now professor and the leader of the “embedded systems design group” at INSAT. His research interests include embedded systems architectures, real time solutions and applications to the co-design of digital control systems and space wire modules. He can be contacted at email: slim.bensaoud@gmail.com.